**Recibir datos del PC**

Ahora vamos a invertir el proceso, vamos a enviar datos del PC al Arduino. El programa básico para recibir datos es el siguiente:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | void setup()  {    Serial.begin(9600);  }    void loop()  {    if(Serial.available()>0) //Comprobamos si en el buffer hay datos    {      int dato=Serial.read();  //Lee cada carácter uno por uno y se almacena en una variable        Serial.println(dato);  //Imprimimos en la consola el carácter recibido    }  } |

Y aquí se complica un poco la cosa.  ¿Por qué? Pues porqué como hemos explicado antes, el buffer del puerto serie recibe los datos byte por byte y en formato decimal, quiere decir, los caracteres recibidos serán números correspondientes al código ASCII (El Arduino ve Símbolos de del tabla ASCII y los traduce a Decimal).

Por ejemplo: Enviamos desde el PC la palabra “ARDUINO”. El buffer lee uno por uno los caracteres recibidos, primero la “A”, después la “R”,…, hasta la “O”, y los captura como carácter decimal correspondiente a la tabla del código ASCII. Quiere decir que el Arduino ve los siguientes valores:

                                         65  82  68  85  73  78  79

Que corresponden a       A    R    D    U    I     N    0

Para solucionar esto podemos usar la función *write()* en vez de *println()*. Porque *write()* envía uno o varios bytes en forma de datos binarios. De esta manera, cuando enviamos desde el PC la palabra “ARDUINO”, en el Monitor Serial imprimirá “ARDUINO”.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | void setup()  {    Serial.begin(9600);  }    void loop()  {    if(Serial.available()>0) //Comprobamos si en el buffer hay datos    {      int dato=Serial.read();  //Lee cada carácter uno por uno y se almacena en una variable        Serial.write(dato);  //Imprimimos en la consola el carácter recibido    }  } |

Lo que hace la función write() con los valores recibidos (65  82  68  85  73  78  79) es enviarlos al Monitor Serial con su carácter correspondiente de la tabla del código ASCII. Así visualizamos los mismos caracteres que hemos escrito inicialmente en el PC.

Otra forma similar de hacerlo sin cambiar el println() es en vez de usar el tipo de variable **int**para leer los datos del buffer, usaremos el tipo de variable ***char*** que es un tipo de dato que ocupa un byte de memoria y almacena un valor de carácter (almacena el símbolo de la tabla ASCII).

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | void setup()  {    Serial.begin(9600);  }    void loop()  {    if(Serial.available()>0) //Comprobamos si en el buffer hay datos    {      char dato=Serial.read();  //Lee cada carácter uno por uno y se almacena en una variable        Serial.println(dato);  //Imprimimos en la consola el carácter recibido    }  } |

Pero y si queremos usar datos recibidos dentro del programa. Por ejemplo: si queremos enviar el valor 51 desde el PC al Arduino, y este queremos que a este valor 51 le sumemos un valor 2 para que nos salga un valor 53. ¿Cómo lo podemos hacer? Ya que cuando el Arduino reciba el valor 51, lo va a interpretar como un 53(5) y 49(1).

Usaremos un código muy sencillo. Creamos un array tipo ***char*** para que almacene los datos en forma de cadena de caracteres (String). Lo que haremos es convertir esta cadena de caracteres a un número entero usando la función ***atoi()***, que no es una función de Arduino sino de C++, pero que está incluido automáticamente en todos los sketches.  También usaremos la función ***memset()*** (también procedente de C++) para limpiar el array después de cada lectura.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | void setup()  {    Serial.begin(9600);  }    char cadena[30]; //Creamos un array que almacenará los caracteres que escribiremos en la consola del PC. Le asignamos  un tope de caracteres, en este caso 30  byte posicion=0;  //Variable para cambiar la posición de los caracteres del array  int valor;  //Variable del valor entero    void loop()  {    if(Serial.available()) //Nos dice si hay datos dentro del buffer    {      memset(cadena, 0,sizeof(cadena));//memset borra el contenido del array  "cadena" desde la posición 0 hasta el final sizeof        while(Serial.available()>0) //Mientras haya datos en el buffer ejecuta la función      {        delay(5); //Poner un pequeño delay para mejorar la recepción de datos        cadena[posicion]=Serial.read();//Lee un carácter del string "cadena" de la "posicion", luego lee el siguiente carácter con "posicion++"        posicion++;      }        valor=atoi(cadena);//Convertimos la cadena de caracteres en enteros      Serial.println(valor+2);//Imprimimos el valor sumandole un valor +2      posicion=0;//Ponemos la posicion a 0    }    } |