MONITOR DE SEÑALES EN UN PDA

Juan C. Herrera Lozada, Israel Rivera Zárate , Agustín Cruz Contreras
Centro de Innovación y Desarrollo Tecnológico en Cómputo (CIDETEC – IPN)
Av. Té 950 Edif. de Graduados UPIICSA 2° PISO, Col. Granjas México, C.P. 08400
Tel (915) 654-3932, 657-7453 Fax: (915) 624-2000 Ext. 70265
E-mails: {jlozada; irivera; acruz} @ ipn.mx

RESUMEN.
El presente trabajo muestra el diseño de un monitor de señales analógicas en una computadora de bolsillo. La particularidad de esta realización radica en la interfaz hardware que adquiere los datos y los envía hacia el PDA, vía el puerto serie. El monitor es de propósito general y se puede adaptar a cualquier aplicación similar.

1. INTRODUCCIÓN

En la actualidad, los dispositivos móviles representan un campo tecnológico fuerte y consolidado en cuanto a su infraestructura y aplicaciones. La telefonía móvil y sobre todo las computadoras de bolsillo, también conocidas como PDAs (*Personal Digital Assistant – Asistente Digital Personal*), resultan elementos indispensables en la informática contemporánea para el intercambio y proceso de información.
La utilización de un PDA como lazo primario de control, con etapas de adquisición de datos vía una interfaz hardware y el monitoreo de señales, permite definir líneas de investigación encaminadas a soluciones biomédicas y de supervisión de sistemas.

2. GENERALIDADES

De manera práctica, las características principales que diferencian entre sí a los PDAs, están definidas por las prestaciones siguientes: *sistema operativo, conectividad, memoria* y *procesador*.

2.1. Sistemas Operativos

Aunque existen otras alternativas, como la posibilidad de instalar *Linux*, la competencia en el mercado comercial de los PDA se libra entre las plataformas hardware que utilizan *Palm OS* y *Windows CE,* en cualquiera de sus diferentes versiones.

2.2. Conectividad

Los puertos de comunicación son los circuitos electrónicos que permiten el intercambio de información entre el PDA y otros dispositivos compatibles; en lo general, los principales puertos presentes en un PDA son: *USB, Serial, IrDA (infrarrojo), Bluetooth* y *Wi-Fi*. Los tres últimos reciben y transmiten datos de manera inalámbrica.

También es posible intercambiar datos utilizando las ranuras de expansión del PDA; esto se logra a través de una tarjeta que puede ser de tipo *Compact Flash*, *MS (Memory Stick), SD (Secure Digital), MMC (Multimedia Card)* o *SDIO Card (Secure Digital Input/Output)*, entre otras.

2.3. Memoria

En la memoria RAM de un PDA se almacenan archivos de datos del usuario y se ejecutan programas. Se trata de memorias *SDRAM* (*RAM Dinámica Síncrona*), donde a mayor cantidad de memoria es posible tener más aplicaciones abiertas al mismo tiempo o guardar más información.

La memoria ROM es la memoria principal del PDA, donde normalmente se almacena el sistema operativo. Comúnmente es de tipo *FLASH ROM* que permite actualizarse vía software. Actualmente, es posible encontrar PDAs con disco duro de hasta 4 GB.

2.4. Procesador

Una gran parte de los procesadores utilizados en los PDAs contemporáneos, están diseñados con tecnología *ARM*, que fundamenta un conjunto reducido de instrucciones (*RISC*) sobre arquitecturas avanzadas tipo *Harvard,* con un nivel de memoria caché independiente y palabras de datos de hasta 32 bits. Las velocidades actuales alcanzan los 400MHz.

3. CARACTERÍSTICAS DEL PDA UTILIZADO

En la particularidad de este trabajo se utilizó una computadora *iPAQ Pocket PC* de *Compaq*, modelo *3950*, con sistema operativo *Windows Pocket PC*

2002 precargado de fábrica. Este sistema operativo es una versión más de la plataforma *Windows CE* (*Compact Edition*) La tabla 1 muestra otras prestaciones importantes.

Tabla 1. 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 pixels, Transflective TFT, 65000 colores

El puerto serie de este dispositivo mantiene el protocolo RS-232 convencional y se sincroniza (para intercambio de archivos y supervisión) con una PC de escritorio a través del puerto USB.

3.1. Programación de Aplicaciones

Windows CE representa la evolución de la experiencia de *Microsoft* sobre dispositivos móviles y está basado en la interfaz de programación para aplicaciones Win32 (API). En la actualidad *Microsoft* dirige *Windows CE* no sólo hacia PDAs, sino también a *Smartphones*, videojuegos de última generación y computadoras de automóviles; por tal motivo, ha denominado *Windows Mobile* a la última versión de *Windows CE (NET)* cuya característica radica en unificar el sistema operativo de los dispositivos portátiles [2].

En dependencia a la plataforma hardware del PDA en cuestión y a su sistema operativo, se eligió *Embedded Visual Tools 3.0*, como ambiente de desarrollo [4]. 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*.

La decisión de utilizar *Embedded Visual Tools* estuvo determinada por cuatro puntos favorables [3]:

1. Su compatibilidad directa con *Windows*, al pertenecer al mismo fabricante.
2. Es de distribución gratuita, previo registro.
3. En el caso de *Embedded Visual Basic*, se tiene un lenguaje visual que es de fácil manejo.
4. No tiene dificultades para realizar la sincronización entre el PDA y la PC de escritorio, permitiendo la evaluación y depuración directa de programas.

4. DESARROLLO DE LA APLICACIÓN

La interfaz que permite introducir datos hacia el PDA consta de dos módulos principales: un microcontrolador que adquiere la señal analógica, la convierte a digital y en vía los datos de manera serial hacia el PDA; y un programa residente escrito en *Embedded Visual Basic* que controla el puerto serie del PDA para recibir los datos y graficar en pantalla la señal analógica adquirida. Véase la figura 1.

Figura 1. Diagrama a bloques de la interfaz.

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 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. En la tabla 2 se detallan las señales características del protocolo RS-232.

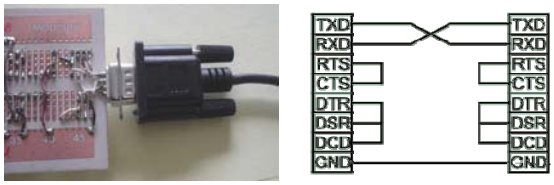

Figura 2. Cable Null-Modem de 3 hilos.

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

Tabla 2. Señales RS-232 para conector DB9.

Señal RS-232	Abrev	Dirección
Tierra GND		
Transmisión de datos	TXD	Salida
Recepción de datos	RXD	Entrada
Petición para enviar	RTS	Salida
Limpiar para enviar	CTS	Entrada
Terminal de datos lista	DTR	Salida
Conjunto de datos listo	DSR	Entrada
Detección de portadora de datos	DCD	Entrada
Indicador de llamada	RI	Entrada

4.1. Programación del Microcontrolador

La señal analógica, previamente adecuada y acondicionada, se conecta a uno de los canales ADC del microcontrolador PIC16F73 [6], específicamente el pin número 2 del diagrama de la

figura 3, para que éste realice la conversión analógico - digital a una frecuencia programada de 50Hz (a razón de un periodo de 20 ms).

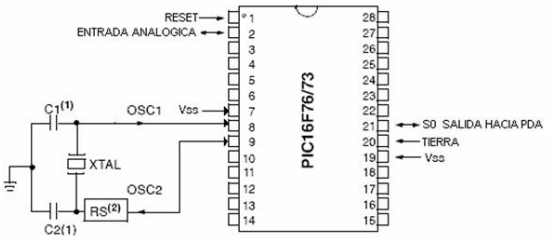

Figura 3. Diagrama de conexiones.

El microcontrolador se encarga de trasmitir serialmente el dato en formato binario de 8 bits (también podría enviarse en formato ASCII) hacia el PDA con una velocidad predeterminada de 2400 baudios, sin paridad y con un bit de paro.

El programa fuente escrito en lenguaje de alto nivel *PicBasic* [5] se lista a continuación. El registro ADCON0 se utiliza para configurar el puerto A como entrada y ADCON1 para configurar el módulo ADC. Obsérvese que de manera directa en código se establece el tiempo de la conversión analógica – digital y las características de la transmisión serial.

```
Symbol  ADCON0 = $1F    ' Dirección ADCON0
Symbol  ADRES = $1E      ' Dirección resultado
Symbol  ADCON1 = $9F    ' Dirección ADCON1
Symbol SO = 0                ' Salida Serial

poke ADCON1, 0     ' Puerto A como entradas analógicas
poke ADCON0, $41    ' Selecciona ADCON0, canal 0

Loop: poke ADCON0, $45   ' Inicia conversión
   pause 20             ' Espera 20 ms para Conversión
   peek ADRESS , B0   ' Asigna resultado al registro B0
   serout SO, N2400, (B0)  ' Envía dato serie hacia SO
   goto Loop          'Ciclo continuo
End
```

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

Se incluyó una resistencia variable que permite introducir una señal de prueba entre 0 y 5 Volts; se tiene un *jumper* adicional para multiplexar entre esta señal y otra proveniente del exterior (sensor, generador, etc.).

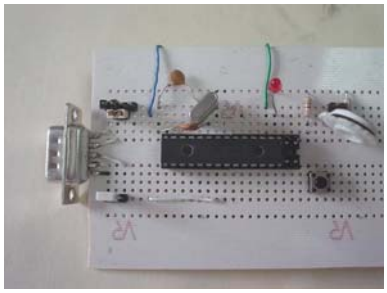

Figura 4. Aspecto del prototipo de adquisición y comunicación por puerto serie.

4.2. Interfaz Gráfica

La interfaz para acceso al puerto serie de la iPAQ 3950 y graficación de la señal, se programó en *Embedded Visual Basic*. Se utilizó el control *MS COMM* con la opción a *disparo*, es decir, al depositar un byte en el buffer del puerto automáticamente se dispara el evento correspondiente. Otra opción es realizar un poleo al buffer del puerto, cada determinado tiempo, buscando el byte recibido.

MS COMM 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 MS COMM 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
```

```
pantalla.ScaleHeight = 14
pantalla.ScaleLeft = 0
pantalla.ScaleTop = 0
pantalla.ScaleWidth = 12
pantalla.DrawWidth = 1
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.

La propiedad *RThreshold* establece y regresa el número de caracteres a recibir antes de que se genere el *CommEvent* ligado *a OnComm*. Con *RThreshold* puesto a 1, se habilita *CommEvent* en espera de un byte en el puerto.
SThreshold establece el número mínimo de caracteres a recibir en el buffer de transmisión antes de que se genere el *CommEvent*; colocando en 0 a esta propiedad se inhabilita la transmisión de datos hacia fuera del puerto del PDA. *InputLen* establece el número de caracteres a leer del buffer de datos; estableciéndola en 0 se lee entero el registro de datos recibidos. *DTREnable* habilita o deshabilita la señal *DTR* en el puerto; colocándolo en 0 se deshabilita esta señal.
El objeto *pantalla* es propio del despliegue gráfico a generar.

El procedimiento principal que controla el puerto del PDA y permite graficar se lista a continuación. En éste se realiza una comparación a través de un *case* para saber qué evento se ha generado en *CommEvent*, en caso de validarse un byte recibido (*comEvReceive*), se lee el dato para graficarlo.; en caso de no ser así, el programa no hace nada debido a que desde el inicio se deshabilitó la transmisión del PDA hacia el PIC 16F73. Si se desea utilizar el PDA como lazo primario de control, será importante considerar el envío y recepción de datos de manera mutua. El conector DB9 utilizado en el prototipo, está alambrado correctamente para soportar la bidireccionalidad de los datos en el protocolo.

```
Private Sub Comm1_OnComm()
Select Case Comm1.CommEvent
Case comEvReceive
bandera = bandera + 1
If bandera = 1 Then
pantalla.DrawPoint iniciox, inicioy, vbYellow
contador = contador + 1
   Else
InputData = Comm1.Input
Cantidad = (Asc(InputData) * 0.0196) + 2
inicio2x = iniciox + incremento
pantalla.DrawLine iniciox, inicioy, inicio2x, Cantidad, vbYellow
iniciox = inicio2x
inicioy = Cantidad
contador = contador + 1
If contador = cuenta Then
contador = 0
pantalla.Cls
iniciox = 0
bandera = 0
End If
End If

Case comEvSend
End Select

End Sub
```

La figura 5 muestra la interfaz de usuario diseñada. Se puede apreciar que se establecieron opciones para graficar frecuencias bajas, medias y altas. A la vez, se adicionaron botones para iniciar y detener la adquisición de datos. Para los ejes, se diseñó un mapa de bits que se carga al momento de abrir la aplicación.

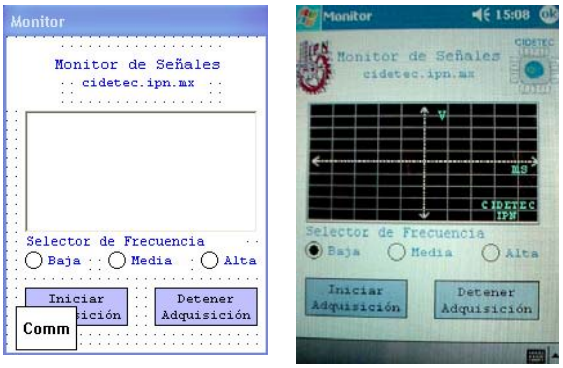

Figura 5. Aplicación en tiempo de diseño y en tiempo de ejecución.

5. PRUEBAS Y RESULTADOS

Con ayuda de un generador, se probó la etapa de adquisición introduciendo señales senoidales, cuadradas y triangulares con amplitud de 0.1 a 5 Volts con un rango de frecuencia menor a 50 Hz (frecuencia de conversión programada en el PIC); las señales fuera de este rango, no se graficarían correctamente. Los resultados obtenidos de esta prueba fueron completamente satisfactorios.

Las figuras 6 y 7, muestran el despliegue de una señal y su respectiva comparación con la obtenida en un osciloscopio.

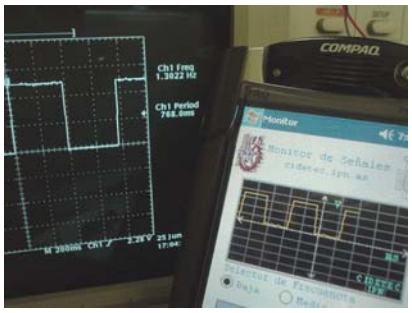

Figura 6. Comparando señales entre el PDA y el osciloscopio. Onda Cuadrada 1.38 Hz.

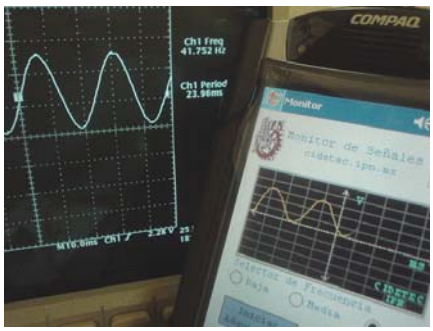

Figura 7. Comparando señales entre el PDA y el osciloscopio. Onda Senoidal 41.7 Hz.

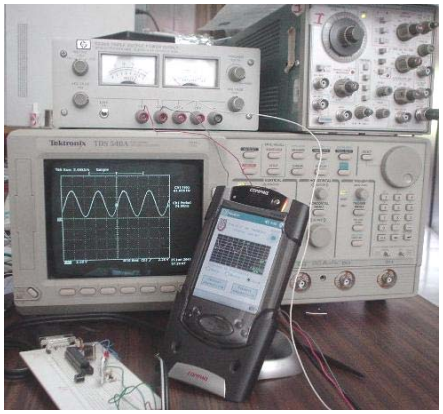
Figura 8. Aspecto general del proyecto.

6. CONCLUSIONES

Para alguna s aplicaciones que no re quieren altas frecuencias de traba jo, el PDA es un recurs o sustentable pa ra m onitorear y controlar sis temas. La transm isión serial es suma mente confiable y sencilla de implementar.

Uno de l os pri ncipales o bjetivos pl anteados al inicio de este p royecto, fue el de utilizar el puerto serie de la iPAQ Pocket PC para recibir datos de un hardware e xterno. L o ant erior con el propósito de no requerir c haquetas de e xpansión e xtras para insertar tarjet as Com pact Flash o del tipo PCMCIA.

En el caso del monitor portátil d iseñado, la principal restricción está marcada por la poca memoria q ue i ncluye el PDA utilizado (64 MB en RAM) por lo que no es del todo viable almacenar datos consta ntes, a demás de que se tendría l a necesidad de reducir el tie mpo de conversión para facilitar el proceso.

Al mantener abierto el puerto y a la vez ejecutar la aplicación, es posi ble saturar el proces o propiciando un f uncionamiento erró neo (inestabilidad). El co ntrol MSCOMM u tilizado para recibir datos del puerto se activa cada vez que se coloca un byte en el buffer del puerto, por lo que no es conve niente realizar varias operaciones durante la ejecución de dicho procedimiento.

El microcontrolador utilizado redujo drásticamente la complejidad de la co municación entre el PDA y el co nvertidor interno; lo s d atos en tregados p or el microcontrolador se i ntroducen directam ente a la iPAQ si n nec esidad de u n acoplador de nivel de voltaje (por ejemplo, el C.I. MAX232). La interfaz hardware utilizada es exten siva para o tras aplicaciones sin ca mbios drásticos en la programación del microcontrolador.

7. BIBLIOGRAFÍA

[1] Ceballos, Fra ncisco J. ***Visual Basic 6 Curso de Programación.*** Alfa omega Ra-Ma. México. 2000. ISBN 97015-0447-X.

[2] ***eMbedded Visual Basic Online Documentation.*** http://www.microsoft.com/windows/embedded/

[3] ***Introduction to Pocket PC 2002 Programming.*** http://www.devhood.com/tutorials/tutorial_details.aspx?tutorial_id=272

[4] **eMdedded Visual Tools 3.0, Descarga.** http://www.microsoft.com/mobile/downloads/emvt30.asp

[5] Frino , Luis. ***PicBasic Compiler, Manual en Línea.*** www.frino.com.ar/

[6] ***Hoja de especificaciones del microcontrolador PIC16F73.*** www.microchip.com