HardScratch

By BorjaLive (B0vE)

HardScratch v1.0 OpenWound

Índice

P1. Descripción

P3. Piezas

P9. Guía práctica

P9. Diseño

P11. Implementación

P12. Simulación

P14. Errores y versiones

Descripción

HardScratch, o HS para abreviar, es una interfaz de programación visual orientada a los lenguajes de descripción de hardware.

Los tokens de HS deben ser traducidos primero a un lenguaje intermedio antes de ser compilados. Actualmente, VME2 es el único módulo de exporte, permite generar código VHDL real.

HS es un proyecto con licencia GPL y todos sus componentes son de código abierto. Está escrito mayoritariamente en Java y compilado con OpenJDK 11, aunque cuenta con partes implementadas en Autoit3. En 2006 Autoit 3.2 cambió de licencia, por esto se compila utilizando Aut2Exe 3.1, aún bajo GPL.

HardScratch viene a facilitar y hacer más accesible el modelado de hardware. No se requieren conocimientos previos de VHDL, aunque se recomienda una base en lenguajes programación.

Es importante remarcar que, aunque Autoit3 tiene buena integración con Wine y Java es conocido por ser multiplataforma, las releases de HardScratch se publicarán para Windows en exclusiva. Aunque esto no cierra las puertas a una futura versión para GNU Linux.

A demás de los módulos en Autoit, el código en Java se apoya en la librería LWJGL para graficos OpenGL en Java y GHDL, un compilador GPL para VHDL.

Este proyecto ha sido realizado por Borja López, alias B0vE, para el concurso universitario de software libre 2019. https://concursosoftwarelibre.org

Notas de la v1.0.x

Este manual de usuario está escrito para la versión v1.0 BETA y LTS, por tanto, puede ser incompatible con funciones futuras del lenguaje. Bastantes cambios que rompen códigos escritos para la 1.1 LTS han sido introducidos.

Ya puedes dejar de leer, el resto de esta página es relleno.

La versión del lenguaje viene definida por el número de iteración del motor. Dado que HardScratch sigue el modelo de aproximaciones sucesivas, nunca existirá una versión definitiva ni una versión concreta; sino un conjunto discreto de estados caracterizados por el numero de veces que se ha reiniciado el desarrollo (X), numero de veces que se han hecho cambios de diseño (Y), número de veces que el creador pensó que había solucionado todos los errores conocidos (Z). Dando como resultado X.Y.Z. Dado que este proyecto es una continuación de VHDLME, es posible escribir HardScratch 1.0 como VHDLME 2.0.

Las versiones Beta no reciben actualizaciones, mientras que las LTS tampoco las reciben. La denominación Beta o LTS ha sido escogida por mera estética.

Piezas

Inspiradas en Scratch, las piezas son las unidades fundamentales del diseño. Están formadas por campos de texto, huecos para tips, constructores de expresiones y puertos de conexión. HS no distingue mayúsculas y minúsculas en el nombre de variables.

Las piezas pueden clasificarse en tres secciones principales.

Definición de variables

Declaran entras, salidas, señales y constantes. Opcionalmente pueden inicializar señales y constantes.

Declarator

Declara una variable. Consta de una entrada de texto, que será el nombre de la variable; y dos huecos. En uno de los huecos se selecciona el E/S, permite:

- Entrada: No puede ser asignada ni inicializada.
- Salida: No puede ser leída ni inicializada.
- Señal: Puede ser leída, escrita e inicializada.
- Constante: Debe ser inicializada, puede ser leída pero no escrita.

El segundo agujero selecciona el tipo de dato de la variable:

- Entero (Integer): Valor numérico entero de 8 bits.
- Bit (Bit): Una variable booleana.
- Cadena de bits (Bit array): Conjunto ordenado de bits.

Tiene dos puertos. Un puerto Inicializer y un puerto extravar.

Extravar

Consta de una entrada de texto y tres puertos. Mediante el puerto superior puede ser conectado a un Declarator para agregar otra variable con el nombre escogido del mismo E/S y tipo de dato que el declarator.

Mediante el puerto inferior se pueden conectar otros Extravar y formar cadenas de variables del mismo tipo. Cuenta con un puerto Inicializer.

Inicializer

Tiene un constructor de tipo I. Se conecta mediante un puerto Inicializer a Declarators o Extravars e inicializa la variable deseada.

Importante: El inicializar puede ser conectado a cualquier Declarator, pero intentar inicializar una Entrada o Salida producirá un error.

Lógica paralela

Su finalidad es realizar operaciones concurrentes. Como su nombre indica la ejecución se realiza en paralelo, por lo que no es posible secuenciar recursivamente estas piezas.

Asignator

Tiene un hueco para variables y un constructor de tipo A. Su funcionamiento es muy simple, asigna la expresión del constructor a la variable del hueco.

Converter

Se compone de dos huecos para variables. En el hueco de la derecha se selecciona la variable a convertir y en izquierdo la variable destino a asignar.

Las conversiones permitidas son Entero a Bit array y Bit array a Entero. Cualquier otra combinación llevará a error.

SetIf

Asignación condicional. Es una estructura que consta de un hueco para la variable a asignar y un número de filas con dos constructores, uno E y otro A. El constructor de la izquierda genera la condición bajo la cual la expresión generada en el constructor de la derecha será asignada a la variable seleccionada.

El último caso tiene un solo constructor tipo A. Si no se diera ninguna de las condiciones anteriores, se realizaría esta última. Es obligatorio completar el constructor de este caso.

SetSwitch

Asignación condicionada al valor de una variable. En un hueco se selecciona la variable a asignar y en otro la variable a comprobar. El funcionamiento es similar al SetIf, diferenciándose en el tipo de los constructores; los dos son tipo A. Cuando el valor generado en el constructor de la izquierda es igual a la variable a comprobar, el valor generado en el constructor de la derecha es asignado a la variable seleccionada.

Igualmente consta con un último valor que será asignado si ninguno de los anteriores se cumple. Es obligatorio completar este constructor.

Una última consideración en la lógica paralela. Asignar múltiples veces una variable llevará a error dado que todas las instrucciones se ejecutan al mismo tiempo. Por esto mismo no es posible ver reflejados los cambios de una estructura en otra, hasta una ejecución siguiente.

Lógica secuencial

Se inicia mediante una pieza que da comienzo al proceso y se van uniendo en serie las distintas estructuras. Son bastante más avanzadas que las paralelas. Los procesos tienen memoria implícita, esto es, si una variable no es modificada, esta mantiene su valor.

Sequencer

Iniciador de bloque secuencial. Es el punto de inicio de un proceso. Consta de un único puerto SEQ-P al cual se conecta cualquier estructura secuencial. Es importante remarcar que el orden de ejecución será de la pieza más cercana al Sequencer hasta la más lejana.

Todas las piezas secuenciales tienen al menos dos puertos SEQ-P. Uno superior que se conecta a la estructura predecesora y otro inferior al cual se conectara una estructura siguiente.

SEQ Asignator

Asignador secuencial. Tiene un hueco para variables y un constructor tipo A. Igual que el asignador. Asigna el valor de la expresión generada en el constructor a la variable seleccionada.

If Then

Tiene un número de filas compuestas por un Constructor de tipo E y un puerto SEQ-P. Cuando la expresión del constructor es verdadera, se ejecutan las estructuras conectadas al puerto de la fila. Es una estructura recursiva, permite incluir cualquier pieza secuencial, incluida ella misma, en sus puertos.

Al igual que los SetIf y SetSwitch, si una fila es ejecutada, las filas que la procedan serán olvidadas. También, su última fila carece de constructor y se ejecuta si ninguna de las otras condiciones fue

verdadera. A diferencia de las otras estructuras, esta última fila puede no estar definida.

Switch Case

Esta pieza es al If Then como el SetSwitch es al SetIf. Los constructores de tipo E han sido remplazados por tipo A. Las estructuras conectadas al puerto SEQ-P de una fila serán ejecutadas si el valor de la variable seleccionada coincide con el generado por el constructor de dicha fila.

Igualmente, la última fila se activa si ningún caso previo fue verdadero. No es estrictamente necesario completar esta fila, pero el Switch Case debe contener todas las posibilidades de la variable, una forma rápida de charlo es con la condición Else la última fila.

Wait For

Pone el proceso a dormir un número determinado de nanosegundos. Este número se genera mediante un constructor de tipo A.

Wait On

Pone el proceso a dormir hasta que se dé un evento en una determinada variable. Los posibles eventos son:

- Flanco de subida (Rising Edge): La variable toma valor alto.
- Flanco de bajada (Lowering Edge): La variable toma valor bajo.

For Next

Actualmente, en la versión v1.0 el For Next no está definido.

Guía práctica

La interfaz visual se compone de pantallas, cada una para una terea específica. Siempre es posible regresar al menú principal desde cualquiera de las otras pantallas.

Menú principal

Muestra la versión de HardScratch. Desde él se puede ir a la pantalla de Abrir, Crear y Ajustes. Adicionalmente tiene un botón ayuda para ir a esta guía de usuario. (Requiere conexión a Internet)

Ajustes

Permite elegir tres configuraciones. Para aplicar los cambios es necesario reiniciar el programa.

- Tema: Paleta de colores clara u oscura.
- Modo de ventana: Pantalla completa o ventana o con marco.
- Resolución: Tamaño en pixeles reales de la ventana. Se debe elegir de entre una lista, lista completa en página 17.

Abrir

Muestra los proyectos existentes. Permite cargarlos o eliminarlos.

Crear

Crea un nuevo proyecto con el nombre introducido. Acepta espacios. Los proyectos se crearán en el directorio de datos de HS y no es posible moverlo.

Las pantallas de diseño, implementación y simulación se explicarán en su propio apartado.

Diseño

La finalidad de esta pantalla es construir un circuito mediante los elementos comunes en los lenguajes de modelado de Hardware.

Finder

También llamada bandeja, es un panel móvil en el que aparecen los elementos que pueden ser arrastrados al tablero. Al pulsar sobre un espacio vacío en el tablero, el Finder mostrará todas las piezas. Cuando se seleccionan Huecos el Finder mostrará los posibles tips que podrían ser asignados.

Tablero

Espacio sobre el cual se distribuyen las piezas. Mediante click derecho se puede arrastrar todo el tablero, o, moverse por él mismo. Pulsando el botón suprimir (Del) se elimina la pieza que está siendo arrastrara o la última pieza seleccionada.

Algunos tips tienen elementos internos, estos pueden ser seleccionados y eliminados independientemente del tip.

Consideraciones sobre los huecos. Un hueco solo puede ser asignado cuando está seleccionado, si se arrastra un tip a un hueco no seleccionado, aunque sea compatible, este no se asignará. En el caso de los tips de variables; todas las variables, sin importar el tipo, pueden ser asignadas. Sin embargo, algunas requieren poder ser leídas y otras poder ser escritas, si la variable seleccionada no satisface esa condición llevará a error.

Si un tip de variable ha sido asignado y el Declarator o Extravar que definía la variable ha desaparecido, el tip se mantendrá. Si no se elimina llevará a error.

Las piezas cuentan con puertos para conectarse entre sí. Estos puertos pueden ser macho o hembra, es necesario que su genero sea distinto para que puedan conectarse. Al arrastrar una pieza, los elementos conectados a sus puertos hembra también se moverán con ella. En caso contrario, las uniones realizadas por puertos macho de una pieza que es arrastrada se rompen. Al eliminar piezas se realiza la misma búsqueda en árbol; cuando una pieza se elimina, todas las que estén conectadas a sus puertos hembra también se eliminaran.

Los constructores permiten generar expresiones, se dividen en los siguientes tipos, por los elementos de los que disponen:

- Tipo I: Inicialziador. Puede usar literales y operadores aritméticos y lógicos, pero no variables.
- Tipo A: Asignación. Puede usar literales y operadores aritméticos y lógicos además de variables.
- Tipo E: Expresión. Puede usar literales, operadores aritméticos y lógicos, variables y el comprobador de igualdad igualdad.

Si se mantiene pulsada la tecla suprimir (Dell), todos los elementos del tablero serán eliminados. Ideal por si se te han perdido piezas y no sabes dónde están.

Todo el tablero forma parte de la misma arquitectura, pero los procesos son independientes, siempre y cuando tengan un Sequenser de inicio diferente.

En el caso de las variables de tipo Bit Array, es posible seleccionar un tip de subArray. Es equivalente a una variable de tipo Bit. Igualmente las variables de tipo Bit pueden ser utilizadas como señal de reloj en constructores, devuelve verdadero cuando se ha llegado al flanco de subida.

Implementación

Trata de asociar variables de entrada o salida a elementos E/S de una placa ficticia. Esta placa está inspirada en una Xilinx Espartan-3E, aunque cuenta con elementos adicionales para facilitar la simulación. También se conoce como Restricciones.

Esta interfaz no restringe el tipo de dato a asociar, únicamente Entradas y salidas. Se aconseja sentido común.

La placa simulada, llamada Borelicious-3E tiene los siguientes elementos:

Interruptor de encendido

Se corresponde con conectar a corriente la placa y cargar el programa que fue diseñado y restringido. No puede ser asignado

Botón de reinicio

Comienza de nuevo la simulación, sin realcalizar el diseño. No puede ser asignado.

Pantalla LCD

Carece de funcionalidad como elemento IO. Se utiliza únicamente para mostrar mensajes sobre la simulación. No puede ser asignada.

Interruptor

La placa contiene 4. Son elementos de entrada para una variable de tipo Bit o un subArray de un Bit Array.

Botón

La placa contiene 4. Son elementos de entrada para una variable de tipo Bit o un subArray de un Bit Array.

Reloj

La placa contiene 1. Son elementos de entrada para una variable de tipo Bit o un subArray de un Bit Array. Se estado cambia periódicamente.

LED

La placa contiene 8. Luces que se encienden cuando el Bit o subArray asignado es 1.

Display de 7 segmentos

La placa contiene 2. Muestran todo tipo de variables. La placa original Espartan-3E no los tiene.

Simulación

Al encender la placa simulada se comprobará el diseño para prevenir erres. Si se descubre un error en esta fase o en tiempo de ejecución, la interfaz volverá a la pantalla de diseño y mostrará la pieza que causa el error junto a una descripción del fallo.

El display LCD muestra el estado de la placa. La placa acepta cambios mientras está en modo “SIMULATION RUNNING”. Es aconsejable no tocar la placa mientras esté en “COMPUTING…”.

Para interactuar los con los elementos de salida basta con hacer clic sobre ellos.

Al cambiar de pantalla, el estado de la simulación se pierde. Cuando se regrese a la pantalla la placa se encontrará apagada y la memoria secuencial se habrá perdido.

Errores y versiones

Bugs conocidos

Errores tan recurrentes que merecen nombre propio

KeepWaiting: Se conoce que si VMESS o GHDL entran en un bucle infinito, HardScratch se quedará esperando indefinidamente hasta que terminen de ejecutarse.

Error handling: No todos los casos de error están registrados. Es posible que la simulación falle y no se indique ningún error.

TipCojo y TimeTraveller: al realizar movimientos colectivos automáticos o cargar de golpe todas las piezas y sus elementos, es posible que se pierdan conexiones o que un tip aparezca descentrado.

Reporte de errores

Si eres un parte del grupo BETA TESTER; primero que todo, muchas gracias. Segundo, la copia que se te ha suministrado incluye un launcher con el modo Debug activado. Puedes reportar errores y adjuntar el log en cuando el programa se cierra.

Si eres usuario de la versión “Estable” puedes reportar errores en el proyecto de Github.

NOTA: En la versión actual, v1.0, prácticamente se descubren nuevos errores y bugs con cada nuevo diseño que se prueba. Es de esperar que queden pequeños trucos provoquen el fallo del simulador 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.

Puedes descargar el programa portable o la versión con instalador, con y sin Java. Si no descargas la versión de Java junto al programa, necesitarás tener el JRE de Java 11 o superior.

La interfaz HardScratch y sus componentes VMESS y VME2 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.