NetMedia SuperRobotica.com

Nota de Aplicación
Basic Express

Clases de bloques de datos (DATA BLOCK)

Introducción

Esta nota de aplicación describe el uso de las clases de enunciados de bloques de datos (DATA BLOCK) en un sistema BasicX.

Inicialización de las matrices

Un problema con las matrices convencionales es que no existe una manera sencilla de iniciarlas sin incurrir en una penalización de rendimiento. Los dialectos tradicionales de Basic utilizan enunciados de datos (DATA) para funciones similares, sin embargo los enunciados (DATA) tienen un modo de uso extraño y no son compatibles con VB.

BasicX intenta resolver este problema al proporcionar al sistema unas clases de datos de bloque (DATA BLOCK) definidos por el sistema, que son los equivalentes de las matrices iniciadas almacenadas en la memoria EEPROM:

En este ejemplo, utilizaremos matrices 1-D de bytes. El primer paso es declarar un objeto de datos de bloque, que debe estar en código de nivel de módulo:

```
Private Vector As New ByteVectorDataRW
```

El siguiente paso es definir la fuente de los datos:

```
Call Vector.Source("ByteVector.txt")
```

El método fuente (Source) especifica un fichero de texto ubicado en el PC. Aunque es legal especificar una ruta completa, en este ejemplo omitiremos la ruta de fuente, lo que implica que el fichero está ubicado en el mismo subdirectorio que el fichero de proyecto BXP. Tenga en cuenta que el método fuente (Source) debe realizarse antes que cualquier otro acceso al objeto de los datos de bloque:

Una vez definido el objeto, podemos leer y escribir en él como si se tratase de una matriz. En este caso, la matriz reside en la memoria EEPROM por ejemplo, que es también donde está memorizado el programa. Sintaxis:

```
Dim B As Byte

' Write to element 1.
Vector(1) = B

' Read element 1.
B = Vector(1)
```

Advertencia – un objeto de datos de bloque es similar a un objeto de datos variables en lo que respecta a las limitaciones del ciclo de escritura y al tiempo que se tarda en escribir en el objeto.

Ejemplo de programa

Este programa lee y escribe los primeros 5 elementos de un objeto de vectores de bytes:

```
Private Vector As New ByteVectorDataRW   ' Read-write.

Public Sub Main()

    Dim N As Byte
    Dim B As Byte

    Call Vector.Source("ByteVector.txt")

    Debug.Print
    Debug.Print "   Data using object access:"

    For N = 1 To 5
        Debug.Print "      Vector("; CStr(N); ") = ";
        Debug.Print Cstr(Vector(N))
    Next

    Debug.Print
    Debug.Print "   EEPROM address of Vector:  ";
    Debug.Print CStr(Vector.DataAddress)

    Debug.Print
    Debug.Print "   Data using direct EEPROM access:"

    For N = 0 To 4   ' Note shift to 0-base.
        Call GetEEPROM( Vector.DataAddress + CLng(N), B, 1 )
        Debug.Print "      Vector("; CStr(N+1); ") = "; CStr(B)
    Next

    ' Modify the second element upon first download.
    If ( FirstTime() ) Then
        Vector(2) = 255
    End If

End Sub
```

El fichero de texto ByteVector.txt contiene 5 líneas de texto:

```
10
20
30
40
50
```

Esta es la salida del programa después de la descarga:

```
Data using object access:
   Vector(1) = 10
   Vector(2) = 20
   Vector(3) = 30
   Vector(4) = 40
   Vector(5) = 50
EEPROM address of Vector:  1495
Data using direct EEPROM access:
   Vector(1) = 10
   Vector(2) = 20
   Vector(3) = 30
   Vector(4) = 40
   Vector(5) = 50
```

Tenga en cuenta que la función FirstTime hace que el programa se ejecute de manera diferente la primera vez después de la descarga. La segunda vez que ejecute el programa después de la descarga, esta sería la salida:

```
Data using object access:
   Vector(1) = 10
   Vector(2) = 255
   Vector(3) = 30
   Vector(4) = 40
   Vector(5) = 50
EEPROM address of Vector:  1495
Data using direct EEPROM access:
   Vector(1) = 10
   Vector(2) = 255
   Vector(3) = 30
   Vector(4) = 40
   Vector(5) = 50
```

Puede comprobar que el valor del elemento 5 ha cambiado de 20 a 255.

Debe considerar que también puede acceder a la dirección EEPROM del objeto. En este caso, el objeto está ubicado en la dirección 1495. Puede volver a comprobar la dirección consultando el mapa de ficheros MPP, que muestra una lista con la ubicación del objeto, así como todas las rutas del PC del fichero de texto ByteVector.txt.

Otros tipos de datos de bloque

El ejemplo anterior muestra cómo utilizar una matriz de 1-D Byte. Puede también declarar matrices 2-D de tipo Byte, Entero (Integer), Largo (Long) y Sencillo (Single). Además, puede controlar si desea que el objeto sea sólo de lectura (read-only), o de lectura-escritura (read-write).

2.00.A

Traducción Española: Alicia Bernal, revisión: Pablo Pompa www.superrobotica.com