**RIPES**

**Índice:**

* Características
* Instalación y Ejecución
* Dependencias necesarias
* Complicaciones encontradas
* Documentación

**Características Generales:**

RIPES es un simulador de libre distribución, escrito principalmente en C++ usando también Qt. Esto, junto a que toda la documentación esté disponible, le da a RIPES cierta robustez si se quieren implementar cambios o limitar ciertas partes del mismo. Continuando con las características generales de este simulador: RIPES es un simulador de interfaz gráfica, que cuenta con una gran variedad de características, citando por encima las más interesantes tenemos: simulación de cachés, entrada salida, ejecución segmentada, se pueden ejecutar códigos escritos en ensamblador o C (para C es necesario tener previamente instalado un compilador de c riscv). Empezando por la simulación de memorias caché, tenemos una interfaz sencilla, pero que nos permite ver de una manera muy visual todas las partes de la misma, donde se nos permite redimensionar la misma para nuestras especificaciones concretas, en este apartado RIPES está bastante por encima de otros simuladores del estilo, que tienden a tener una representación más abstracta. Para la ejecución segmentada, no podemos configurar más allá de pequeñas cosas, donde además no podemos desactivar la misma. Sin embargo, al igual que con las cachés, la ejecución segmentada tiene apartado propio dentro del simulador y nos ofrece una experiencia bastante gráfica y visual. Continuando con el apartado de entrada y salida, RIPES cuenta con algunas herramientas para simular MMIO (Memory-Mapped I/O), como: una matriz de leds, unos interruptores, y un D-PAD. Entrando en la propia ejecución y ensamblado, el simulador cuenta con un compilador propio para el ensamblador, sin embargo, si se quiere compilar código c se necesita proporcionar previamente la ruta del un compilador c a riscv que genere el binario, otros simuladores como Venus o RARS nos permiten generar el archivo binario que podríamos simular con RIPES. Como conclusión: este simulador es muy robusto y completo, siendo que, en los temas de cachés, entrada-salida y ejecución segmentada se posiciona como una de las mejores opciones por su sencillez y lo visual que es, lo que viene muy bien tanto para las explicaciones como para las prácticas. Sin embargo, para los temas de ensamblador puro es posible que otras opciones como RARS o Jupiter puedan ser mejores opciones: para esta situación no es necesaria tanta carga de información, y además le resta el que no se pueda deshabilitar la ejecución segmentada.

**Instalación y Ejecución:**

Dentro de este apartado tenemos dos vertientes principales para completar una instalación correcta de este simulador, ambas con dependencias previas que se verán en el apartado de dependencias. Empezando por la manera más sencilla: estaríamos instalando el ejecutable directamente en vez de tener que compilar los archivos y montarlo nosotros. El ejecutable se puede encontrar en alguno de los enlaces dentro del repositorio oficial, o si se prefiere, a partir del siguiente enlace: <https://github.com/mortbopet/Ripes/releases>. Dependiendo de si estamos trabajando en Windows o en Linux tendremos dos situaciones distintas, en Windows tendremos que ejecutar el programa ripes.exe una vez extraídos los archivos del “.zip”, mientras que en Linux, nos tendremos que descargar el archivo “.AppImage”, que para poder ejecutar necesitar tener una serie de dependencias previamente instaladas. Luego, por otra parte, si no queremos instalar el ejecutable, podemos clonar todos los archivos del propio repositorio y montar el programa a partir de estos. Este proceso es un poco engorroso porque, necesita entre otros, tener instalado Qt y cmake para poder montar el ejecutable. Una vez se tiene el repositorio clonado, dentro del directorio Ripes, se deben ejecutar las siguientes sentencias: “cmake .” y “make”, como si se tratase de un proyecto cmake estándar.

**Dependencias necesarias:**

Para arrancar el programa en Linux, como es del tipo “.AppImage”, se necesitará de FUSE para esto. Véase la página oficial de FUSE, donde se pueden encontrar las distintas opciones para instalar dependiendo del entorno: <https://github.com/AppImage/AppImageKit/wiki/FUSE>

Para montar el ejecutable a partir del código fuente necesitaremos de tener instalado en el equipo CMake y Qt, de versión preferiblemente 6. Con respecto a Qt necesitaremos de tener el entorno bien configurado, sino, el montaje fallará irremediablemente, es decir no se debe instalar solo desde apt sino que hay que además hay configurar el entorno. Véase el enlace: <https://doc.qt.io/qt-5/cmake-manual.html>

**Complicaciones Encontradas:**

Dentro del propio simulador no se ha llegado a encontrar nada problemático, nos encontramos ante un simulador bastante completo. Sin embargo, el proceso de instalación puede volverse bastante complejo: Para empezar, en el caso de trabajar con Linux e instalar solo el ejecutable, sino se instala correctamente FUSE puede llegar a dar fallos en la ejecución, aún así esto puede considerarse minúsculo en comparación a los problemas que pueden surgir cuando se trata de montar el ejecutable desde el código fuente, puesto que en este caso se necesita instalar y configurar de manera correcta Qt, lo que en las pruebas se ha dificultado bastante. Si se conoce previamente Qt y como configurar el entorno esta puede ser una forma bastante viable, sin embargo, en caso contrario no estaría para nada recomendado.

**Documentación:**

Enlace al repositorio oficial:

<https://github.com/mortbopet/Ripes>

Enlace a la wiki oficial con los distintos enlaces de documentación:

<https://github.com/mortbopet/Ripes/tree/master/docs>