

RIPES

Índice:

1	
1 Simulador	2
1.1 Introducción	2
1.2 Interfaz	2
1.3 Memorias Caché	2
1.4 Pipeline	3
1.5 Entrada Salida	4
1.6 Trabajo con el simulador	4
1.7 Adicionales	5
2	
2 Instalación y Ejecución	5
2.1 Linux	5
2.2 Windows	6
3	
3 Complicaciones Encontradas	7
4	
4 Referencias	8

1 Simulador:

1.1 Introducción:

Este documento se dedicará a la exploración del simulador del estándar RISC-V [1], RIPES [2], sus posibilidades y sus principales funciones.

RIPES es un simulador del estándar RISC-V, compatible con varias implementaciones del mismo (rv32im, rv64im, rv32imc, etc.), de libre distribución, escrito principalmente en C++ y orientado a la ejecución sobre un único fichero ensamblador.

1.2 Interfaz:

RIPES cuenta con una implementación muy gráfica para la interfaz de trabajo desde la ejecución de los programas, la visualización de los espacios de memoria, las fuentes y periféricos, etc. Por otro lado, el trabajo general con el código es bastante cómodo, con una visualización muy gráfica y dinámica dentro del proceso de ejecución y depuración.

1.3 Memorias Caché:

Para la visualización de la memoria, como se había dicho con anterioridad, RIPES cuenta con una visualización muy gráfica y detallada. Se muestra bastante pulida, tanto para la memoria principal como para la caché, donde se distingue entre caché de datos y de instrucciones, y podemos ver exactamente donde se ubican los bloques junto a su contenido.

A diferencia de otros simuladores que solo muestran si se ha producido un fallo en caché, o muestran apenas un pequeño esbozo, esta implementación muestra todo lo contrario entrando en bastante detalle dentro del proceso.

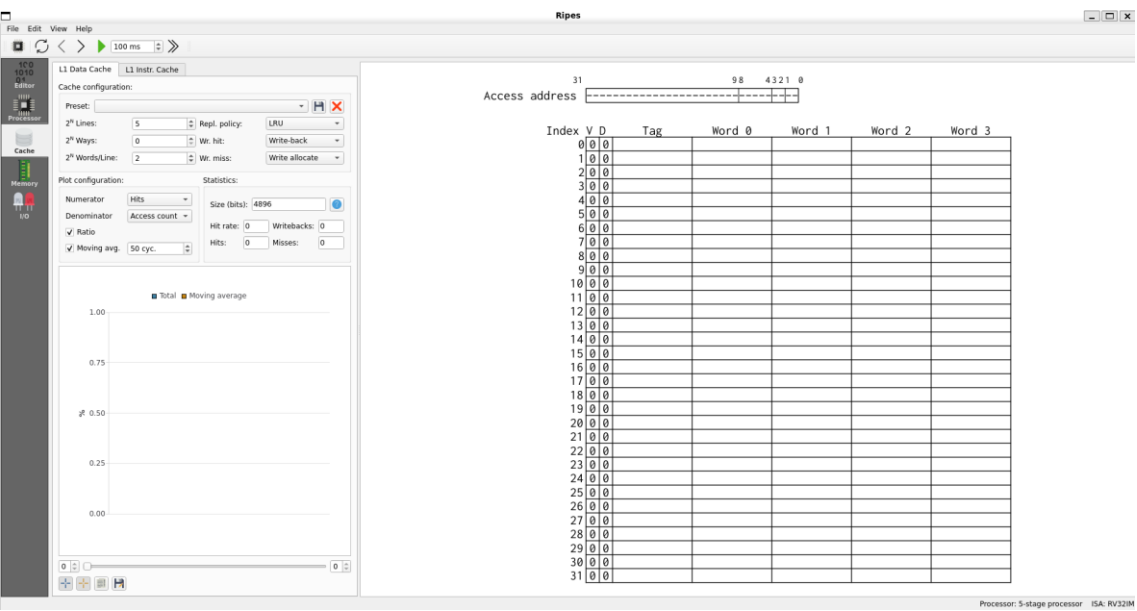
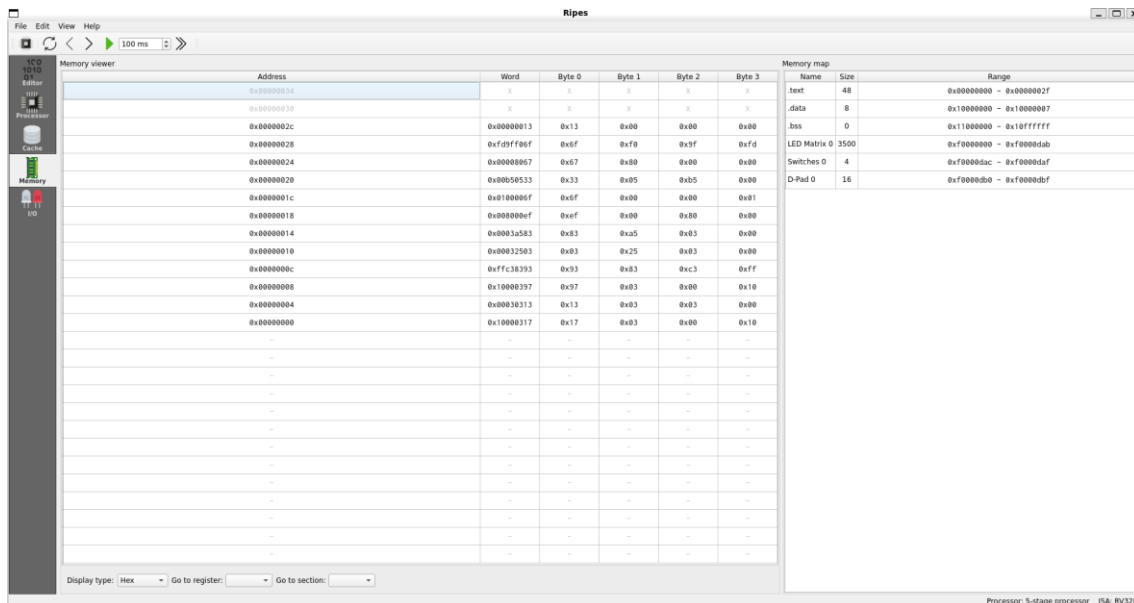


Figura 1.1



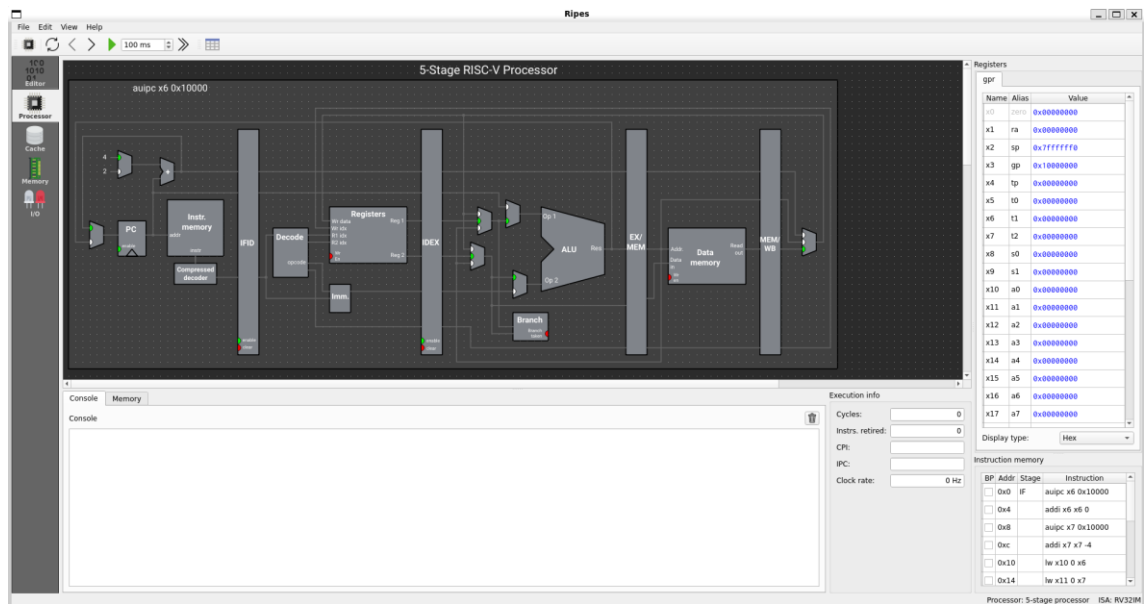


Figura 1.4

1.5 Entrada Salida:

Incluso cuando este simulador cuenta con un apartado específico para la entrada salida, la implementación de la misma es solo programada puesto que las interrupciones y excepciones no están implementadas en el simulador. Adicionalmente a esto, solo cuenta con tres periféricos implementados.

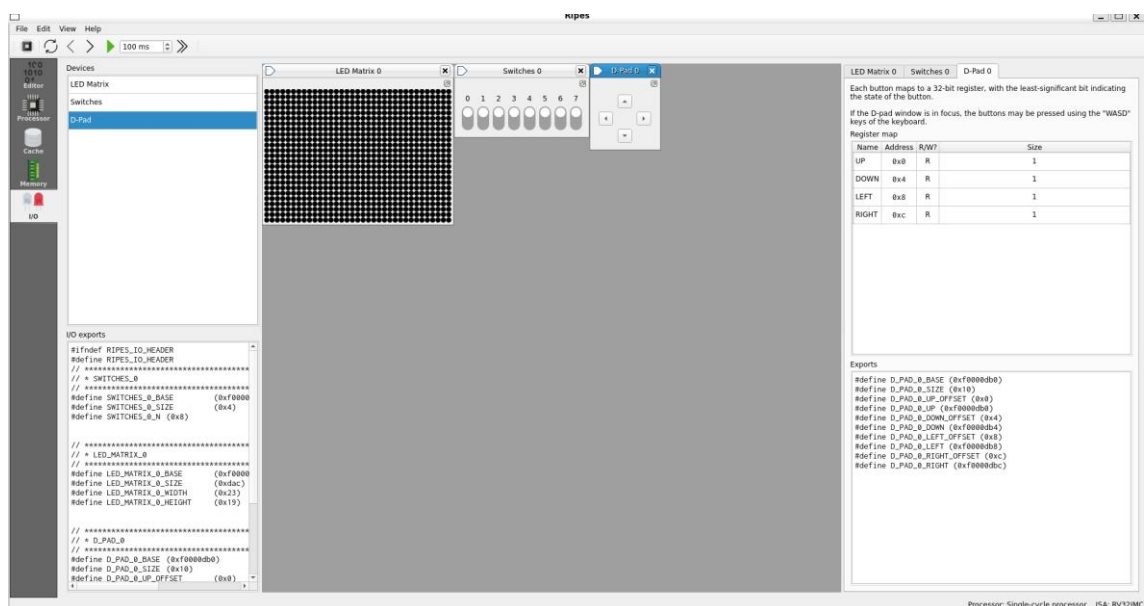


Figura 1.5

1.6 Trabajo con el simulador:

Para trabajar con un único fichero, en el que se encuentre contenido todo el código del programa este simulador es casi ideal: cuenta con un editor en el que podemos ver a su misma vez el código fuente junto al desensamblado cuando se carga el programa ejecutable, cuenta con un sistema de búsqueda por etiquetas en el programa, se resalta la instrucción que se está ejecutando en el momento y

cuenta con un sistema de ejecución automática con una velocidad configurable dentro de unos límites.

Sin embargo, si contamos con un proyecto formado por varios ficheros de código, o incluso si queremos ver varios proyectos en el simulador y trabajar con ellos de una manera cómoda, como si de un editor de texto general se tratase, nos encontramos en un aprieto puesto que este simulador no cuenta con soporte para la visualización de varios ficheros al mismo tiempo, y adicionalmente, en caso de querer generar un programa a partir de varios ficheros de código se tendrá que montar desde fuera puesto que no se cuenta con soporte esto desde el propio simulador.

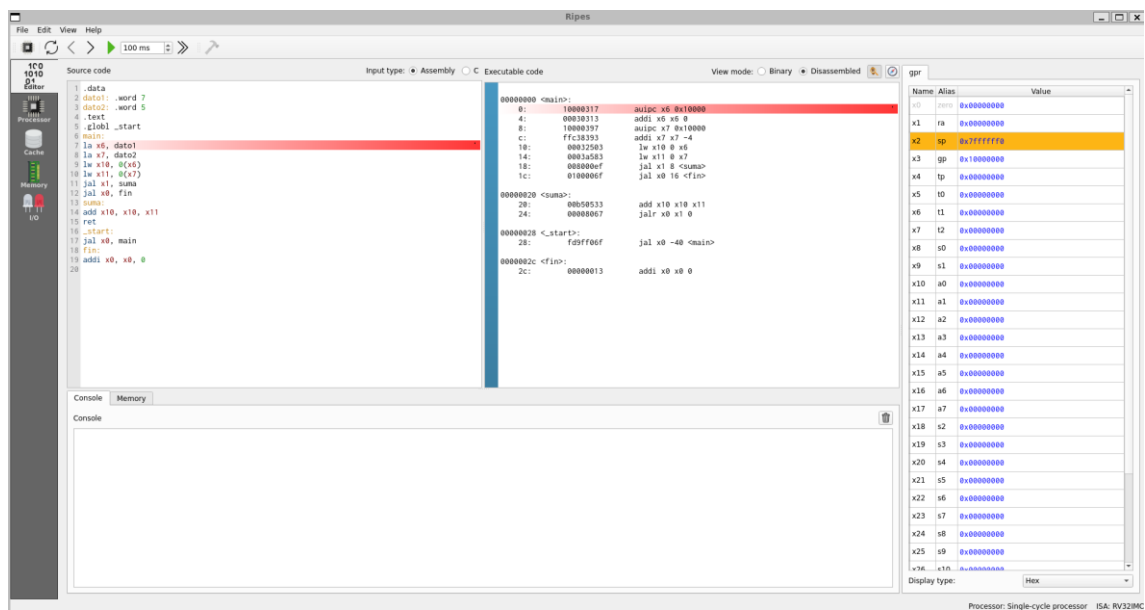


Figura 1.6

1.7 Adicionales:

Adicionalmente, como aspecto positivo a resaltar de este simulador frente a otros, RIPES permite cargar programas ya generados desde fuera del simulador, es decir: si se cuenta con un programa ejecutable ya generado el simulador puede procesarlo como un programa normal, aunque no se podrá modificar el código desde el mismo. Esto le permite suplir la incapacidad de generar o trabajar con proyectos compuestos por varios ficheros de código, puesto que, aunque no se puedan generar desde dentro, se pueden generar fuera y seguir ejecutándose desde dentro.

Seguindo con el anterior punto, si se cuenta con el compilador de C para RISC-V adecuado, se puede trabajar con programas que estén escritos en C, ya no solo en ensamblador. Y si se genera el programa desde fuera, podrían estar escritos en otros lenguajes de programación como por ejemplo C++, o inclusive mezclar varios ficheros de código escritos en distintos lenguajes de programación como C y ensamblador.

2 Instalación y Ejecución:

Dentro de este apartado se describirá el proceso necesario para la instalación y configuración inicial del simulador tanto para los sistemas operativos tipo Windows como tipo Linux.

2.1 Linux:

Dentro del apartado “Releases” del repositorio oficial [2] nos encontramos con las diferentes versiones que hay del simulador hasta la fecha. En este punto se elige preferentemente la versión más actualizada y se procede con la selección de la versión adaptada para el sistema operativo sobre el que se vaya a trabajar, en este caso, Linux.

Continuous release Pre-release

Commits

- [56dc4b2](#) : Attempt WASM event filter fix (Morten Borup Petersen)
- [2209967](#) : Windows fix (Morten Borup Petersen)

Assets 5






 Ripes-v2.2.6-61-g2209967-linux-x86_64.AppImage	30.5 MB	3 weeks ago
 Ripes-v2.2.6-61-g2209967-mac-universal2.zip	61.8 MB	3 weeks ago
 Ripes-v2.2.6-61-g2209967-win-x86_64.zip	15.3 MB	3 weeks ago
 Source code (zip)		3 weeks ago
 Source code (tar.gz)		3 weeks ago

Figura 2.1

Una vez se tiene instalado y ubicado el archivo de extensión “AppImage” en el sistema, ya se podrá iniciar el simulador ejecutando este mismo archivo. Es posible que la ejecución falle porque no se tienen instaladas las dependencias necesarias en el sistema.

```
bruno@ABURKOS:~/riscv/ripes$ ls
Ripes-v2.2.6-61-g2209967-linux-x86_64.AppImage
bruno@ABURKOS:~/riscv/ripes$ ./Ripes-v2.2.6-61-g2209967-linux-x86_64.AppImage
dlopen(): error loading libfuse.so.2

AppImages require FUSE to run.
You might still be able to extract the contents of this AppImage
if you run it with the --appimage-extract option.
See https://github.com/AppImage/AppImageKit/wiki/FUSE
for more information
```

Figura 2.2

En este caso solo se necesitaría instalar las mismas para poder empezar a trabajar, en este caso no se cuenta con las librerías de fuse [3]. Si se cuenta con apt en el sistema, se puede hacer con la siguiente sentencia: “sudo apt install libfuse2”.

2.2 Windows:

Dentro del apartado “Releases” del repositorio oficial [2] nos encontramos con las diferentes versiones que hay del simulador hasta la fecha. En este punto se elige preferentemente la versión más actualizada y se procede con la selección de la versión adaptada para el sistema operativo sobre el que se vaya a trabajar, en este caso Windows.

Continuous release Pre-release

Commits

- [56dc4b2](#) : Attempt WASM event filter fix (Morten Borup Petersen)
- [2289967](#) : Windows fix (Morten Borup Petersen)

Assets 5






 Ripes-v2.2.6-61-g2209967-linux-x86_64.AppImage	30.5 MB	3 weeks ago
 Ripes-v2.2.6-61-g2209967-mac-universal2.zip	61.8 MB	3 weeks ago
 Ripes-v2.2.6-61-g2209967-win-x86_64.zip	15.3 MB	3 weeks ago
 Source code (zip)		3 weeks ago
 Source code (tar.gz)		3 weeks ago

Figura 2.3

Una vez hemos instalado el archivo comprimido, archivo de extensión zip. Para iniciar el simulador se tendrán que extraer los archivos comprimidos y llamar al ejecutable dentro de estos: “ripes.exe”.








 libGLESv2.dll	18/01/2025 12:47	Extensión de la ap...	3.306 KB
 Qt5Charts.dll	18/01/2025 12:47	Extensión de la ap...	1.386 KB
 Qt5Core.dll	18/01/2025 12:47	Extensión de la ap...	5.883 KB
 Qt5Gui.dll	18/01/2025 12:47	Extensión de la ap...	6.844 KB
 Qt5Svg.dll	18/01/2025 12:47	Extensión de la ap...	323 KB
 Qt5Widgets.dll	18/01/2025 12:47	Extensión de la ap...	5.370 KB
 Ripes.exe	18/01/2025 12:47	Aplicación	6.276 KB

Figura 2.4

3 Complicaciones Encontradas:

A pesar de que el propio simulador es uno de los más completos dentro de su entorno, podemos encontrar algunos aspectos a mejorar. Por ejemplo: no cuenta con una implementación de interrupciones ni excepciones lo que puede dificultar la depuración de los errores; por otro lado, no cuenta con una instrucción para la finalización del programa, y cuando finaliza no se puede volver una instrucción hacia atrás como cuando se ejecuta normalmente, sino que se tiene que reiniciar la ejecución del programa desde el principio. Para finalizar, tenemos que el trabajo con varios ficheros de código, o incluso con varios proyectos al

mismo tiempo se vuelve bastante incómodo siendo que no puede trabajar con el mismo simulador como se podría con otros del estilo como RARS por ejemplo.

Como punto por fuera del simulador, el lenguaje y formato en que está escrito este simulador se puede considerar bastante complejo siendo casi una simulación hardware, lo que complica el añadir características o configuraciones a medida, o incluso cambiar algunas de las ya existentes.

Referencias

- [1] RISC-V, «Ratified RISC-V Specifications,» [En línea]. Available: <https://lfriscv.atlassian.net/wiki/spaces/HOME/pages/16154769/RISC-V+Technical+Specifications>.
- [2] mortbopet, «Repositorio Oficial de RIPES,» [En línea]. Available: <https://github.com/mortbopet/Ripes>.
- [3] Fuse, «FUSE,» [En línea]. Available: <https://github.com/AppImage/AppImageKit/wiki/FUSE>.
- [4] mortbopet, «Wiki oficial de RIPES,» [En línea]. Available: <https://github.com/mortbopet/Ripes/tree/master/docs>.