# Lenguajes de programación

Son muchos los lenguajes en los que se pude programar la controladora CNICE pero en este caso nos centraremos en dos de ellos:

M*SWLogo*

C

## Lenguaje MSWLogo

MSWLogo se puede usar en cualquier Sistema Operativo de entorno Windows pero no MSDOS por tratarse de un lenguaje visual.

Mención especial a los sistemas Windows 2000 y Windows XP, en los cuales se permite el acceso al puerto paralelo mediante las funciones **PortOut** y **PortIn** de la **librería io.dll**. Para ello, hay que cargar previamente la librería io.dll , mediante la orden de MSWLogo:

*cargadll "io.dll*

**Es importante que sepa que se ha desarrollado una versión de MSWLogo 6.5a traducida al Castellano por el ITE (descarga en la sección de Descargas de este CD-Rom). Este compilador de MSWLogo incluye todas las rutinas primitivas necesarias para manejar la controladora CNICE, para leer y escribir en las entradas y salidas digitales y analógicas y puesto que este compilador ya realiza su carga de la librería io.dll no será necesario realizar su carga. Si intenta realizar la carga se le mostrará una pantalla de error. Para poder saber el valor que tienen en cada momento las entradas y salidas el usuario solo necesitará realizar las llamadas a los procedimientos y funciones que se proporcionan con el compilador, liberando de esta forma al usuario de la tediosa tarea de definir funciones y procedimientos para leer/escribir por el puerto.**

No obstante y si no desea utilizar este compilador, a continuación tiene los procedimientos básicos para activar/desactivar las salidas digitales y para leer las entradas digitales y analógicas aunque, recordamos que estos procedimientos están ya implementados como funciones del propio MSWLogo 6.5a desarrollado por el ITE.

A continuación se explican los procedimientos básicos necesarios implementados en este lenguaje de programación para realizar las siguientes acciones:

*Escribir en las salidas digitales*

*Leer las entradas digitales*

*Leer las entradas analógicas*

Destacar que los nombres que se han empleado para las funciones se pueden cambiar:

*SALIDA_DIGITAL, VE? y Leer_Entrada_Analogica_num*.

Lo importante es el código que contiene cada función.

## *Función para escribir en las salidas digitales*

```
PARA SALIDA_DIGITAL :VALOR

LLAMADLL [v PortOut w 7 w 890]

HAZ ''x (LISTA ''v ''PortOut ''w :Dato ''w 888)

LLAMADLL :x

FIN
```

Procedimiento que pone un número entre 0 y 255 que viene representado por el parámetro *VALOR* en las salidas digitales. *VALOR* es un número decimal que representa a un número binario, el cual indica qué salidas digitales se quieren activar.

Para poder realizar lo anterior se usa la función *PortOut* de la librería io.dll. Para poder invocar a las funciones de la librería io.dll se necesita la instrucción en MSWLogo *LLAMADLL*.

La llamada al procedimiento es:

```
SALIDA_DIGITAL Dato
```

Si por ejemplo queremos activar las salidas digitales 0 y 5 deberíamos ejecutar la instrucción siguiente:

```
SALIDA_DIGITAL 33
```

## *Función para leer las entradas digitales*

```
para VE?

HAZ ''bajo LEER_ENTRADA_DIGITAL_BAJA

HAZ ''alto LEER_ENTRADA_DIGITAL_ALTA

HAZ ''dato bito :alto * 16 :bajo

DEV :dato

Fin

para LEER_ENTRADA_DIGITAL_BAJA

LLAMADLL [v PortOut w 3 w 890]

HAZ ''BAJO PRIMERO LLAMADLL [w PortIn w 889]

HAZ ''MASCARA 120

HAZ ''BAJO bity :BAJO :MASCARA

HAZ ''BAJO :BAJO / 8

HAZ ''BAJO BITINVERSO :BAJO

HAZ ''MASCARA 15
```

*HAZ ''BAJO bity :BAJO :MASCARA*

*DEV :BAJO*

*Fin*

*para LEER_ENTRADA_DIGITAL_ALTA*

*LLAMADLL [v PortOut w 1 w 890]*

*HAZ ''ALTO PRIMERO LLAMADLL [w PortIn w 889]*

*HAZ ''MASCARA 120*

*HAZ ''ALTO bity :ALTO :MASCARA*

*HAZ ''ALTO :ALTO / 8*

*HAZ ''ALTO BITINVERSO :ALTO*

*HAZ ''MASCARA 15*

*HAZ ''ALTO bity :ALTO :MASCARA*

*DEV :ALTO*

*Fin*

El procedimiento *VE?* devuelve el estado de los ocho sensores digitales como un número decimal equivalente a un número binario que es la suma de las entradas que toman valor 1. Para ello se usa la función *PortIn* y *PortOut* de la librería io.dll.

El procedimiento *VE?* necesita como procedimientos auxiliares

*LEER_ENTRADA_DIGITAL_BAJA*

*LEER_ENTRADA_DIGITAL_ALTA*

los cuales leen las entradas digitales bajas (de la 0 a la 3) y las entradas digitales altas (de la 4 a la 7), respectivamente.

Si queremos almacenar en una variable llamada *entradas* el valor de las ocho entradas digitales, la llamada al procedimiento sería:

*HAZ "entradas VE?*

Si por ejemplo, el valor que almacenara la variable *entradas* fuese 141, quería decir que las entradas digitales 0, 2 , 3 y 8 estaría recibiendo un valor de 1 mientras que el resto de entradas tomaría valor 0.

## *Función para leer una entrada analógica determinada*

```
para Leer_Entrada_Analogica_num :SELECC

;Lectura del nibble bajo de la entrada analógica

LLAMADLL [v PortOut w 5 w 890]

HAZ ''x (LISTA ''v ''PortOut ''w :SELECC ''w 888)

LLAMADLL :x

LLAMADLL [v PortOut w 2 w 890]

HAZ ''BAJO PRIMERO LLAMADLL [w PortIn w 889]

ESPERA 1

HAZ ''MASCARA 120

HAZ ''BAJO ( bity :BAJO :MASCARA )

HAZ ''BAJO :BAJO / 8

HAZ ''MASCARA 15

HAZ ''BAJO ( bity :BAJO :MASCARA )

;Lectura del nibble alto de la entrada analógica

LLAMADLL [v PortOut w 0 w 890]

HAZ ''ALTO PRIMERO LLAMADLL [w PortIn w 889]

ESPERA 1

HAZ ''MASCARA 120

HAZ ''ALTO ( bity :ALTO :MASCARA )

HAZ ''ALTO :ALTO / 8

HAZ ''MASCARA 15

HAZ ''ALTO ( bity :ALTO :MASCARA )

HAZ ''ALTO :ALTO * 16

HAZ ''DATO_DIGITAL ( bito :ALTO :BAJO )

DEV :DATO_DIGITAL

Fin
```

Procedimiento que devuelve en la variable *DATO_DIGITAL* el valor leido correspondiente a la entrada analógica que se le indique en el parámetro *SELECC*. El valor que puede tomar *SELECC*

es 1, 2, 4 u 8 para hacer referencia a las entradas analógicas 1, 2, 3 ó 4 respectivamente. El proceso es similar al explicado en la función de Visual Basic para leer las entradas digitales.

Si deseamos saber el valor de tensión, sólo debemos introducir las siguientes líneas en el programa principal:

*HAZ "dato_tension (Leer_Entrada_Analogica_num ENT)*5 / 256*

donde *ENT* puede tomar los valores 1, 2, 4 u 8 dependiendo de la entrada analógica de la que queramos saber su valor de tensión.

La llamada al procedimiento es:

*Leer_Entrada_Analogica_num ENT*

donde *ENT* puede tomar los valores 1, 2, 4 u 8 dependiendo de la entrada analógica de la que queramos saber su valor de tensión.

### Lenguaje C

A continuación se explican los procedimientos básicos necesarios implementados en este lenguaje de programación para realizar las siguientes acciones:

*Escribir en las salidas digitales*

*Leer las entradas digitales*

*Leer las entradas analógicas*

Destacar que los nombres que se han empleado para las funciones se pueden cambiar:

*Escribir_Salidas_Digitales, Leer_Entrada_Digital y Leer_Entrada_Analógica.*

Lo importante es el código que contiene cada función.

## *Función para escribir en las salidas digitales*

```
void Escribir_Salidas_Digitales (int Dato)

{

PortOut (0x37A,0x7)

PortOut (0x378,Dato)

}
```

Procedimiento que pone un número entre 0 y 255 que viene representado por *Dato* en las salidas digitales. Para ello se usa la función *PortOut* de la librería io.dll .

Las direcciones hay que escribirlas en hexadecimal anteponiendo al número los caracteres **0x**.

La llamada al procedimiento es:

*Escribir_Salidas_Digitales (Dato)*

## *Función para leer las entradas digitales*

```
void Leer_Entrada_Digital (int *bajo,int *alto,int *dato)

{

PortOut (0x37A, 0x3);

*bajo = PortIn(0x379);

*bajo = (*bajo & 0x78) / 8;

*bajo = ~*bajo;

*bajo = *bajo & 15;

PortOut(0x37A, 0x1);

*alto = PortIn(0x379);

*alto = (*alto & 0x78) / 8;

*alto = ~*alto;

*alto = *alto & 15;

*dato=(*alto * 16) | *bajo;

}
```

Procedimiento que lee un dato correspondiente a las entradas digitales y lo almacena en el parámetro *dato.* El procedimiento es similar al explicado en Visual Basic para leer las entradas digitales.

La llamada al procedimiento es:

```
Leer_Entrada_Digital (&bajo, &alto, &dato)
```

## *Función para leer una entrada analógica determinada*

```
void Leer_entrada_analogica (int *selecc,int *bajo,int *alto,int *dato)

{

PortOut(0x37A,5);

PortOut(0x378,selecc);

PortOut(0x37A,8);

PortOut(0x37A,2);

sleep(1);

*bajo=PortIn(0x379);
```

```
*bajo=(*bajo & 0x78)/8;

*bajo=~*bajo;

*bajo=*bajo & 15;

PortOut(0x37A,5);

PortOut(0x378,selecc);

PortOut(0x37A,8);

PortOut(0x37A,0);

sleep(1);

*alto=(*alto & 0x78)/8;

*alto=~*alto;

*alto=*alto & 15;

*dato=(*alto * 16)|*bajo;

}
```

Procedimiento que lee un dato correspondiente a las entrada analógica que se le indique en el parámetro *selecc* y lo devuelve en la variable *dato*.

El parámetro *selecc* puede tomar los valores 1, 2, 4 u 8, correspondientes a las entradas analógicas 1, 2, 3 y 4 respectivamente.

Si deseamos saber el valor de tensión, sólo debemos introducir las siguientes líneas en el programa principal:

```
dato_tension=dato*5 / 256
```

La llamada al procedimiento es:

```
Leer_Entrada_Analógica (&selecc, &bajo, &alto, &dato)
```

**Es muy importante destacar que las llamadas a las funciones *PortOut* y *PortIn* sustituyen a las antiguas outp e inp que se localizaban en la librería conio.h. Estas funciones accedían directamente al hardware de la máquina. Por motivos de seguridad, esta capacidad fue restringida a partir del sistema operativo Windows 2000. Por ello, tanto en este S.O. como en Windows XP el acceso a los puertos se debe hacer mediante el uso de las funciones PortOut y PortIn de la librería io.dll.**