UNIDAD BÁSICA DE COMUNICACIÓN SERIAL EN UN MICROCONTROLADOR

Juan Carlos Herrera Lozada, Profesor – Investigador del CIDETEC IPN.
Ian Ilizaliturri Flores, Alumno de la Maestría en Tecnología de Cómputo del CIDETEC IPN.
E-mails: jcrls@ipn.mx; ferroian@hotmail.com

RESUMEN
El presente artículo infiere el diseño de una unidad de comunicación serial implementada en un microcontrolador. La funcionalidad de este prototipo permite enviar y recibir datos de cualquier sistema de cómputo que soporte el estándar RS-232; éste es genérico pudiendo adaptarse a cualquier aplicación sin cambios drásticos. Este trabajo expone los resultados preliminares del proyecto *"Sistemas de Control para Motores, Supervisados por Computadoras de Bolsillo"* que se desarrolla actualmente en el CIDETEC IPN.

1. INTRODUCCIÓN

Los microcontroladores actuales, por lo general, cuenta con un modulo funcional de comunicación serie *USART/SCI (Universal Synchronous Asynchronous Receiver Transmitter/Serial Comunication Interface)* o en otro caso éste puede ser implementado en la mayoría de sus funciones mediante software. Este módulo es ampliamente utilizado en sistemas de cómputo en general, por lo que el microcontrolador visto como un *core* o unidad básica puede establecer comunicación de manera simple con el procesador del equipo y acceder a los recursos del sistema completo.

El medio de transmisión para la interfaz USART requiere cuando menos de dos vías por lo que la complejidad, costos y el uso de pines del microcontrolador se reducen; esta interfaz tiene dos modos de comunicación: asíncrona o síncrona. La primera es *full duplex* con una vía RX (Recepción) y TX (Transmisión), la segunda es *half duplex* utiliza una vía de CLK (reloj) y otra de DT (Transmisión de datos) con sus configuraciones por dispositivo como maestro o esclavo.
En los microcontroladores pueden existir otros módulos de comunicación serie como el *SPI (Serial Peripherial Interface)* o *I2C (Inter.-Integrated Circuit)* que son de tipo síncrono, y que de momento están fuera del alcance de este artículo.

Cuando el módulo USART sólo contiene o se maneja la comunicación en modo asíncrono se le denomina *UART (Universal Asynchronous Receiver/Transmitter)*. Para establecer comunicación mediante el UART se requiere que los parámetros sean los mismos para TX y RX; estos parámetros son: el número de bits del dato, el número de bits de paro *(stop)*, la velocidad de transmisión (en baudios), el control de paridad y los niveles lógicos utilizados.

La línea que transmite los datos en serie está inicialmente en estado alto; al comenzar la transferencia se envía un bit a "0" lógico ó *bit de inicio*. Tras él irán los 8 bits de datos a transmitir (en ocasiones son 7, 6 ó 5): estos bits están espaciados con un intervalo temporal fijo y preciso, ligado a la velocidad de transmisión que se esté empleando.
Tras ellos podría venir o no un bit de paridad que indica si se ha enviado un número par o impar de bits con un nivel lógico alto en la palabra; al final, aparecerá un bit (a veces un bit y medio ó dos bits) a "1" lógico, que serían los bits de paro, lo de medio bit significa que la señal correspondiente en el tiempo a un bit dura la mitad; realmente, en comunicaciones se utiliza el término baudio para hacer referencia a las velocidades y normalmente un baudio equivale a un bit por segundo. La presencia de bits de inicio y parada permite sincronizar la estación emisora con la receptora, haciendo que los relojes de ambas vayan a la par por eso el tipo de transmisión es llamada asíncrono por paquete y síncrono por bit.

Existen varios estándares de comunicación serie asíncrona, que emplean tensiones diferenciadas que son apropiadas para distancias largas entre dispositivos como RS-422, RS-485 u otros que emplean niveles de tensión para la inmunidad a ruido como RS-232. El UART tiene compatibilidad con este último en su tercera versión denominada con "C".

1.2 ESTÁNDAR RS-232C

Propuesto por la *Asociación de Industrias Electrónica (EIA)*, realizándose posteriormente una versión internacional por el *CCITT ((Consultative Committee for International Telegraphy and Telephony)* conocida como V.24 con pocas diferencias entre ambas. El estándar estipula las características mecánicas, eléctricas, y de un protocolo orientado al enlace físico punto a punto. Las características eléctricas ofrecen una alta inmunidad a ruidos y distancia de enlace de hasta 16 metros dependiendo de la velocidad de transmisión utilizada. Los niveles de voltajes aplicado para el 0 lógico están en el rango de +3V a +15V, y para el 1 lógico se tiene un rango de -3V a -15V; entre -3V a +3V se tiene una zona de transición. Este estándar cuenta con dos tipos de conectores físicos: el conector DB9 y el conector DB25. La *tabla 1* muestra la configuración física de los pines en ambos casos.

Tabla 1. Pines de los conectores DB9 y DB25.

Pin DB9	Pin DB25	Nombre	Dirección	Descripción
1	8	CD	Entrada	Detección de portadora
2	3	RxD	Entrada	Recepción de datos
3	2	TxD	Salida	Transmisión de datos
4	20	DTR	Salida	Terminal de datos preparadas
5	7	GND	-	Masa del sistema
6	6	DSR	Entrada	Set de datos preparados
7	4	RTS	Salida	Petición para enviar
8	5	CTS	Entrada	Listo para enviar
9	22	RI	Entrada	Indicador de llamada

2. DISEÑO DEL PROTOTIPO

El *core* diseñado está dispuesto como enlace entre el sistema de cómputo y una aplicación externa (monitoreo de señales, adquisición de datos, sistemas de control, etc.). El microcontrolador intercambia información con el equipo de cómputo vía el puerto serie. Por ejemplo, una PC común utiliza circuitos integrados UART en modelos como 8250, 16450, 16550, 16650, 16750; cada uno de estos con sus variantes en máxima velocidad de comunicación, elementos de almacenamiento, control de flujo de datos, funcionalidades adicionadas y en el cumplimiento de la norma RS-232C; por lo que adecuar los requerimientos de la unidad básica de comunicación serial (microcontrolador programado como tal) no representa mayor problema.

2.1 CONSTRUCCIÓN DEL CABLE.

Para la comunicación serial simplificada, considerando la robustez que pueda presentar el diseño, se requiere 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 1*.
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.

CIDETEC IPN, México. Revisión: Enero de 2009.
Juan Carlos Herrera Lozada
jlozada@ipn.mx

Figura 1. Cable Null-Modem de 3 hilos.

2.2 UNIDAD SERIAL IMPLEMENTADA EN MICROCONTROLADOR

Es posible utilizar cualquier microcontrolador considerando las posibles excepciones que se comentarán posteriormente en este mismo artículo. Para unificar la metodología se utilizó primeramente el microcontrolador *PIC16F628A* de *Microchip*, debido a que es un dispositivo de bajo coste y que se consigue fácilmente en el mercado nacional; este circuito integrado dispone entre otras bondades de 2048x14 bits en memoria de código, 224x8 bits en memoria de datos, 128x8 bits en memoria EEPROM, una frecuencia de trabajo máxima de 20MHz, 16 puertos E/S, dos comparadores analógicos, USART y módulo PWM de 10bits. El módulo USART incluido tiene una memoria FIFO, *BRG (Baud Rate Generador)* y control de paridad. Se recomienda acceder a la hoja de especificaciones de este microcontrolador para profundizar en detalles más consistentes.

2.2.1 PROGRAMACIÓN CON PICBASIC PRO

Para la programación del microcontrolador se utiliza *PICBASIC PRO* que es un lenguaje de alto nivel que disminuye considerablemente la complejidad y los tiempos de diseño. Este lenguaje contiene dos instrucciones básicas para la comunicación serial bajo un protocolo simple: *SERIN y SEROUT* (comunicación serial asíncrona para entrada/salida). El compilador, así como el manual de usuario y algunos tutoriales, se puede obtener directamente del sitio web del fabricante *http://www.melabs.com/pbpdemo.htm*.

La sintaxis de la instrucción de entrada serial *SERIN* es la siguiente:

SERIN Pin, Mode, {Timeout,Label,}{[Qual...],} {Item...}

recibe los datos sobre el pin indicado (de acuerdo al pinout del microcontrolador) en el formato estándar asíncrono que define utilizar 8 bits de datos, sin bit de paridad y un solo bit de paro *(8N1)*.

El modo selecciona la frecuencia de transmisión y el tipo de la misma de acuerdo a la tabla 2.

Tabla 2. Modos de transmisión aceptados por SERIN y SEROUT.

Symbol	Value	Baud Rate	Mode
T2400	0	2400	TTL True
T1200	1	1200	
T9600	2	9600†	
T300	3	300	
N2400	4	2400	TTL Inverted
N1200	5	1200	
N9600	6	9600†	
N300	7	300	
OT2400	8	2400	Open Drain
OT1200	9	1200	
OT9600	10	9600†	
OT300	11	300	
ON2400	12	2400	Open Source
ON1200	13	1200	
ON9600	14	9600†	
ON300	15	300	

CIDETEC IPN, México. Revisión: Enero de 2009.
Juan Carlos Herrera Lozada
jlozada@ipn.mx

Mode (*p.ej. T2400* para indicar el baudaje) está definidos en el archivo *MODEDEFS.BAS* .que se incluye utilizando la línea *Include "modedefs.bas"*

La sintaxis de la instrucción para la salida serial *SEROUT* es:

SEROUT Pin, Mode,[Item[,Item...]]

Envía los datos de manera serial a través del pin indicado, utilizando el mismo formato estándar de la instrucción *SERIN*.

Para utilizar el protocolo RS-232 completo con la interfaz USART (no sólo tres hilos, sino todos los del conector), es posible utilizar *HSERIN* y *HSEROUT*. Es este caso será necesario definir los parámetros de la comunicación y construir el cable adecuado.

3. APLICACIÓN CON COMUNICACIÓN COMPLETA.

En el circuito de la *figura 2* se puede apreciar el microcontrolador PIC16F628A conectado a un motor CD. El bloque denominado *TERMINAL* en la misma figura, es la interfaz de usuario que permite enviar datos hacia el microcontrolador y visualizar los datos recibidos. Esta terminal puede ser de propietario o es posible diseñar una en Visual Basic, en C, o en algún otro lenguaje que conceda acceso hacia el puerto serie. En este trabajo se utilizaron dos terminales serie, una para simulación (*contenida en la herramienta software de simulación, ISIS de Proteus*) y otra para pruebas físicas que se descargó de http://sites.google.com/site/terminalbpp/.

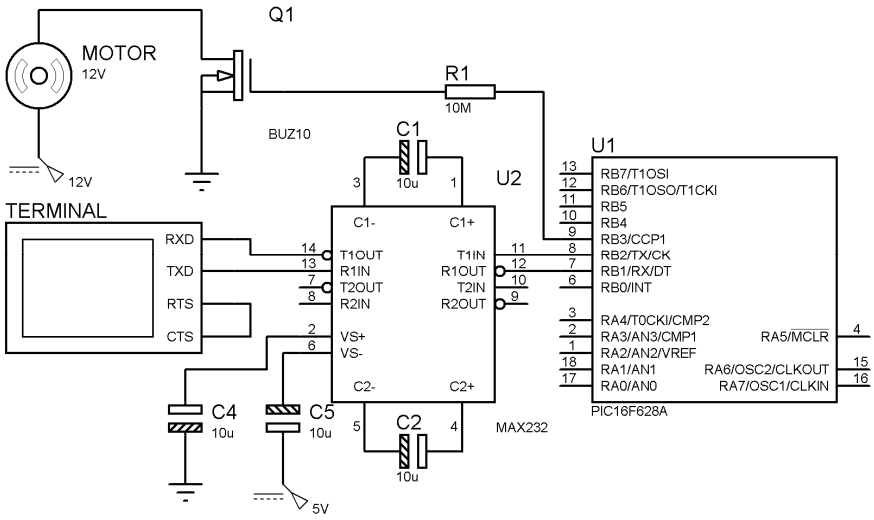

Figura 2. Control Simple de un Motor a CD vía el puerto serie. PIC16F628A.

En código programado destina tiempo de proceso para establecer parámetros iniciales de funcionamiento que se estipulan en correspondencia al manejo de una USART en protocolo completo. En este circuito, la velocidad de giro y el paro del motor pueden ser realizadas en cualquier momento mediante la terminal. Obsérvese que el control del motor se realiza a través de la instrucción HPWM que permite modular el ancho del pulso y obtener un equivalente analógico en voltaje que permite reducir o aumentar la velocidad del motor.

El circuito integrado *MAX232* permite acoplar los voltajes de la USART y el PIC (opción más recomendable), aunque los microcontroladores de *Microchip* permiten acoplar los voltajes sin necesidad de un dispositivo adicional.

```
Include "modedefs.bas"
       'BITS DE CONFIGURACION DEL MICROCONTROLADOR
@ DEVICE MCLR_OFF, INTRC_OSC, WDT_OFF,
```


CIDETEC IPN, México. Revisión: Enero de 2009.
Juan Carlos Herrera Lozada
jlozada@ipn.mx

```
@ DEVICE LVP_OFF, BOD_OFF, PWRT_OFF, PROTECT_OFF

       'CONFIGURACION DE TERMINALES
DEFINE HSER_BAUD 2400         'VELOCIDAD
DEFINE HSER_SPBRG 25
DEFINE HSER_BITS 8            'BITS DE DATOS

       'DECLARACION DE VARIABLES
RESPUESTA VAR BYTE

       'PROGRAMA PRINCIPAL
HSEROUT ["CONTROL DE VELOCIDAD PARA MOTOR DC",10,13]
INICIO: HSEROUT ["INTRODUZCA OPCIÓN DE VELOCIDAD 1 2 3, STOP 0",10,13]
HSERIN [str RESPUESTA \1\"A"]
HSEROUT [RESPUESTA,10,13]
SELECT CASE RESPUESTA
CASE 49        'Velocidad 1
    HPWM 1,85,245
case 50        'Velocidad 2
    HPWM 1,170,245
case 51        'Velocidad total
    HPWM 1,255,245
case 48        'Paro de motor
    HPWM 1,0,245
end select
GOTO INICIO
END
```

4. APLICACIÓN CON COMUNICACIÓN SIMPLIFICADA

El microcontrolador utilizado en el circuito de la *figura 3* es un *PIC12C508A* que cuenta con 768x12 bits en memoria de código, 25x8 bits en memoria de datos, frecuencia máxima de 4MHz y 6 líneas E/S y 8 pins. Este dispositivo no tiene módulos PWM y USART por lo que su funcionamiento debe ser programado en su totalidad y tomando tiempo de procesamiento para su utilización. En el caso de la función PWM sólo es utilizada para poder procesar la siguiente acción, en la comunicación no se utilizó el circuito de acoplamiento de voltajes por lo que el estado de transición en cero volts o menos en RS-232C es tomado en el microcontrolador como “0” lógico y cualquier voltaje arriba de 5 volts es tomado como “1” lógico, del lado del RS-232 los cero volts del microcontrolador son tomados como “1” lógico y los 5 volts equivalen a “0” lógico. El no adecuar el acoplamiento de voltajes aumenta los errores y reduce la velocidad en la comunicación. La comunicación en su totalidad realizada por software por ejemplo limita la verificación de error y la pérdida de datos a una velocidad menor que con módulos dedicados

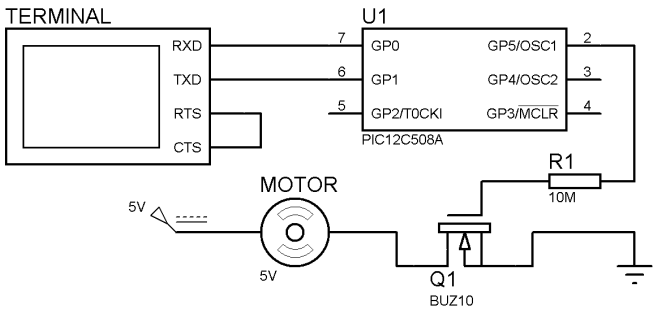

Figura 3. Control Simple de un Motor a CD vía el puerto serie. PIC12C508A.

```
Include "modedefs.bas"
@ DEVICE MCLR_OFF,INTRC_OSC,WDT_OFF,PROTECT_OFF

       'DECLARACIÓN DE VARIABLES
RESPUESTA var byte
VELOCIDAD VAR BYTE
```


CIDETEC IPN, México. Revisión: Enero de 2009.
Juan Carlos Herrera Lozada
jlozada@ipn.mx

```
TIEMPO    VAR BYTE

       'PROGRAMA PRINCIPAL
VELOCIDAD = 0
SEROUT GPIO.0,N2400,["CONTROL DE VELOCIDAD PARA MOTOR DC",10,13]
INICIO: SEROUT GPIO.0,N2400,["DEME LA VELOCIDAD 1 2 3 PARO POR TIEMPO",10,13]

SERIN  GPIO.1,N2400,[],RESPUESTA'Espera para respuesta
SEROUT GPIO.0,N2400,[RESPUESTA,10,13]
 SELECT CASE RESPUESTA
CASE 49        'Velocidad 1
    VELOCIDAD = 85
case 50        'Velocidad 2
    VELOCIDAD = 179
case 51        'Velocidad total
    VELOCIDAD = 255
end select
    FOR TIEMPO = 1 TO 5      'Micro 4MHz,5ms*255*5 = 6 seg
        PWM GPIO.5,VELOCIDAD,255
    NEXT TIEMPO
        PWM GPIO.5,0,1
SEROUT GPIO.0,N2400,["FIN DE PWM",10,13]
GOTO INICIO
END
```

5. CONCLUSIONES

La transmisión serial resulta fácil de implementar, además de que es segura y de bajo coste. En el caso de un microcontrolador como interfaz hardware de intercambio con cualquier sistema de cómputo (principalmente PC's,y computadoras de bolsillo), es viable considerar las propuestas realizadas en este trabajo, ya que se adaptan a cualquier aplicación similar, debido a que la unidad básica de comunicación serie es la misma en todo momento.

Se mostraron dos aplicaciones que utilizan el *core* diseñado, por un lado un microcontrolador de la gama media con un dispositivo adicional para el acoplamiento de voltajes para la transmisión; por otro, un microcontrolador de la gama baja y sin acoplamiento de voltajes, con lo que se tiene una referencia mayor para realizar otras aproximaciones *(ver figura 5)*.

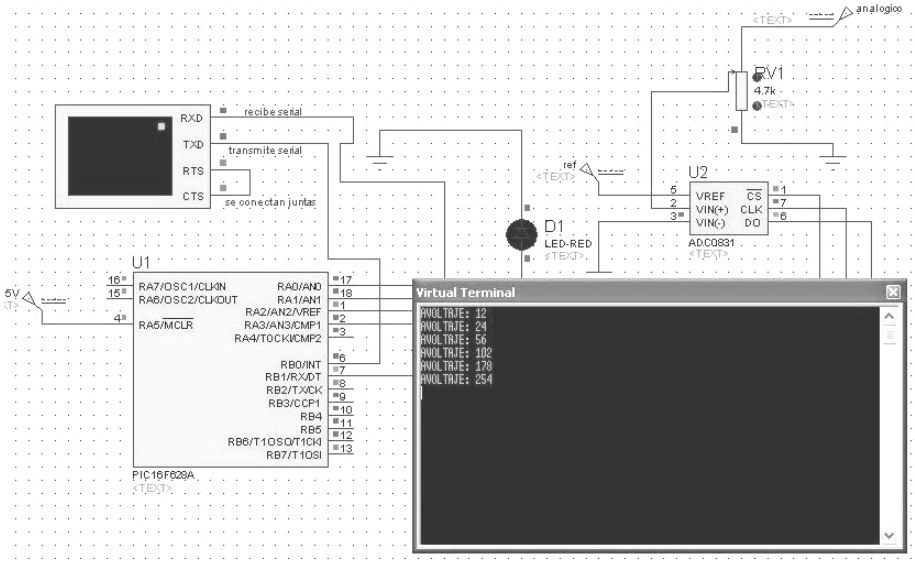

Figura 4. Aplicación que adquiere datos y los muestra.

La utilización de lenguaje PICBASIC PRO para la programación de microcontroladores facilita en gran medida el diseño, aunque no se debe olvidar que al momento de obtener a

CIDETEC IPN, México. Revisión: Enero de 2009.
Juan Carlos Herrera Lozada
jlozada@ipn.mx

través del compilador el ensamblador equivalente y posteriormente el archivo hexadecimal con el que se programará el microcontrolador, éste último implica más uso de la memoria del dispositivo comparado con un diseño total en ensamblador. Aún así el lenguaje de alto nivel resulta un recurso ampliamente socorrido por estudiantes y diseñadores profesionales.

Con respecto al proyecto que dio pie a este artículo *"Sistemas de Control para Motores, Supervisados por Computadoras de Bolsillo"*, cabe mencionar que la portabilidad y el poder de procesamiento de un *PDA (Asistente Personal Digital)* permiten concebir a estos dispositivos como sistemas altamente rentables para analizar datos en sitio, sin la necesidad de otros aparatos de monitoreo y medición. La propuesta es programar bajo *Microsoft Windows CE* (actualmente *Windows Mobile*) para estudiar la factibilidad de la implementación de sistemas de control, idealmente de motores a CD, motores a pasos y servomotores, controlados por la Pocket PC vía el puerto serie y protocolo RS-232.

Adicionalmente, se plantean pruebas utilizando la comunicación *RF* (radiofrecuencia) y la tendencia actual hacia *RFID* de identificación inalámbrica; ambas coinciden en el esquema de transmisión serial. La tecnología *RFID* reconoce códigos válidos a través de una señal de radiofrecuencia en un esquema de emisor - receptor wireless (inalámbrico). En la actualidad existen innumerables aplicaciones que van desde sustituir un lector de código de barras, hasta descontar saldos en cuentas (tarjetas del Metrobús y boletos del metro, de la Ciudad de México).

BIBLIOGRAFÍA

1. *PIC Microcontroller Project Book*
John Iovine
Segunda edición
Mc Graw Hill.

2. Jose M. Angulo Usategui
Microcontroladores PIC: primera y segunda parte
Tercera edición
Mc Graw Hill

3. *Manual de PIC16F628/628A/648*A
Microchip, Inc.

4. *Manual de PicBasic Pro Compiler*
MicrioEnginnering Labs. Inc.

5. http://www.cidetec.ipn.mx/pofesores/jcrls/

6. http://www.todopic.com.ar/pbp_sp/

Juan Carlos Herrera Lozada
jlozada@ipn.mx