**Dispositivos de Lógica programable. Variables en CUPL.**

Como en todo Lenguaje existen formas específicas para declarar las variables. Así mismo se usa cierta sintaxis para declarar las funciones lógicas.

**[ ! ] var [ .ext ] = exp;**

* **Var** es una variable o lista de variables indexadas o no indexadas. Si se trata de una lista, la expresión se aplica a los diferentes elementos de la lista.
* **.ext** es una extensión a las variables que las vincula con las características de los dispositivos programables.
* **Exp** es una expresión, que es una combinación de variables y operadores.
* **=** el operador de asignación que se aplica a una variable o a un conjunto de variables
* **!** El operador de complemento.

El conjunto de variables representadas por **.ext** son:

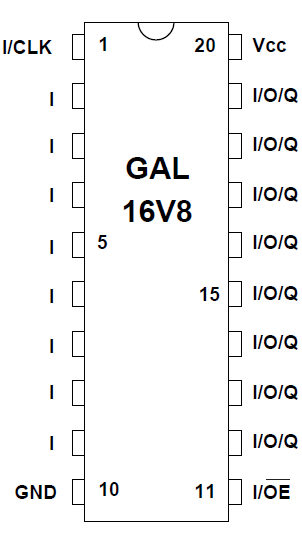


Para este caso podemos definir que un programa hecho en CUPL utiliza este tipo de representación de las variables para indicar que hace una entrada o pin.

O sea que estas extensiones le avisan al compilador sobre la configuración inherente de cada terminal dado el tipo de dispositivo usado.

Para este ejemplo usaremos un registro de carga paralelo que utiliza reloj para un latch. Para ello usaremos un dispositivo GAL16V8 que tiene la siguiente disposición de terminales para su uso.

Este circuito carga los flip flop tipo “D” internoa al sincronizarlos con el reloj. Es importante mencionar que un flip –flop tiene una entrada D y una entrada sincrona de reloj clk y una salida Q. Estas se declaran en la estructura funcional del programa.



Como primer paso seleccionaremos las terminales de entrada de la 2 a la 9 . La terminal 1 será el reloj y las salidas son de la 12 a la 19. La terminal 11 se usará para definir la salida de tercer estado de habilitación.

Creamos el dispositivo usando el WCPUL5.0 daremos

Entradas 8

Salidas 8

Intermedias 0

La tabla de verdad es

~ CE = L O7-O0

~ CE= H CLK= pulso D7  = I7  , D6  = I6  , .. , D0  = I0

**NOTA : USAREMOS PARTNO CPLD**

**Con esta forma de codificar evitamos la necesidad de crear las funciones booleanas.**

Name Latch ;

PartNo CPLD ;

Date 05/10/2014 ;

Revision 01 ;

Designer Engineer ;

Company ESIME ;

Assembly None ;

Location ;

Device g16v8a ;

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* INPUT PINS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PIN 1=clk;

PIN 11=oe;

PIN [2,3,4,5,6,7,8,9]=[i0,i1,i2,i3,i4,i5,i6,i7];

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT PINS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PIN [12,13,14,15,16,17,18,19] = [q7..0];

q0.D=i0; /\* la salida q0 es d0\*/

q1.D=i1; /\* la salida q1 es d1\*/

q2.D=i2; /\* la salida q2 es d2\*/

q3.D=i3; /\* la salida q3 es d3\*/

q4.D=i4; /\* la salida q4 es d4\*/

q5.D=i5; /\* la salida q5 es d5\*/

q6.D=i6; /\* la salida q6 es d6\*/

q7.D=i7; /\* la salida q7 es d7\*/

**Desarrolle el ejercicio en Proteus. Use el archivo GAL16V8 Latch.zip que tiene el modelo de Proteus mostrado abajo. Codifique el archivo de Latch.pld compile y cargue en el AM16V8 el archivo JEDEC.**

El uso de estos modificadores en las variables no siempre es necesario, depende mucho del dispositivo usado. Simulemos usando Proteus. El reloj es de 2 hertz Un botón controla el reloj y otro el CE que pone la salida en tercer estado.

