COMPUTADORAS DE BOLSILLO COMO UNA ALTERNATIVA PARA EL CONTROL DE SERVOMOTORES EN ROBÓTICA

C. Lozada, I. Rivera, M. Olguín
CIDETEC, Instituto Politécnico Nacional

Resumen

Este trabajo muestra el diseño de un sistema básico de control para servomotores convencionales, implementado en una computadora de bolsillo (PDA). 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 este último. El sistema es de propósito general y se puede adaptar sin cambios drásticos a cualquier aplicación de robótica.

Palabras Clave: PDA, Computadora de Bolsillo, Servomotor, Microcontrolador, PIC16F628, Control, Robótica, Puerto Serie, Lógica Difusa.

Abstract

This paper presents the design of a basic control system for conventional servomotors, implemented on a handheld computer (PDA). The distinguishing feature of this implementation is the hardware interface formed by a microcontroller that connects the respective servomotor to the PDA through its serial port. The system is general-purpose and can be adapted without drastic changes to any robotics application.

Keywords: PDA, Pocket Computer, Servomotor, Microcontroller, PIC16F628, Control, Robotics, Serial Port.

1. 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** (*Personal Digital Assistant* — Asistente Digital Personal), es posible entrever a este tipo de dispositivos como elementos indispensables en la informática contemporánea para el intercambio y procesamiento de información.

La portabilidad que representa por sí mismo un aparato con estas características los predispone como equipos de cómputo versátiles y completos al incluir una pantalla táctil que permite introducir datos y visualizar resultados. Entre muchas otras aplicaciones, 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; 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. El microcontrolador utilizado es el **PIC16F628**, de bajo costo y uso general, que se consigue fácilmente en el mercado nacional, lo que propicia una alternativa sustentable que los estudiantes podrían adoptar para sus proyectos.

Cabe mencionar que no es la intención de este artículo profundizar en la teoría del funcionamiento de los motores indicados, por lo que se recomienda consultar bibliografía especializada en el tema [**?**].

El control simple aplicado en este desarrollo, solo para validar la propuesta, fue de tipo **PID con lazo cerrado**, socorrido en el control digital directo [**?**]. Como ya se mencionó con anterioridad, en este trabajo se aborda el envío de un tren de pulsos para posicionar el eje de dos servomotores controlados de manera individual.

1.1 Servomotor

Los servos son una derivación de los motores de CD; 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 del servomotor hacia una posición angular determinada. Se dice que el tren de pulsos es una señal codificada: cuando ésta cambia en el ancho de sus pulsos, la posición angular del eje también cambia.

Para comprender mejor el funcionamiento, obsérvese la figura 1, donde se aprecian las señales de un servomotor estándar.

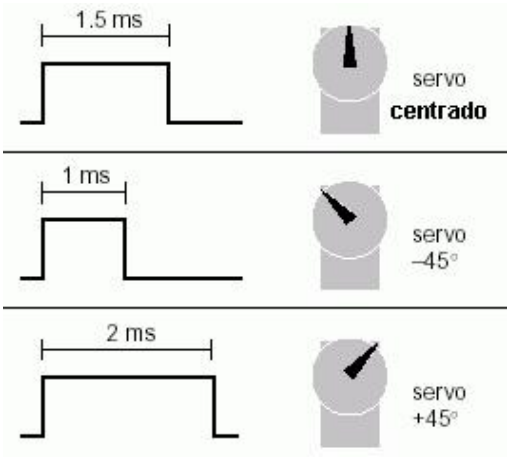

Figura 1 Movimiento de un servomotor estándar.

1.2 Agenda digital portátil (PDA)

Para la particularidad de esta realización se utilizó una computadora de mano **iPAQ Pocket PC de Compaq**, modelo 3950, con sistema operativo *Windows Pocket PC 2002* precargado de fábrica. Este sistema operativo es una versión de la plataforma *Windows CE* (*Compact Edition*). La tabla 1 muestra las prestaciones más importantes del dispositivo.

Cuadro 1 Características de la Pocket PC iPAQ 3950.

Característica	Especificación
Procesador @ Velocidad	Intel PXA250 @ 400 MHz
Conectividad integrada	USB, Serial, IrDA
SDRAM @ FLASH	64 MB @ 32 MB
Resolución pantalla táctil	240 × 320 píxeles, TFT transflectivo, 65 000 colores

Dada la 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*.

Para mayor referencia sobre la programación de aplicaciones en esta plataforma, consultar el trabajo complementario «Módulo de Comunicación Serial y Despliegue en una Computadora de Bolsillo», presentado en este mismo congreso. Es importante mencionar que los PDAs más actuales se programan, entre otras herramientas, con *Visual Basic .NET*.

2. 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 [**?**].

Para la comunicación serial se requirió construir un cable **Null-Modem de solo 3 hilos**, interconectando las señales sobrantes en el mismo conector DB9, tal y como se aprecia en la figura 2. 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, y viceversa.

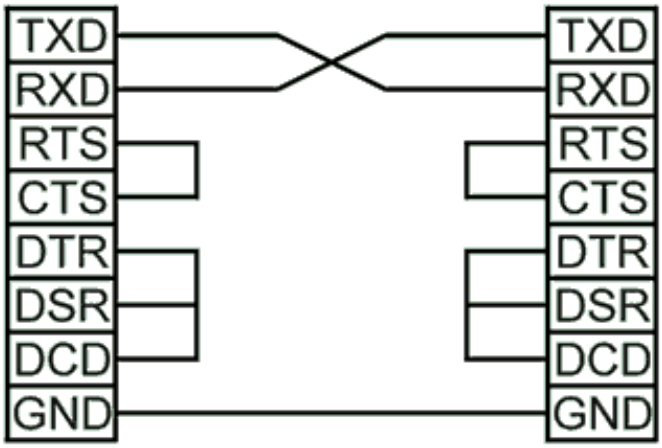

Figura 2 Cable Null-Modem de 3 hilos.

El prototipo construido incluye un conector DB9 que se une con su contraparte de la cuna de sincronización (*cradle*) del PDA. Obsérvese en la figura 3 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.

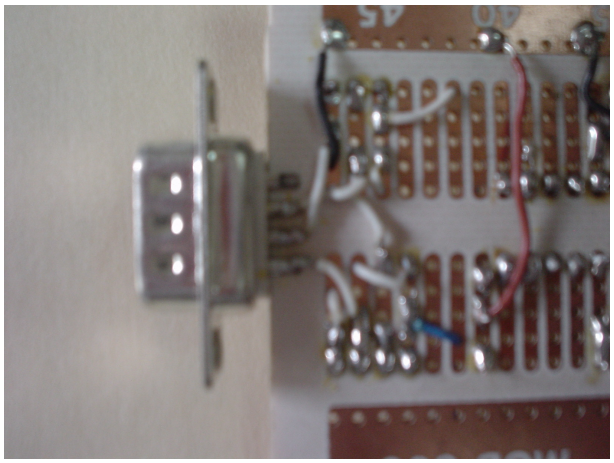

Figura 3 Tablilla del prototipo con PIC16F628.

Originalmente, este prototipo se diseñó para soportar un PIC16F73 que contiene internamente 4 canales de conversión A/D; 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.

En la figura 4 es posible apreciar al PDA empotrado en su cuna de sincronización y el prototipo en operación.

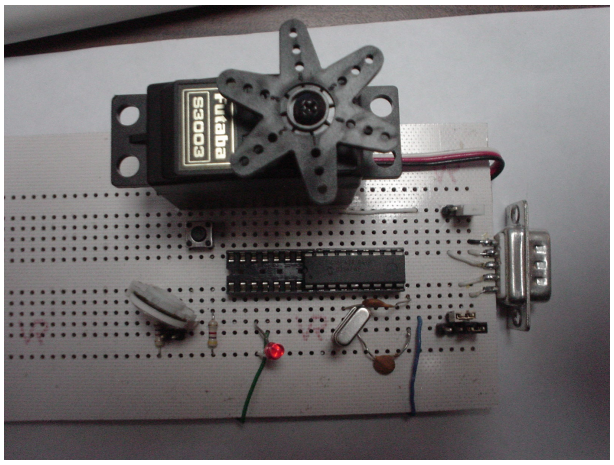

Figura 4 Prototipo en operación con cuna de sincronización y PIC16F73 (solo para comprobar compatibilidad).

El microcontrolador **PIC16F628** de Microchip es un dispositivo CMOS FLASH de 8 bits con arquitectura RISC, capaz de operar con frecuencias de reloj hasta 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 proporciona:

- Memoria de datos EEPROM de 128 $\times$ 8 bits.
- Memoria de programa FLASH de 2024 $\times$ 14 bits.
- Memoria de datos RAM de propósito general de 224 $\times$ 8 bits.
- Módulo de captura/comparación/PWM.
- USART, 2 comparadores analógos, referencia de voltaje programable.
- Tres temporizadores.

3. Interfaz de usuario

Las interfaces para acceso al puerto serie de la iPAQ 3950 se programaron en *Embedded Visual Basic*. Se utilizó el control **MSCOMM** (control tipo *ActiveX*) 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:

- `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 y bits de paro. En este trabajo se configuró la cadena `2400,n,8,1`, con *Handshaking* puesto a cero, dado 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`. A continuación se lista una aproximación al procedimiento inicial del programa:

```
1 Private Sub Adquirir_Click()
2     Comm1.PortOpen  = True
3     Comm1.RThreshold = 1
4     Comm1.SThreshold = 0
5     Comm1.InputLen   = 0
```

```
    Comm1.DTREnable  = False
End Sub
```

Listing 1 Inicialización del puerto serie en Embedded Visual Basic.

El objeto `Comm1` responde al evento `OnComm`, el cual genera una interrupción indicando cuándo 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 son:

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

Listing 2 Instrucciones de envío y recepción de datos.

La figura 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. Asimismo, es posible seleccionar el servomotor a operar y abrir o cerrar el puerto en cualquier momento.

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

4. Servomotores y programación del microcontrolador

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

Figura 6 Servomotor Futaba s3003.

El microcontrolador recibe 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.

El programa fuente escrito en lenguaje de alto nivel **PicBasic** [**?**, **?**] se lista parcialmente a continuación. `serin` es la instrucción que permite al microcontrolador recibir datos provenientes del puerto serie del PDA, y `serout` es la contraparte que permite enviar datos del microcontrolador hacia el PDA. Las líneas 0 y 1 del puerto B se utilizan para controlar los servomotores. El diagrama de conexiones se muestra en la figura 7.

```
aux = 230        ' Con 1 mS se posiciona a la extrema izq.

servo1: pause 500
        PORTB.1 = 0

letra:  pause 1000
        serout PORTA.2, N2400, ["Comienza Movimiento",13,10]
        serout PORTA.2, N2400, ["Introduce Posicion",13,10]
        serout PORTA.2, N2400, ["1)-90  2)-45  3)0  4)+45  5)+90",13,10]
        serin  PORTA.0, N2400, tiempo

        if (tiempo < $31 OR tiempo > $35) then
            serout PORTA.2, N2400, ["Solo 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    ' 0.3 ms - Futaba s3003 (Osc 4 MHz)
            CASE "2": dato = 80
            CASE "3": dato = 130
            CASE "4": dato = 180
            CASE "5": dato = 230
        end select
        RETURN
```

Listing 3 Programa PicBasic para el control de un servomotor (fragmento).

El *push button* es opcional para probar las conexiones en la circuitería. La línea 0 del puerto A recibe la palabra de control proveniente del PDA, y el bit 1 del mismo puerto A envía mensajes de respuesta hacia el PDA.

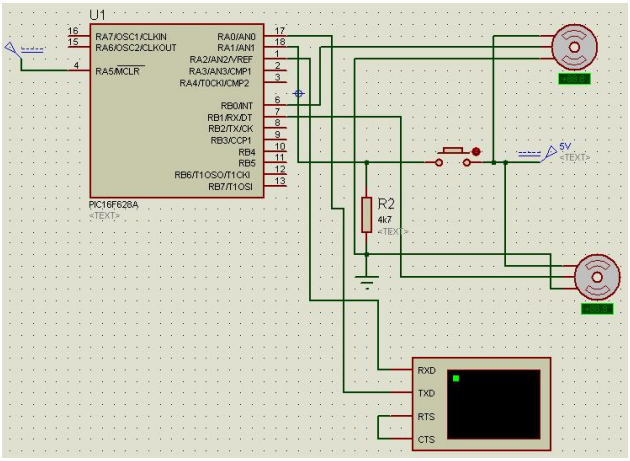

Figura 7 Diagrama de conexiones entre el PDA y los dos servomotores Futaba s3003.

5. 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*. El sistema de control básico funcionó correctamente para el caso de un solo servomotor y para una situación de dos servomotores.

Se montó un pequeño laboratorio de pruebas (véase figura 8), en el cual se utilizó un osciloscopio para medir los anchos de los pulsos generados por el PIC que posicionan los ejes de los servomotores.

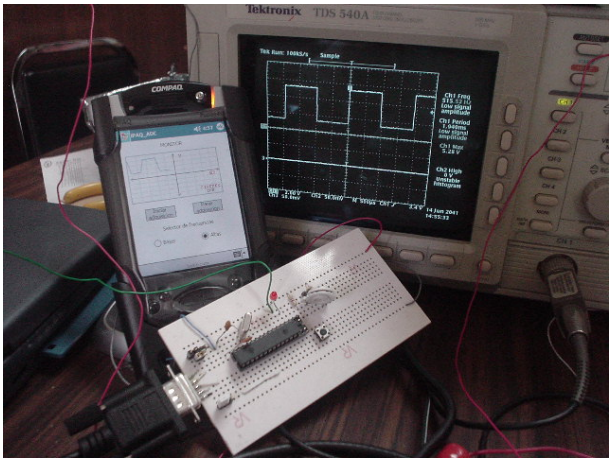

Figura 8 Aspecto general del laboratorio de pruebas para el prototipo.

6. Conclusiones

Como se menciona a lo largo de este trabajo, para algunas aplicaciones que no requieren altas frecuencias de trabajo, el PDA es un recurso sustentable para implementar sistemas de control. La transmisión serial es sumamente confiable y sencilla de implementar.

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.

El prototipo, por sí mismo, acepta las modificaciones pertinentes para adecuarse sin problemas a otras aplicaciones similares. La interfaz hardware soporta no solo al PIC16F628, sino también uno con mayores ventajas como el PIC16F73; además, es compatible con el puerto serie de una PC estándar y de un PDA de manera indistinta. Cabe mencionar que la interfaz software para el monitoreo en el PDA sí depende de la plataforma empleada.

Referencias

[1] 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 Transactions on Education*, vol. 41, núm. 2, pp. 146--158, mayo 1998.

[2] P. Cominos y N. Munro. ``PID controllers: recent tuning methods and design to specification''. *IEE Proceedings: Control Theory and Applications*, vol. 149, núm. 1, pp. 46--53, 2002.

[3] S. N. Vukosavic y M. R. Stojic. ``Suppression of torsional oscillations in a high-performance speed servo drive''. *IEEE Transactions on Industrial Electronics*, vol. 45, núm. 1, pp. 108--117, 1998.

[4] P. Nilas, T. Sueset y K. Muguruma. ``A PDA-based high-level human-robot interaction''. *Robotics, Automation and Mechatronics*, IEEE Conference, vol. 2, pp. 1158--1163, diciembre 2004.

[5] J. Herrera, J. González y A. Cruz. ``Programación de Dispositivos de Cómputo Móviles''. *POLIBITS*, año XV, vol. 1, núm. 31, enero--junio 2005. CIDETEC, México.

C. Lozada, I. Rivera, M. Olguín *(2026). Computadoras de Bolsillo como una Alternativa para el Control de Servomotores en Robótica. Boletín UPIITA. año XX, (NÚM) 2026.*