VHDL Made Easy

By BorjaLive (B0vE)

Índice

Descripción

VHDLME o, para abreviar, VME es un lenguaje de descripción de hardware intermedio con una sintaxis orientada a scripts.

El código en VME debe ser traducido primero a VHDL para después ser compilado. Las herramientas ayudas sintácticas están disponibles para Notepad++ y el compilador se distribuye en dos versiones, con y sin interfaz gráfica.

Esta idea surgió a partir de haber observado por primera vez VHDL. En comparación con otros lenguajes de programación convencionales, aparenta una tipología demasiado estricta y una sintaxis enrevesada, compleja y, en algunos momentos, inconsistente; teniendo en cuenta que se trata, no de un lenguaje propiamente, sino de una notación como lo podría ser JSON.

Dos personas llegaron a esta conclusión.

VME viene a simplificar los lenguajes de descripción de hardware aportando una mayor flexibilidad y legibilidad al código.

Actualmente se encuentra escrito en AutoIt3, un lenguaje de scripting para Windows; nada aconsejable para proyectos de esta índole. Decisiones del creador.

Más descripción de como surgió esta cosa

Sintaxis

Inspirada en C y adaptada a las peculiaridades de VHDL.

VME no distingue mayúsculas y minúsculas. No utiliza ‘;’ para cerrar líneas. La comprobación de igualdad y la asignación son iguales “=”

Secciones

Se definen con limitadores de inicio y fin; adicionalmente pueden incluir un modificador en el limitador de apertura.
Su uso no es obligatorio; no obstante, no está permitido alternar el uso de una sección en un mismo fichero. Es decir, si usas una sección para delimitar cierta porción de código que comparte la misma función, no podrás escribir ese mismo tipo de sentencia fuera de las secciones.

Var … VarEnd

No acepta modificadores.

Se utiliza para declarar las variables. Estas pueden ser puertos o señales. Más sobre variables en PAG X.

Logic [MOD] … LogicEnd

Acepta los modificadores “parallel” y “sequential”

Contiene todo tipo de sentencias excepto declaración de variables. Al usarse sin modificadores, admite tanto código paralelo como secuencial, al agregar los modificadores estos restringen las sentencias al tipo que definen.

Nota: Si no se usan secciones, el compilador ignorará todas las líneas que no pueda parsear; si se usan, y la línea está dentro de las secciones, pero no puede parsearla, lo mostrará como un error.

Implement … ImplementEnd

No acepta modificadores

Obligatoria para asignar los componentes de la FPGA con los puertos de la entidad.
NOTA: Solo se utilizarán si se le pide al compilador que genere las restricciones.
NOTA: En la versión actual 1.0.2 aún no está implementado.

Asignaciones

Sintácticamente se realizan igual que en la mayoría de lenguajes de alto nivel.

```
Variable = Expresion
```

Conversiones

Simulan las conversiones explicitas de C

```
VariableConvertida = (Tipo) VariableAconvertir
```

Los tipos a convertir son:

(integer)

NOTA: En la versión actual, 1.0.2, se llama (int)

(binary[x])

Dónde ‘x’ es el tamaño del nuevo array.

Funciones

Las funciones de VME son una simplificación de las funciones basadas en módulos de apoyo conocidos. El compilador sabe que realizan las

funciones, gracias a ello puede remplazarlas con expresiones incluidas en IEEE para no depender de otras librerías. Su sintaxis es:

```
Función Argumento1 Argumento2 … ArgumentoN
```

Las funciones permitidas son las siguientes:

LCDFG: ANDz, ORz, NANDz, NORz, XORz, XNORz

Dónde 'z' es el número de entradas de la puerta lógica. La última variable siempre será la salida.

Estructuras concurrentes

VME cuenta con las cuatro estructuras que surgen de: asignar directamente un valor o ejecutar una expresión y comprobar una igualdad o una expresión.

NOTA: En VHDL estas estructuras exigen que el último caso involucre a todas las otras opciones, lo que es lo mismo, sea un "Else". VME no es tan rígido, si detecta que una estructura es ambigua, automáticamente asignará al último caso la condición "ELSE" y lo notificará con un Warning.

SetIf (When … Else)

Realiza una asignación múltiple evaluando expresiones.

```
Set Variable
Valor1 If Expresion1
Valor2 If Expresion2
Valor3 If Else
```

ValorX puede ser una variable o un literal.

Es una estructura paralela.

SetSwitch (With … Case)

Realiza una asignación múltiple evaluando igualdades.

```
Set F Switch Variable
Valor1 Case Expresion1a|Expresion1b
Valor2 Case Expresion2
Valor3 Case Else
```

ValorX puede ser una variable o un literal.

Expresión tiende a ser un literal.

Es una estructura paralela.

NOTA: VHDL no permite realizar esta estructura para comprobar cadenas, si VME detecta esto lo convertirá a un SetIf y lo mostrará con un Warning.

IfThen (If … Then)

Realiza diferentes instrucciones evaluando expresiones.

```
If Expresion1 Then Instruccion1
Else If Expresion 2 Then Instruccion2
Else Instruccion3
```

Es una estructura secuencial.

SwitchCase (Switch … Case)

Realiza diferentes instrucciones evaluando igualdades.

```
Switch Variable
Instrucción1 Case Valor1a|Valor1b
Instrucción2 Case Valor2
Instruccion3 Case Else
```

Es una estructura secuencial.

NOTA: En la versión actual, 1.0.2, no permite evaluar cadenas igual que el SetSwitch, pero se planea agregar esa funcionalidad.

Estructuras cíclicas

ForNext (For)

Ejecuta repetidamente un bloque de estructuras. Define un contador, no modificable, para contar los ciclos.

```
For contador In Numero
        Instrucciones …
Next
```

```
For contador In Variable
        Instrucciones …
Next
```

Si el límite se define mediante una variable, esta deberá ser un vector y se iterará hasta el número elementos de la variable.

Operadores

Para la asignación se usa '=' el sentido es de derecha a izquierda.

Para comprobar la igualdad también se usa '='.

Los operadores lógicos son "And", "Or" y "Not".

Para concatenar binarios, generalmente en una cadena, se usa '&'.

Guía práctica

Uso del compilador

El parser de VME es un freeware, open source, bajo licencia GNU GPL. Por tanto, el código fuente puede ser descargado libremente y su distribución o modificación se regulan bajo Copyleft.

Cualquiera con unos conocimientos mínimos de Autoit3 o con experiencia en la programación podría implementar el parser en una interfaz personalizada; pero si estas buscando una forma sencilla de compilar un .vme sin quebrarte la cabeza, ya he desarrollado dos versiones.

Línea de comandos: el binario puede ser llamado desde un símbolo del sistema. Los parámetros del compilador se pasan como argumentos.

Interfaz Gráfica: Tan sencillo como seleccionar la fuente y pulsar el botón grande que dice “Compilar”

Parámetros del compilador

NoHeader

Desactiva la cabecera por defecto que se escribe en el archivo de salida.
En línea de comandos es “--noHeader”

LibStrict

Fuerza el uso excluyente de las librerías IEEE, si se usaran elementos de otras librerías, estos se implementaran de otra forma.
En línea de comandos es “--libStrict”

Verbose

Muestra información detallada de las actividades del compilador.
En línea de comandos es “--verbose” o “-v”

Silent

Evita crear cuadros de dialogo emergente. No mostrará avisos y, si lo necesita, tomara decisiones por si mismo.
En línea de comandos es “--silent” o “-s”

La línea del error

Aunque VME puede detectar la línea en la cual se encuentra el error, lo cierto es que el compilador funciona de una forma tan interdependiente y destartalada, que los errores pueden ser descubiertos a la hora de la escritura y, al estar sintetizado el documento, a veces es imposible dar con la línea exacta.

Por lo general, si el error se encuentra en la cabecera de una estructura, se culpara a la última línea de la estructura.

Por si acaso, no te fíes demasiado de la línea que indique.

Errores y versiones

Bugs conocidos

Errores tan recurrentes que merecen nombre propio

Derrape: Cuando se recorren las líneas de un archivo y se encuentra una estructura de varias líneas, el bucle que avanza dentro de la estructura a veces no comprueba si es el final del archivo. Como resultado; si una estructura de varias líneas acaba justo en el final del documento, arrayOutOfBoundsException al canto. Derrapa.

Reporte de errores

El desarrollador está abierto a mensajes que informen de posibles errores en el código del compilador. (Así al menos parece que hay personas interesadas en VME y el desarrollador no piensa que su trabajo es un vano)

NOTA: En la versión actual, 1.0.2, prácticamente se descubren nuevos errores y bugs con cada nuevo archivo que se prueba. Es de esperar que queden pequeños trucos sintácticos que provocan el fallo del compilador y no son detectados a tiempo.

Control de versiones

En el repositorio de GitHub siempre estará la versión más reciente del código fuente y los últimos binarios disponibles.

En la página de SourceForge solo se publican las releases más estables.

Los binarios se distribuyen compilados para arquitecturas de 32 y 64 bits, con y sin compresión UPX.

Teóricamente VME puede ser ejecutado desde Windows 95 hasta en Windows 10. No se esperan portes a otros sistemas operativos.