TEMARIO

RS-232

INGENIERIA EN MICROCONTROLADORES

Tutorial del Protocolo RS-232

Teoría y Aplicaciones

PROTOCOLO RS-232

Elaborado por el Ing. Eric López Pérez

© Ingeniería en Microcontroladores
Teléfono 044 55 11 29 55 05 • E-mail: elpmk5@yahoo.com.mx

Capituló

1

Comunicaciones Seriales

El puerto serial de las computadoras es conocido como puerto RS-232, la ventaja de este puerto es que todas las computadoras traen al menos un puerto serial, este permite la comunicaciones entre otros dispositivos tales como otra computadora, el mouse, impresora y para nuestro caso con los microcontroladores.

Existen dos formas de intercambiar información binaria: la paralela y la serial.

La comunicación paralela transmite todos los bits de un dato de manera simultánea, por lo tanto la velocidad de transferencia es rápida, sin embargo tiene la desventaja de utilizar una gran cantidad de líneas, por lo tanto se vuelve mas costoso y tiene las desventaja de atenuarse a grandes distancias, por la capacitancia entre conductores así como sus parámetros distribuidos.

Tipos de Comunicaciones Seriales:

La Existen dos tipos de comunicaciones seriales: la síncrona y asíncrona

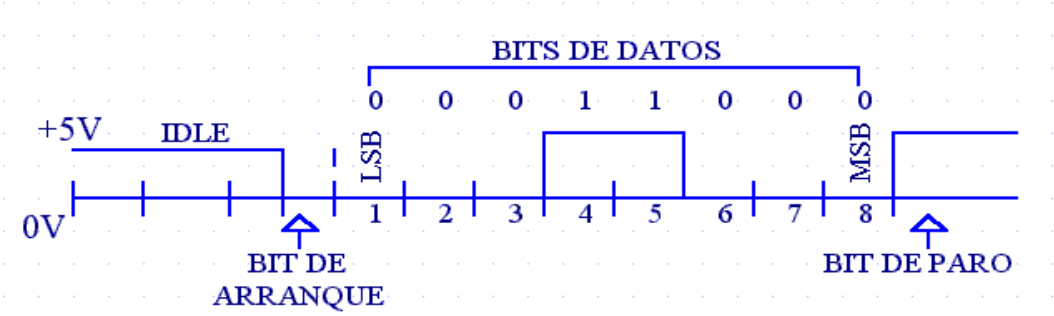
En la comunicación serial sincronía además de una línea sobre la cual se transmitirán los datos se necesita de una línea la cual contendrá los pulsos de reloj que indicaran cuando un datos es valido.

Ejemplos de este tipo de comunicación son:

- **I^2C**
- **ONE WIRE**
- **SPI**

En la comunicación serial asíncrona, no son necesarios los pulsos de reloj.
La duración de cada bit esta determinada por la velocidad con la cual se realiza la transferencia de datos.

La siguiente figura muestra la estructura de una carácter que se trasmite en forma serial asíncrona.



Normalmente cuando no se realiza ninguna transferencia de datos, la línea del transmisor se encuentra en esto de (idle) este quiere decir en estado alto.

Para iniciar la transmisión de datos , el transmisor coloca esta línea en bajo durante determinado tiempo, lo cual se le conoce como bit de arranque (start bit) y a continuación empieza a transmitir con un intervalo de tiempo los bits correspondientes al dato, empezando siempre por el BIT menos significativo (LSB), y terminando con el BIT mas significativo.

Si el receptor no esta sincronizado con el transmisor, este desconoce cuando se van a recibir los datos.

Por lo tanto el transmisor y el receptor deberán tener los mismos parámetros de velocidad, paridad, numero de bits del dato transmitido y de BIT de parada.

En los circuitos digitales, cuyas distancias son relativamente cortas, se pueden manejar transmisiones en niveles lógicos TTL (0-5V), pero cuando las distancias aumentan, estas señales tienden a distorsionarse debido al efecto capacitivo de los conductores y su resistencia eléctrica. El efecto se incrementa a medida que se incrementa la velocidad de la transmisión.

Todo esto origina que los datos recibidos nos sean igual a los datos transmitidos, por lo que nos e puede permitir la transferencia de datos.

Una de las soluciones mas lógica es aumentar los márgenes de voltaje con que se transmiten los datos, de tal manera que las perturbaciones a causa de la línea se pueden corregir.

La Norma RS-232

Ante la gran variedad de equipos, sistemas y protocolos que existen surgió la necesidad de un acuerdo que permitiera a los equipos de varios fabricantes comunicarse entre si. La **EIA (Electronics Industry Association)** elaboro la norma RS-232, la cual define la interfase mecánica, los pines, las señales y los protocolos que debe cumplir la comunicación serial

Todas las normas RS-232 cumplen con los siguientes niveles de voltaje:

- **Un “1” lógico es un voltaje comprendido entre –5v y –15v en el transmisor y entre -3v y –25v en el receptor.**
- **Un “0” lógico es un voltaje comprendido entre +5v y +15 v en el trasmisor y entre +3v y +25 v en el receptor.**

El envío de niveles lógicos (bits) a través de cables o líneas de transmisión necesita la conversión a voltajes apropiados. En los microcontroladores para representar un 0 lógico se trabaja con voltajes inferiores a 0.8v, y para un 1 lógico con voltajes mayores a 2.0V. En general cuando se trabaja con familias TTL y CMOS se asume que un "0" lógico es igual a cero Volts y un "1" lógico es igual a cinco Volts.

La importancia de conocer esta norma, radica en los niveles de voltaje que maneja el puerto serial del ordenador, ya que son diferentes a los que utilizan los microcontroladores y los demás circuitos integrados. Por lo tanto se necesita de una interfase que haga posible la conversión del niveles de voltaje a los estándares manejados por los CI TTL. Para mayor información en lo referente a la norma **TIA/EIA-232,** favor de ver la Bibliografía.

El Circuito MAX-232

Este circuito soluciona los problemas de niveles de voltaje cuando se requiere enviar unas señales digitales sobre una línea RS-232. Este chip se utiliza en aquellas aplicaciones donde no se dispone de fuentes dobles de +12 y –12 Volts. El MAX 232 necesita solamente una fuente de +5V para su operación, internamente tiene un elevador de voltaje que convierte el voltaje de +5V al de doble polaridad de +12V y –12V. Cabe mencionar que existen una gran variedad de CI que cumplen con la norma RS-232 como lo son: MAX220, DS14C232, MAX233, LT1180A.

Acceso al Puerto Serial a través de Vbasic

Para poder acceder al puerto serial y así poder enviar datos utilizado una aplicación creada en Visual Basic, se hace uso del control **MS COMM**, el cual trae incorporadas todas las funciones para configurar el puerto. Es gracias a este control que el manejo del puerto serial se facilita enormemente. Las propiedades más importantes de este control 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.

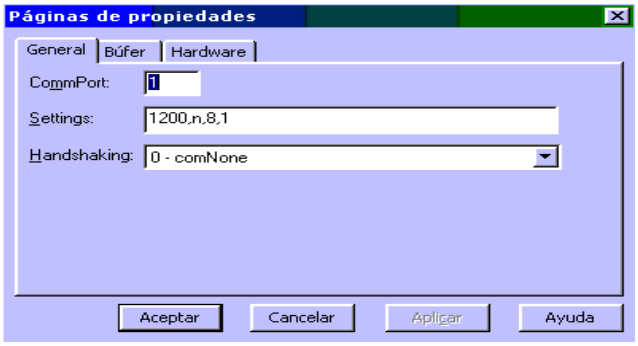
Para poder tener acceso a cualquier propiedad del puerto serial se utiliza la siguiente sintaxis:

Nombre del Control . Propiedad = Valor

En este caso el objeto es MS Comm1, por lo tanto si quisiera abrir el puerto, la instrucción sería:

MS Comm1.PortOpen = True

Sin embargo, para poder utilizar el puerto serial, primeramente, se debe colocar el control MS Comm1 en la forma y hacer clic con el botón derecho del mouse, para que puedan aparecer sus propiedades, tal y como lo muestra la sig. figura:



Configuración de los Parámetros del Puerto Serial en V Basic

Como la computadora sobre la cual se está trabajando solamente tiene un puerto serial y este es el comm1, en la propiedad **CommPort** debe tener el número 1, como los Microcontroladores envían y reciben la información a una velocidad de 1200 bps, 8 bits, sin paridad y 1 BIT de paro, en la propiedad **Settings** se debe configurar con la siguiente cadena: **1200, n ,8, 1**, y como no se va a realizar ningún control sobre el flujo de la información la propiedad **handshaking** debe ser igual a 0.

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

Para poder enviar una cadena de caracteres a través del puerto serial, lo único que se tiene que hacer es utilizar la propiedad output del objeto MS Comm:

Ejemplo:

MSComm1.Output = "Esto es una prueba"

Como se observa, una vez configurado el puerto serial, con esta instrucción se envía a través del puerto la cadena de caracteres, **"Esto es una prueba".**

Pasos para poder enviar datos a través del Puerto Serial:

- Insertar el control **MS Comm** sobre la forma:
- Establecer las siguientes propiedades :
  ComPort:
  Settings:
  Handshaking:

- Abrir el puerto, si este ya está abierto por otra aplicación, entonces se debe cerrar esa aplicación, para después volverlo a abrir el puerto con una aplicación en Visual Basic, esto se hace utilizando la siguiente instrucción:

```
MSComm1.Portopen = true
```

- Definir el tamaño del buffer receptor, esto se hace con la propiedad **InputLen**

```
MSComm1.InputLen = 1024
```

- Enviar los datos que se desean
- Cuando la aplicación se termine se debe cerrar el puerto.

Aplicación #1

1. Cree un Nuevo proyecto Form1 (es creado por default).
2. Seleccione del menú Project ,verifique que el control **Microsoft Comm**, este en la barra de herramientas
3. Agregue el control **MSCOMM** a la forma.
4. Agregue 2 controles **Command Buttons** a la forma.

Agregue el siguiente código a sus respetivos controles:

```
Option Explicit
Const Xon = &H11
Const Xoff = &H13

Private Sub Form_Load()
        Form1.Caption = "Primera aplicación con el Puerto Serial"
        With MSComm1
                .Handshaking = 2 - comRTS
                .RThreshold = 1
                .RTSEnable = True
                .Settings = "9600,n,8,1"
                .SThreshold = 1
                .PortOpen = True
        End With

        Command1.Caption = "&Send Xoff"
        Command2.Caption = "Send &Xon"
 End Sub


Private Sub Command1_Click()
    MSComm1.Output = "123456789" & Chr$(Xoff)
 End Sub


Private Sub Command2_Click()
        MSComm1.Output = "987654321" & Chr$(Xon)
End Sub


Private Sub Form_Unload(Cancel As Integer)
        MSComm1.PortOpen = False
End Sub
```

Aplicación #2

5. Cree un Nuevo proyecto Form1 (es creado por default).
6. Seleccione del menú Project , verifique que el control **Microsoft Comm**, este en la barra de herramientas
7. Agregue el control **MSCOMM** a la forma.
8. Agregue los siguientes controles, modificando las siguientes propiedades

Textbox: cambiar la propiedad **MultiLine=True**
Label: cambiar la propiedad **Caption= "Puerto Serial"**

Agregue el siguiente código a sus respetivos controles:

```
Const Xon = &H11
Const Xoff = &H13

Private Sub Form_Load()
    Form1.Caption = "Aplicación 2 con el Puerto Serial"
    With MSComm1
        .CommPort = 1
        .Handshaking = 2 - comRTS
        .RThreshold = 1
        .RTSEnable = True
        .Settings = "9600,n,8,1"
        .SThreshold = 1
        .PortOpen = True
    End With

    Text1.Text = ""
    Label1.Caption = "No input yet"
End Sub

Private Sub Form_Unload(Cancel As Integer)
    MSComm1.PortOpen = False
End Sub
```

```
Private Sub MSComm1_OnComm()
        Dim InBuff As String

        Select Case MSComm1.CommEvent
                                        ' Eventos Posibles
                Case comEvReceive
                        Label1.Caption = "Llegada de Datos"
                        InBuff = MSComm1.Input
                        Call ParseChars(InBuff)
                Case comEvSend
                 Case comEvEOF
        End Select
End Sub


Sub HandleInput(InBuff As String)
        Text1.Text = Text1.Text & InBuff
End Sub



Sub ParseChars(ByVal InString As String)
        Dim temp As String
        Dim x As Long
        Dim OutString as String

        For x = 1 To Len(InString)
                temp = Mid$(InString, x, 1)
                        If temp = Chr$(Xoff) Then
                                Label1.ForeColor = vbRed
                                Label1.Caption = "Xoff recivido"
                                  temp = ""
                        ElseIf temp = Chr$(Xon) Then
                                Label1.ForeColor = vbGreen
                                Label1.Caption = "Xon recivido"
                                  temp = ""
                        End If
                OutString = OutString & temp
                temp = ""
        Next x
                Call HandleInput(OutString)
End Sub
```

Bibliografía

Dicho tutorial fue una recopilación de las siguientes libros, paginas de Internet, etc.

http://www.senet.com.au/~cpeacock
http://www.lvr.com