Universidad del Táchira
Departamento de Ingeniería Electrónica
Instrumentación Electrónica

Utilización de los puertos serial y paralelo de una PC usando LabView®

Hecho Por:
Ing. Rafael Chacón
Ing. José Andrickson
Br. Juan Parada

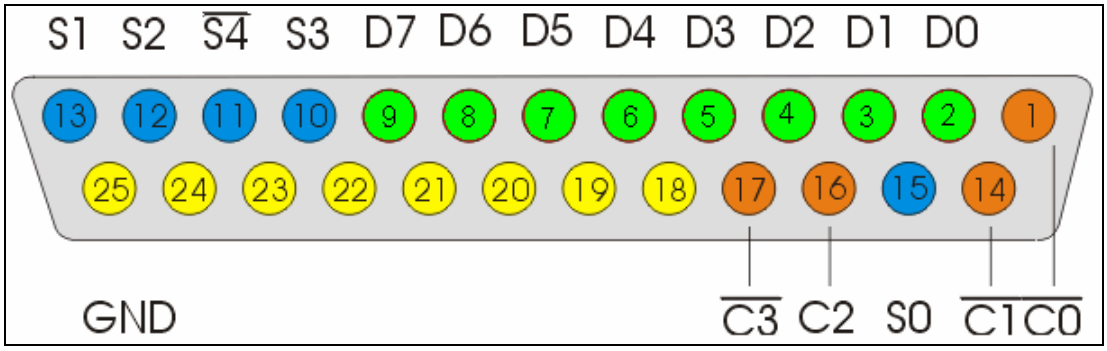
San Cristóbal, 2003

Puerto Paralelo

El puerto paralelo de una PC ha sido generalmente utilizado para el manejo de dispositivos de adquisición e impresión de imágenes, es por esto que desde un inicio se le dio el nombre de puerto de impresión o LPT1. Sin embargo, este puerto puede ser gestionado como una interfase para la adquisición de datos de una manera simple.

El puerto paralelo tiene varios modos de trabajo, para las aplicaciones comunes se utilizara la configuración sencilla o SPP. Este modo es el que se describirá a continuación.

En un esquema del puerto paralelo podemos describir las funciones básicas de sus pines, dividiéndolo en 4 zonas.



La primera de las zonas es la que corresponde a los datos (normalmente de salida desde la computadora hacia los dispositivos), ubicada entre los pines 2 y 9. Esta zona se puede acceder a través de la "dirección" (en formato hexadecimal) **378**, donde el pin 2 es el bit0 (LSB) de este byte y el pin 9 es el bit 7 (MSB).

La segunda es la zona de entrada de datos (hacia la computadora). Ocupa los pines 10, 11, 12, 13 y 15. Esta zona se puede acceder a través de la dirección (en formato hexadecimal) **379**. Aquí el pin 15 es el bit 3 de este byte, el pin 13 es el bit 4, el pin 12 es el bit 5, el pin 10 es el bit 6 y el pin 11 es el bit 7 (MSB), pero este último se encuentra negado.

La tercera es la zona de control, la cual incluye los pines 1, 14, 16 y 17. La computadora se vale de esta zona para gerenciar las diferentes señales de control sobre los dispositivos periféricos conectados al puerto paralelo. Esta zona no es usada para las aplicaciones comunes, pero se puede hacer referencia a ella a través de la dirección **37A**.

Ahora bien, una aplicación interesante es la utilización del bit 5 de la zona de control. Si se coloca en alto este bit (por software, escribiendo en la zona de control),

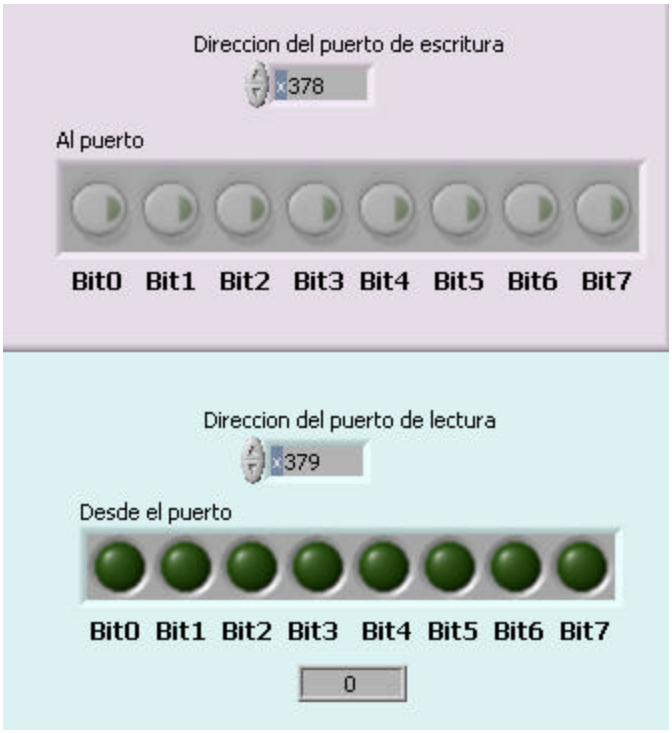
se habilita la zona de datos para ser utilizada como entrada (hacia la computadora), en vez de su uso común de salida (hacia los periféricos).

Los pines del 18 al 25 normalmente constituyen la cuarta zona. Esta zona solo se usa como "tierra". **No se debe conectar nada a esta zona que no vaya a estar a nivel de referencia 0V, pues se podría dañar el puerto**.

La lógica usada en el puerto paralelo es la TTL, es decir 0V es un "Cero" y 5V es un "Uno". Aun así, **el SPP no fue diseñado para manejar más de 20 mA**. Es por esto que se debe tener cuidado con los niveles de corriente y voltaje que se conecten al puerto.

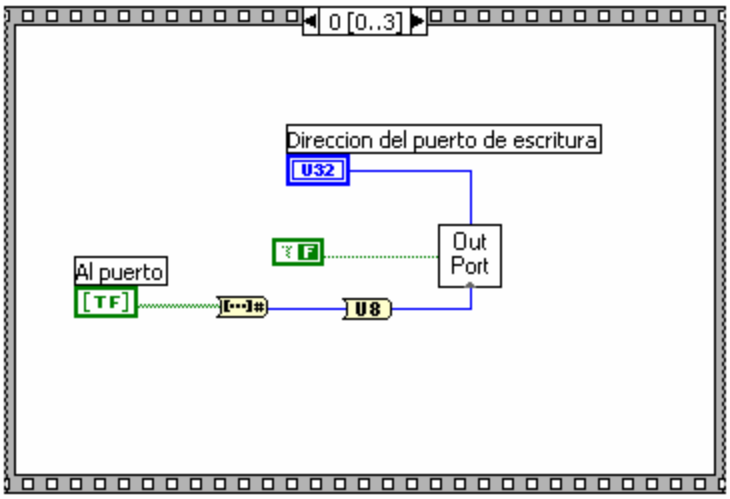
Para la utilización del puerto paralelo usando LabVIEW se hará un programa muy simple con fines educativos.

Primero construimos un panel frontal como el siguiente:



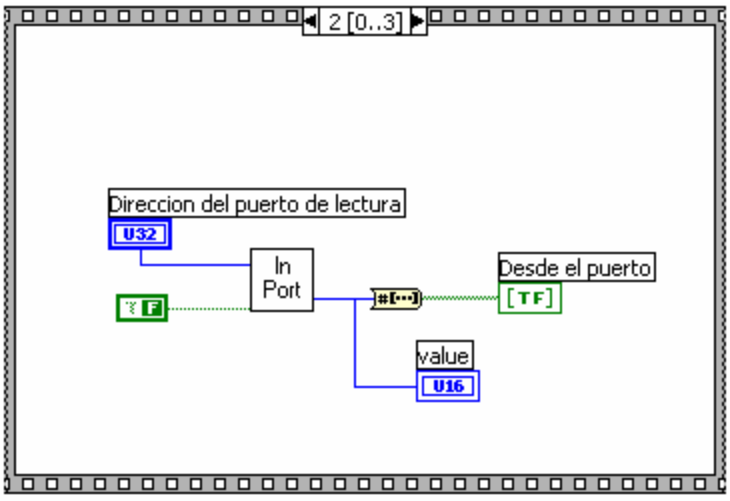
Aquí se incluyen dos vectores de tipo **Boolean** de 8 datos. Cada uno a manera de un byte u ocho bits. El superior se usara para enviar datos a una zona determinada del puerto y mientras que el inferior servirá para mostrar los datos que llegan a otra zona.

En una simple secuencia colocamos como primer frame el siguiente:



Donde el dato a escribir en el puerto (el vector Boolean superior en el panel frontal) se transforma en un número decimal y se "formatea" al tipo de dato U8. Este dato se introduce en la función **OutPort**, junto con la dirección correspondiente a la zona en la que se desea "escribir" el dato. La bandera boolean de la función **OutPort** se coloca a TRUE si el dato a escribir es una palabra (16 bits) o en FALSE si se va a escribir un byte (8bits).

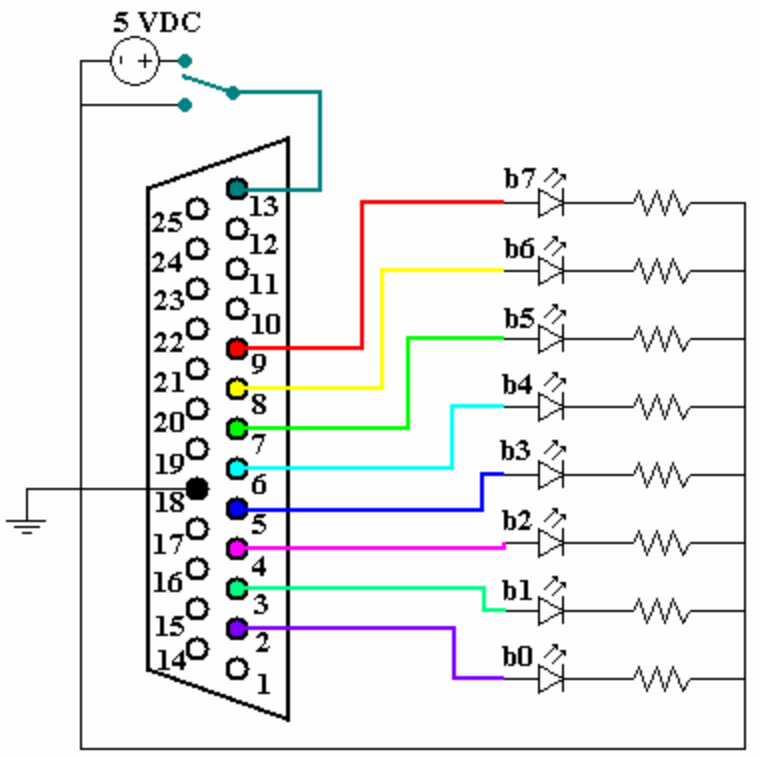
En el siguiente paso de la secuencia se coloca un retardo de unos 100 milisegundos. Y Luego en el tercer frame se coloca lo que sigue:



Donde a la función **InPort** se le introduce la dirección de la zona del puerto de la que se quiere "leer" un dato, y una bandera boolean que se coloca en FALSE si se desea leer un byte y en TRUE si se desea leer una palabra. La función InPort devuelve un valor numérico en formato U16, este se convierte en un vector de boolean y se muestra en el panel frontal.

Y en el último frame se coloca otro retardo de unos 100 milisegundos.

Una implementación circuital, muy común, utilizada para probar el puerto paralelo es la siguiente:



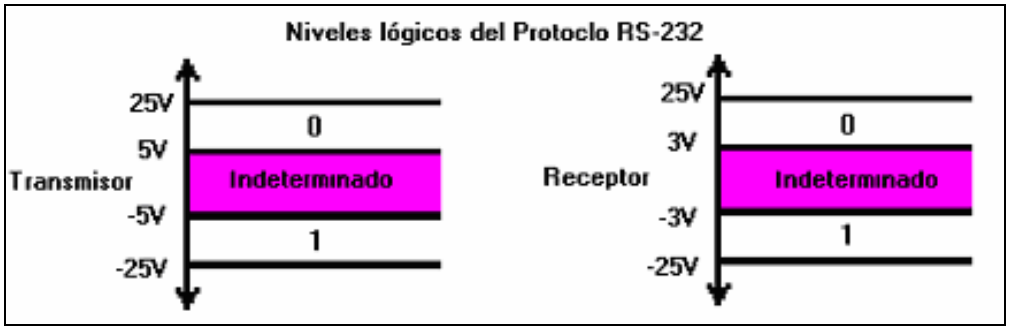
Donde un arreglo de leds nos permite verificar el funcionamiento de la escritura, en la zona de datos 378. Junto a cada led se coloca una resistencia limitadora de corriente, a fin que esta no supere los 20 mA. Un valor recomendado para esta resistencia es 1KO (5V/1K O = 5mA).

Se utiliza también un switch, para intercambiar entre 0V y 5V, el estado de uno de los pines de la zona de entrada, en este caso el pin 13 (bit 4 de la dirección 379).

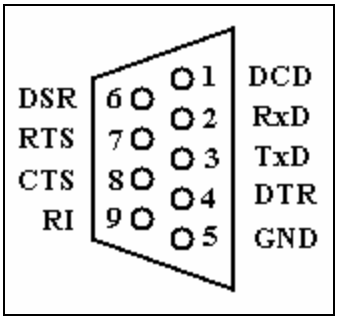
Con un montaje de este tipo se corre continuamente el programa anteriormente desarrollado y se hacen las pruebas con el puerto paralelo.

Puerto Serial

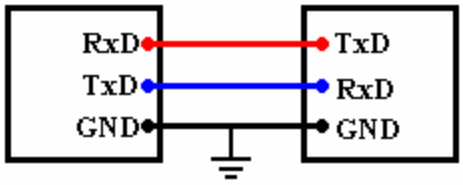
La transmisión de un dato a través del puerto serial de la PC se hace a través de un protocolo de comunicaciones denominado RS-232. En él los niveles lógicos se definen:



Para la conexión de dispositivos vía puerto serial, se creó un conector de 25 pines, que luego fue simplificado por una versión de 9 pines, denominado DB-9. Este último se describe en el siguiente gráfico:



Donde los pines que hacen efectiva la comunicación son el RxD (Recepción de datos), el TxD (Transmisión de datos) y el SG (Signal Ground) o GND (o nivel de referencia cero o tierra). Así se tiene que el pin de transmisión de un dispositivo se conecta con el de recepción de otro equipo y viceversa, estando ambos referenciados a un mismo “nivel cero”.



VISA

La alianza de sistemas VXIplu&play se creó con la meta de incrementar la interoperatibilidad entre los distribuidores y los usuarios finales de los sistemas que trabajan con VXIbus. Logrando hacer más fácil el uso de estos dispositivos a través de un nuevo estándar tanto para hardware como para software.

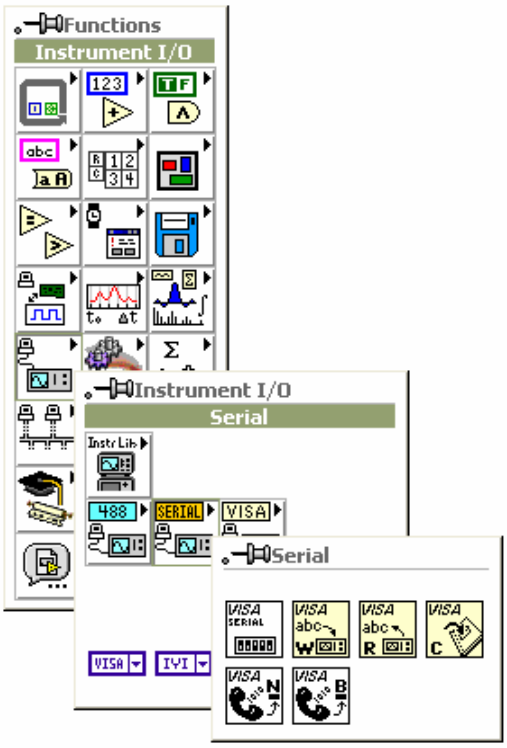
En el corazón de esos estándares se encuentra la Arquitectura de Software de Instrumento Virtual o VISA (por sus siglas en inglés). Siendo este estándar en el que están basados todos los componentes de software VXIplug&play. En el pasado existían muchos softwares I/O diferentes para controlar los dispositivos que usan los protocolos GPIB y VXI. Ahora más de 35 compañías (incluyendo Tektronix, Agilent y National Instruments) se unieron para hacer que el software fuera intercambiable, reutilizable y que soportara el paso del tiempo.

Un Recurso de Instrumento de Control VISA (o INSTR) permite al controlador interactuar con el dispositivo asociado a dicho recurso. LabVIEW instala un software denominado: *VISA Interactive Control*, el cual permite visualizar y obtener información de todos los dispositivos GPIB y VXI que estén conectados actualmente a la PC en la que se está trabajando. Usualmente, aparecerán al menos los puertos de comunicaciones (serial y paralelo) y sus distribuciones (cuantos COM seriales y cuantos LPT tiene la computadora).

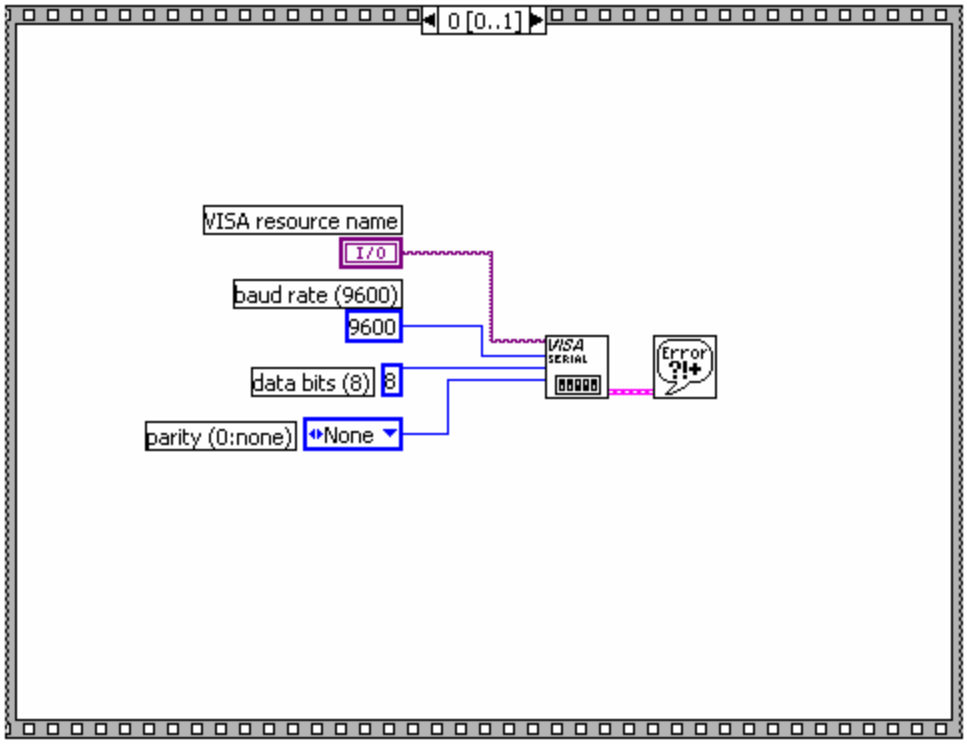
De lo antes mencionado, se concluye que LabVIEW (® National Instruments) manejara los puertos de comunicación como recursos tipo INSTR VISA. Además, al utilizar el *VISA Interactive Control* se observa que LabVIEW clasifica los puertos de comunicación como dispositivos ASRL :: INSTR. Donde ASRL1 está relacionado con el COM1, ASRL2 con el COM2 y ASRL10 con el LPT1.

A continuación se desarrollara un programa ejemplo de un transmisor serial, más adelante se desarrollara el programa ejemplo para el receptor.

Las funciones para el manejo del puerto serial se encuentran dentro de la categoría Instrument I/O, tal y como se muestra a continuación:



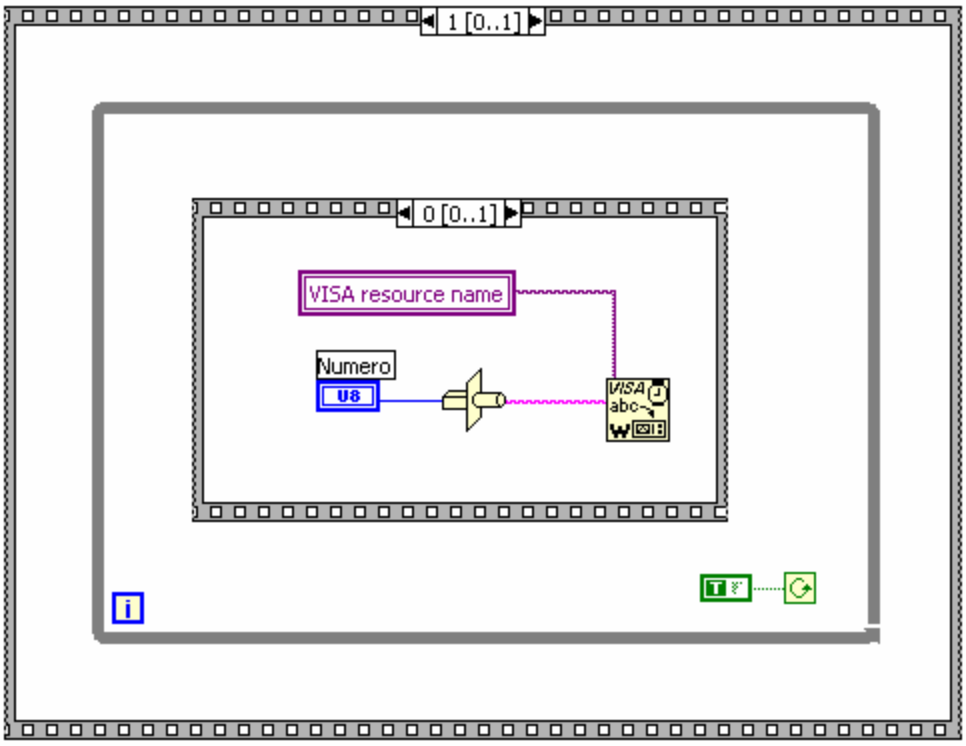
Lo primero que hacemos es implementar en nuestro diagrama, una estructura tipo secuencia, donde en el primer frame de la misma se inicializaran los parámetros del puerto serial utilizando la función **VISA Configure serial Port** 



En el parámetro de entrada **VISA resource name** creamos un control (que será mostrado en el panel frontal de nuestro VI). Es desde este control que elegiremos el recurso a utilizar para la transmisión serial, por ejemplo: el COM1 de la computadora.

Otro parámetro a establecer es la tasa de transferencia o baud rate. También se debe decidir cual es la cantidad de bits que serán consideradas un byte y si se hará chequeo de paridad (y de que tipo: par o impar).

Una vez configurado el puerto, hacemos el siguiente paso de la secuencia principal.

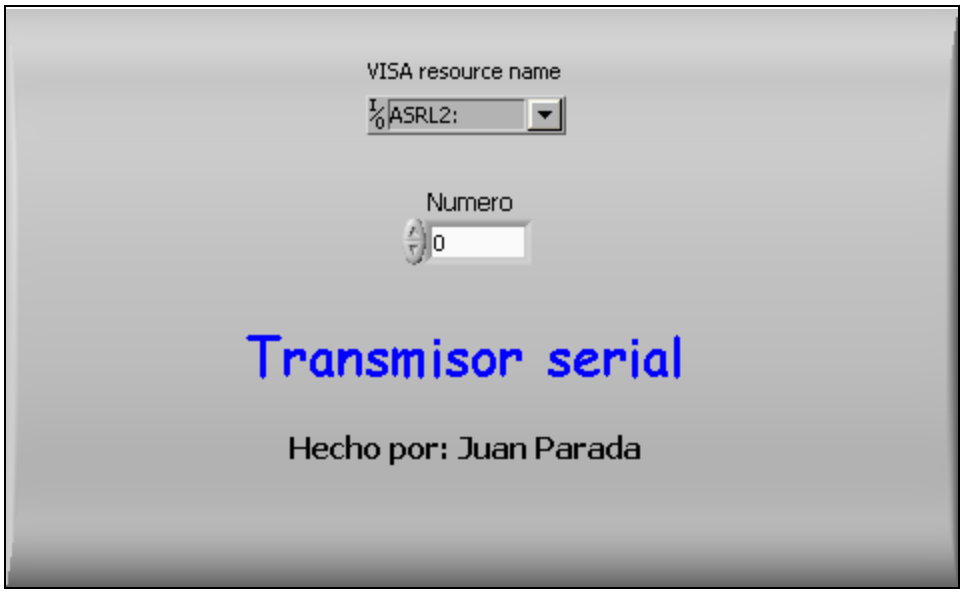


Donde colocamos un ciclo WHILE que se ejecuta de manera infinita y que contiene una secuencia de dos pasos.

En el primer frame, de la secuencia menor, se tiene un número (que es un control colocado en el panel frontal) que se transforma en un dato tipo string utilizando la función **Type Cast**  . Este dato se introduce en la función **VISA Write**  para ser enviado al dispositivo reseñado e inicializado anteriormente por **VISA resource name.**

Finalmente en el segundo frame de la secuencia menor se tiene un pequeño retardo del orden de los 100 milisegundos.

Así pues, obtenemos un panel frontal muy simple, del siguiente tipo:



Ahora elaboraremos el receptor serial, basados en el programa del transmisor.

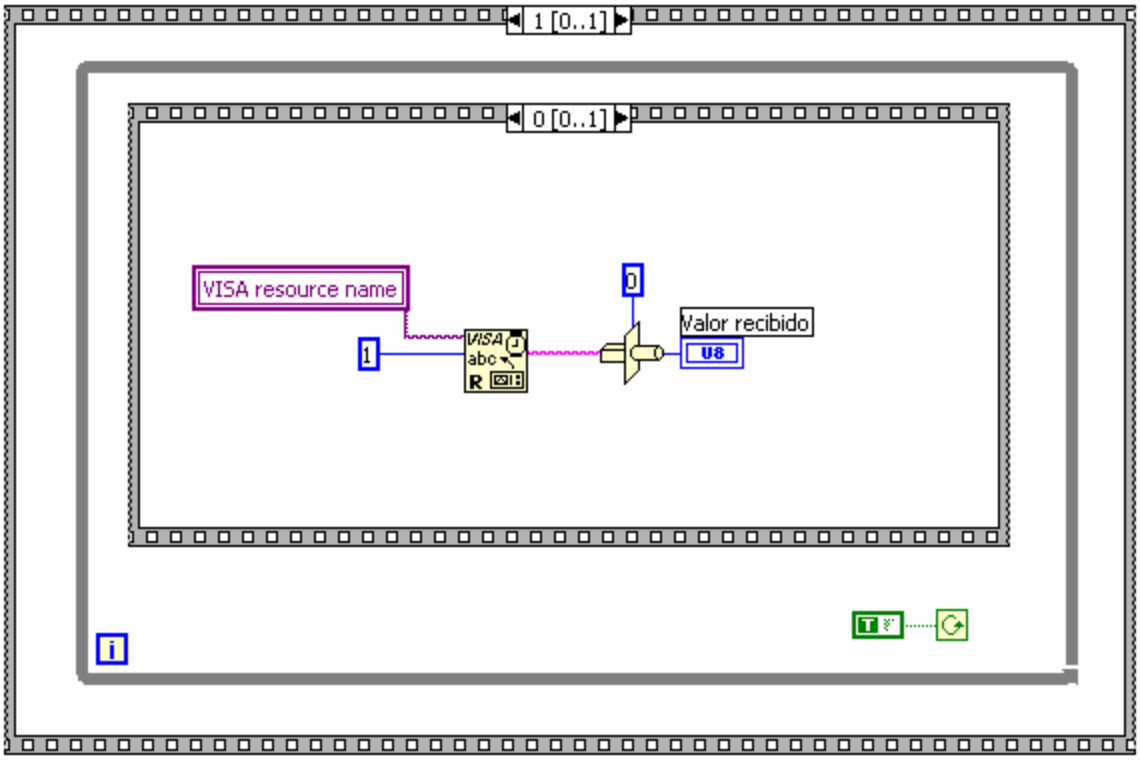
Aquí también se tendrá como estructura principal una secuencia, cuyo primer frame será idéntico al primer frame del transmisor, es decir se hará la configuración del puerto con los mismos valores.

Una vez configurado el puerto, hacemos el siguiente paso de la secuencia principal. En él volvemos a encontrar un ciclo WHILE infinito que contiene a una secuencia de dos frames. La diferencia es que ahora vamos a recibir datos en vez de enviarlos; Por esto procedemos utilizar la función **VISA Read**  .

Un punto importante de la función **VISA Read** es que ella necesita saber cuantos bytes se van a leer. A pesar de que no la vamos a utilizar en este programa de ejemplo, la propiedad **VISA Bytes at Serial Port** 

devuelve el número de bytes que se encuentran disponibles para ser leídos en el buffer del puerto serial. Si se utiliza esta propiedad, se puede hacer una programa que lea todos los bytes enviados al puerto, sin importar cuantos sean estos.

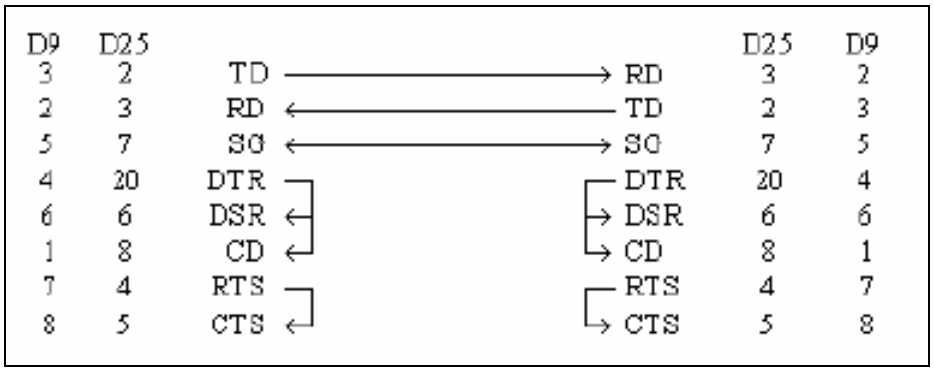
Para nuestro ejemplo, le indicamos a la función **VISA Read** que lea un solo byte desde el dispositivo reseñado e inicializado anteriormente por **VISA resource name.**



Como ya debe haber notado, los datos para comunicación serial son tratados por LabVIEW como datos tipo string. Por esto el dato recibido lo transformamos en un número utilizando la función **Type Cast**, colocándole como ejemplo un dato numérico entero para obtener un dato tipo U8, es decir un byte. Este valor, al igual que el control **VISA resource name**, son mostrados en el panel frontal de nuestro VI.

Para hacer las pruebas a los VIs desarrollados anteriormente se debe disponer una circuiteria que reciba (desde el VI transmisor) las señales enviados y/o tener un circuito que envié (hacia el VI receptor) señales en modos serial.

Si se quieren hacer las pruebas utilizando 2 computadoras se debe construir un cable “Null Modem”, que en sus versiones de conectores DB-25 y DB-9, se conectarían así:



Es muy importante recordar que: **La configuración del dispositivo transmisor y del dispositivo receptor debe ser exactamente la misma (mismo baud rate, número de bits, chequeo de paridad, etc) para que se pueda establecer la conexión**.