Polibits

ISSN: 1870-9044

polibits@nlp.cic.ipn.mx

Instituto Politécnico Nacional

México

Herrera Lozada, J. C.; Rivera Zárate, I.; Olguín Carbajal, M.

Computadoras de Bolsillo como una Alternativa para el Control de Servomotores en Robótica

Polibits, vol. 38, 2008

Instituto Politécnico Nacional

Distrito Federal, México

Computadoras de Bolsillo como una Alternativa para el Control de Servomotores en Robótica

J. C. Herrera Lozada, I. Rivera Zárate y M. Olguín Carbajal

***Resumen*—Este trabajo muestra el diseño de un sistema básico de control para servomotores convencionales, implementado en una computadora de bolsillo. La particularidad de esta realización radica en la interfaz hardware conformada por un microcontrolador que conecta al respectivo servomotor con el PDA a través del puerto serie de éste último. El sistema es de propósito general y se puede adaptar sin cambios drásticos a cualquier aplicación similar.**

***Palabras clave*—Servomotores, computadoras de bolsillo, robótica.**

PDA COMPUTERS AS AN ALTERNATIVE FOR SERVO MOTORS CONTROL IN ROBOTICS

***Abstract*—This paper presents a system that allows for control of conventional servo motors using PDA computers. The advantage of the proposed implementation is related to hardware interface, namely, to the usage of the specialized microcontroller that connects PDA with the servo motor using serial port of the PDA. The system can be easily adapted to other similar applications.**

***Index Terms*—Servo motors, PDA computers, robotics.**

I. INTRODUCCIÓN

Considerando la creciente proliferación de la tecnología de los dispositivos móviles, en específico la tendencia al uso de teléfonos celulares y sobre todo las computadoras de bolsillo o de mano, también conocidas como PDAs (en inglés, *Personal Digital Assistant*, Asistente Digital Personal), es posible afirmar que este tipo de dispositivos son elementos indispensables en la informática contemporánea para el intercambio y proceso de información. Entre muchas otras aplicaciones, en este trabajo se propone su adecuación como lazo primario de control en la supervisión de procesos.

Esta es la continuación de un proyecto que inició con la adquisición de datos para utilizar el PDA como un monitor de señales [1]; ahora, la intención es demostrar de manera sencilla cómo se pueden enviar datos hacia un microcontrolador y que éste pueda controlar la posición del rotor de dos servomotores independientes entre sí, demostrando una alternativa sustentable dirigida hacia el área de la robótica supervisada.

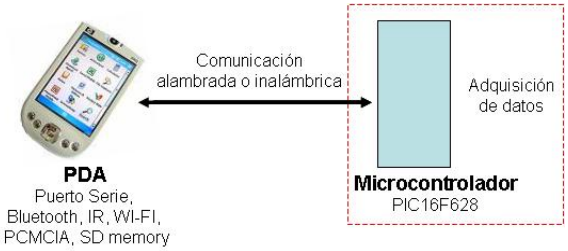

Fig. 1. Prototipo general de adquisición y monitoreo.

En un caso formal de diseño, será importante el análisis detallado de los parámetros y las condiciones del proceso a controlar con la intención de optimizar el sistema como lo demanda la teoría del control [2], [3], [4].

A. Servomotor

Los servos son una derivación de los motores a CD, estos se caracterizan por su capacidad para posicionarse de forma inmediata y exacta dentro de su intervalo de movimiento al estar operando. Para lo anterior, el servomotor espera un tren de pulsos; cada uno de estos pulsos tiene una duración específica para mover el eje de rendimiento del servomotor hacia una posición angular determinada. Se dice que el tren de pulsos es una señal codificada; por lo que cuando ésta cambia en el ancho de sus pulsos, la posición angular del eje también cambia.

Para comprender mejor el funcionamiento de estos dispositivos electromecánicos, obsérvese la Fig. 2, en donde se aprecian las señales que permiten el movimiento de un servomotor estándar.

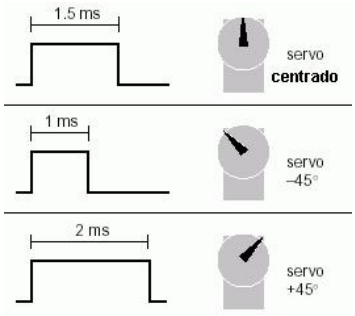

Fig. 2. Movimiento de un servomotor estándar.

Manuscrito recibido el 20 de agosto del 2008. Manuscrito aceptado para su publicación el 3 de noviembre del 2008.

J. C. Herrera Lozada, I. Rivera Zárate, M. Olguín Carbajal, CIDETEC-IPN, México D. F (e-mail:irivera@ipn.mx).

B. Computadora de Bolsillo (PDA)

En la particularidad de esta realización se utilizó una computadora de mano *iPAQ Pocket PC* de *Compaq*, modelo *3950*, con sistema operativo *Windows Pocket 2002* precargado de fábrica. Este sistema operativo es una versión más de la plataforma *Windows CE* (*Compact Edition*) La Tabla I muestra otras prestaciones importantes del equipo.

TABLA I.
CARACTERÍSTICAS DE LA POCKET PC 3950

Procesador @Velocidad	**Conectividad integrada**	**SDRAM @FLASH ROM**	**Resolución pantalla táctil**
Intel PXA250 @400MHz	USB, Serial, IrDA	64MB @32MB	240 x 320 pixeles, Transflective TFT, 65000 colores

En dependencia a la plataforma hardware del PDA (características del procesador y de la memoria) y a su sistema operativo, se eligió *Embedded Visual Tools 3.0* como ambiente de desarrollo. Éste contiene *Microsoft Embedded Visual C++ 3.0* y *Microsoft Embedded Visual Basic 3.0*, con los *kits* de desarrollo (*SDKs*) para *Pocket PC 2002* y *Smartphone 2002*.

Es importante mencionar que los PDAs más recientes, con prestaciones más sofisticadas incorporadas como por ejemplo las conectividades *Bluetooth* y *Wi – Fi*, se programan entre otras herramientas, con *Visual Basic.NET* si fuera el caso del sistema operativo *Windos Mobile* en sus versiones 5.0 y 6.0, o en otro caso, para una independencia del sistema operativo es posible acceder a la programación con *j2me* [6].

En el prototipo planteado, el grueso del procesamiento lo realiza el microcontrolador por lo que también es posible considerar el uso de alguna *hyperterminal* que permita el acceso a los puertos del PDA, para evitar el programar una interfaz de usuario, aunque este aspecto realmente depende de la complejidad de la aplicación misma.

II. DESARROLLO DE LA APLICACIÓN

La interfaz que permite la conexión con el PDA consta de dos módulos principales: un microcontrolador y un programa residente escrito en *Embedded Visual Basic* que controla el puerto serie del PDA para interactuar con el microcontrolador [1], este programa residente es el que podría ser sustituido por la *hyperterminal*, si fuera el caso.

Para la comunicación serial se requirió construir un cable *Null – Modem* de sólo 3 hilos, interconectando las señales sobrantes en el mismo conector DB9, tal y como se aprecia en la Fig. 3.

Este procedimiento emula el protocolo CTS/RTS y DSR/DTR por hardware; para controlar el flujo de datos se recurre al protocolo software XON/XOFF.

Todo el proceso se resume en sincronizar RXD en el PDA con la señal TXD a la salida del microcontrolador; a la vez, TXD del PDA con RXD del microcontrolador.

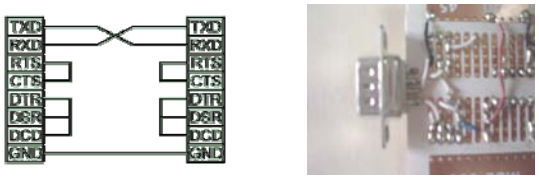

Fig. 3. Cable Null-Modem de 3 hilos.

El prototipo construido incluye un conector DB9 que se une con su contraparte de la cuna de sincronización (*cradle*) del PDA. Obsérvese en la Fig. 4, que en la tablilla que contiene al microcontrolador, las conexiones hacia el DB9 se controlan a través de *jumpers* con la finalidad de concebir otras configuraciones sin realizar cambios significativos en el circuito.

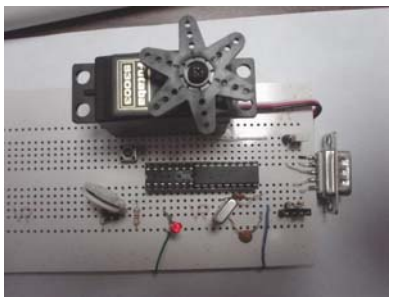
Fig. 4. Prototipo con PIC16F628.

Originalmente, este prototipo se diseñó para soportar un PIC16F73 que contiene internamente 4 canales de conversión A/D en correspondencia a las necesidades del proyecto anteriormente referido [1]; en el caso del PIC16F628 se desconectó el oscilador externo a través de los mismos *jumpers* dado que este dispositivo incorpora un oscilador interno. El prototipo resulta compatible para ambos microcontroladores.

El microcontrolador *PIC16F628* de *Microchip* es un dispositivo CMOS FLASH de 8 bits con arquitectura RISC, capaz de operar con frecuencias de reloj hasta de 20 MHz. Posee internamente un oscilador de 4 MHz y un circuito *Power-On Reset.* Ofrece dos puertos de datos con un total de 16 líneas I/O de propósito general.

Adicionalmente, el PIC16F628 proporciona una memoria de datos EEPROM de 128x8, una memoria de programa FLASH de 2024x14, una memoria de datos RAM de propósito general de 224x8, un módulo de captura/comparación/PWM, un USART, 2 comparadores análogos, una referencia de voltaje programable y tres temporizadores. Se recomienda revisar la hoja de especificaciones de este circuito integrado para complementar la información.

III. INTERFAZ DE USUARIO

Como se mencionó con anterioridad las interfaces para acceso al puerto serie de la iPAQ 3950 se programaron en *Embedded Visual Basic*. Se utilizó el control MSCOMM (control tipo Active X) con la opción *a disparo*, es decir, al depositar tanto para recibir como para enviar datos.

En el caso de recibir datos provenientes del microcontrolador, un byte en el buffer del puerto automáticamente dispara el evento correspondiente.

MSCOMM incorpora todas las funciones para configurar el puerto; sus propiedades más importantes son las siguientes:

ComPort: Activa y regresa el número del puerto serial (Comm1, Comm2).

PortOpen: Activa y regresa el acceso al puerto.

Input: Regresa los caracteres del buffer receptor.

Output: Escribe una cadena sobre el buffer Transmisor.

Settings: Activa y regresa la razón de Baudios, paridad, número de bits, bits de paro. En el caso particular de este trabajo se configuró la cadena *2400, n, 8, 1*, con *Handshaking* puesto a cero, debido a que no se realiza ningún control sobre el flujo de información.

Dentro del programa escrito para este proyecto, se hace referencia al control MSCOMM a través del objeto declarado como *Comm1*. Una aproximación al procedimiento inicial del programa se lista a continuación.

```
Private Sub Adquirir_Click()
  Comm1.PortOpen = True
  Comm1.RThreshold = 1
  Comm1.SThreshold = 0
  Comm1.InputLen = 0
  Comm1.DTREnable = False
End Sub
```

El objeto *Comm1* responde al evento *OnComm*, el cual genera una interrupción, indicando cuando hay comunicación o si algún error ha ocurrido en la transferencia de la información.

Una vez abierto el puerto, se procede a interactuar con el microcontrolador enviando y recibiendo palabras de datos. Las instrucciones básicas se listan a continuación, sólo de manera indicativa.

```
Comm1.Output = DatoEnviar.Text & vbCr
ValorLeido = Comm1.Input
```

La Fig. 5 muestra la interfaz para los servomotores, tanto en tiempo de diseño como en tiempo de ejecución. En esta aplicación se utiliza un monitor de mensajes y una caja de texto para validar una acción. Así mismo, es posible seleccionar el servomotor a operar.

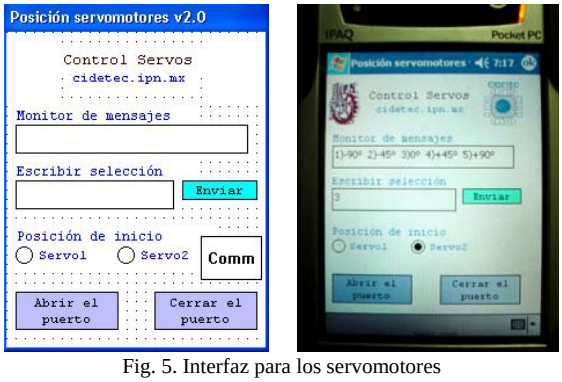

Fig. 5. Interfaz para los servomotores en tiempo de diseño y en ejecución

Se puede apreciar en ambos casos que es posible abrir y cerrar el puerto en cualquier momento para controlar la comunicación con el microcontrolador.

IV. Servomotores y Programación del Microcontrolador

Los servomotores utilizados son de la marca *Futaba* modelo *s3003* (Fig. 6), con un rango de movimiento de 0º a 180º. Para posicionar el eje de este servomotor se requieren trenes de pulsos con anchos de 0.3 ms. y hasta 2.3 ms. para conseguir el máximo ángulo.

Fig. 6. Servomotor *Futaba s3003*.

El microcontrolador se encarga de recibir serialmente el dato proveniente del PDA en formato estándar binario (también podría enviarse en formato ASCII) con una velocidad predeterminada de 2400 baudios, sin paridad y con un bit de paro. A través de una palabra codificada es posible controlar las condiciones del servomotor, antes mencionadas.

El programa fuente escrito en lenguaje de alto nivel *PicBasic*, se lista parcialmente a continuación [7]. En éste, entiéndase *serin* como la instrucción que permite al microcontrolador recibir datos provenientes del puerto serie del PDA y *serout* como la contraparte que permite enviar datos del microcontrolador hacia el PDA. Para simplificar el código sólo se atañe un servomotor; el código se hace extensivo en el caso de los dos motores indicados en el diagrama de la Fig. 7. En este mismo diagrama, el monitor serial se sustituye por el puerto serie del PDA en cuestión. Las líneas 0 y 1 del puerto B se utilizan para controlar los servomotores.

```
aux = 230       'Con 1 Ms, extrema izq.
servo1: pause 500
PORTB.1 = 0
letra:
pause 1000
serout PORTA.2, N2400, ["Introduce Posición",13,10]
serout PORTA.2, N2400, ["1)-90º  2)-45º  3)0º  4)+45º
                                          5)+90º",13,10]
serin PORTA.0, N2400, tiempo
if (tiempo < $31 OR tiempo > $35) then
serout PORTA.2, N2400, ["Sólo valores entre 1 y 5",13,10]
pause 250
goto letra
else
call deco
aux = dato
pause 250
endif
envia:  if dato = 231 then goto regre
dato=dato + 1
pulsout PORTB.1, dato
pause 20
goto envia
regre:  if dato = aux then goto letra
```

```
dato = dato - 1
pulsout PORTB.1, dato
pause 20
goto regre

deco:
select case tiempo
Case "1":  dato=30  '30 x 10 uS, futaba s3003, 0.3 ms a 2.3 ms; considerando Osc de 4MHz
CASE "2":  dato=80
CASE "3":  dato=130
CASE "4":  dato=180
CASE "5":  dato=230
end select
```

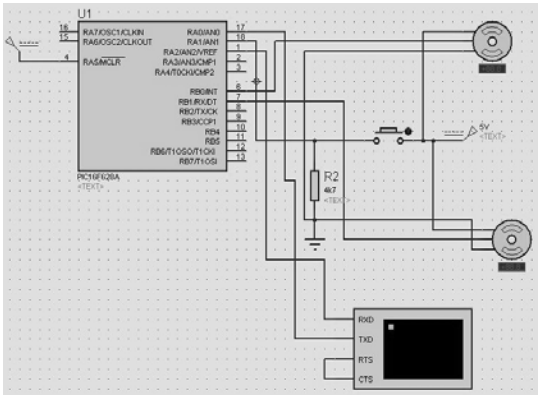

Fig. 7. Diagrama de conexiones.

V. Pruebas y Resultados

El microcontrolador con las rutinas seriales se utilizó primeramente con una PC de escritorio para verificar el funcionamiento correcto de la electrónica de la interfaz para los motores; posteriormente se comprobó el funcionamiento en el PDA ejecutando la aplicación generada en *Embedded Visual Basic* para *Windows Pocket 2002*. El sistema de control básico funcionó correctamente para el caso de un solo servomotor y para la situación extendida hacia dos servomotores.

En la Fig. 8 es posible apreciar al PDA empotrado en su cuna de sincronización y el prototipo en operación. El desarrollo final infiere un PIC16F628.


Fig. 8. Prototipo en operación con cuna de sincronización y PIC16F73 (sólo para comprobar compatibilidad).

También se realizaron pruebas con un PDA modelo *HP iPAQ hx2490b*, con sistema operativo *Windows Mobile 5.0*. En este caso, no se programó una interfaz de usuario, sino que se recurrió al uso de una *hyperterminal*. En la Fig. 9, se aprecia la *hyperterminal* en funcionamiento, así como la evolución del prototipo hacia un diseño más robusto con el PIC16F628.


Fig. 9. iPAQ hx2490b con Windows Mobile 5.0 y prototipo modificado (PIC16F628).

Se montó un pequeño laboratorio de pruebas, tal y como se puede apreciar en la Fig. 10, en el cual se utilizó un osciloscopio para medir los anchos de los pulsos generados por el PIC y que posicionan los ejes de los servomotores.

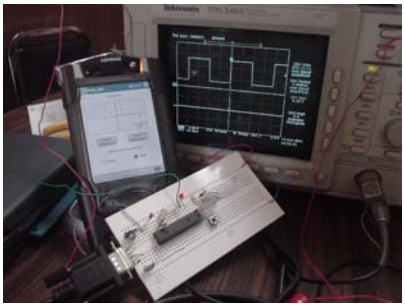
Fig. 10. Aspecto general del laboratorio de pruebas para el prototipo.

VI. Conclusiones

Como se menciona a lo largo de este trabajo, para algunas aplicaciones que no infieren alta complejidad de procesamiento, el PDA es un recurso sustentable para implementar sistemas simples de control, asumiendo que quien realiza el grueso del procesamiento es el microcontrolador, y la computadora de bolsillo se limita a la supervisón y monitoreo de resultados. En el CIDETEC, estas primeras aproximaciones se han aplicado con éxito en el control de aplicaciones robóticas con pocos grados de libertad, en donde cada grado está constituido propiamente por un servomotor.

La transmisión serial es sumamente confiable y sencilla de implementar; además de que actualmente se ha migrado hacia la comunicación inalámbrica utilizando el principio de la transmisión serial convencional, como sucede con el puerto *Bluetooth*, lo que aumenta los alcances futuros del prototipo

Uno de los principales objetivos planteados al inicio de este proyecto, fue el de utilizar el puerto serie de la *iPAQ Pocket PC* para recibir datos de un hardware externo. En el caso del sistema diseñado, el microcontrolador utilizado redujo drásticamente la complejidad de la comunicación entre el PDA y los servomotores; éste se conecta directamente a la iPAQ sin necesidad de un acoplador de nivel de voltaje (por ejemplo, el C.I. MAX232). De manera confiable se puede generar el

comando para habilitar el tren de pulsos que posiciona con exactitud el rotor de los motores.

REFERENCIAS

[1] J. C. Herrera Lozada, I. Rivera Zárate, A. Cruz Contreras. Monitor de Señales en un PDA. En: Proceeding XXVII International Congress of Electronic Engineering ELECTRO 2005. División de Estudios de Posgrado, ITCH, pp. 267 – 271.

[2] M. Mazo, J. Ureña, F. J. Rodríguez, J. J. García, F. Espinosa, J. L. Lázaro, J.C. García. Teaching Equipment for Training in the control of DC, Brushless and Stepper Servomotors. IEEE Trans. On Education, vol. 41, nº2, 1998, pp 146-158.

[3] P. Cominos, N. Munro. PID controllers: recent tuning methods and design to specification. In: IEEE Proceedings: Control Theory and Applications, vol. 149, no. 1, 2002, pp. 46–53.

[4] S. N. Vukosavic, M. R. Stojic. Suppression of torsional oscillations in a high-performance speed servo drive. IEEE Transactions on Industrial Electronics, vol. 45, no. 1, 1998, pp. 108 – 117.

[5] Nilas, P.; Sueset, T.; Muguruma, K. A PDA-based high-level human-robot interaction. In: Robotics, Automation and Mechatronics, 2004 IEEE Conference on Volume 2, 1-3 Dec. 2004, pp1158 – 1163.

[6] J. C. Herrera Lozada, J. C. González Robles, A. Cruz Contreras. Programación de Dispositivos de Cómputo Móviles. POLIBITS, Año XV, Vol. 1, Número 31, Enero – Junio de 2005. CIDETEC, México.

[7] J. Iovine. PIC Microcontroller Project Book. Mc. Graw Hill, 2004, 272 p.