CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

Emulador de microprocesador Leon3 para desarrollo de software satelital y simuladores

Autor:
Ing. Iriarte Fernandez Nicolás Ezequiel

Director:
Lic. Nicolás Eduardo Horro (INVAP)

Jurados:
Nombre del jurado 1 (pertenencia)
Nombre del jurado 2 (pertenencia)
Nombre del jurado 3 (pertenencia)

Este trabajo fue realizado en la ciudad de Didcot, Oxfordshire, Reino Unido, entre Agosto 2024 y Diciembre de 2024.

Resumen

En la presente memoria se describe el diseño e implementación de un emulador de microprocesador Leon3 desarrollado para la empresa INVAP. SE, pensado para su uso en simulaciones espaciales. Este trabajo abarca las etapas de diseño, planificación, desarrollo y documentación, con el objetivo de ofrecer una herramienta robusta que se adapte a las necesidades de la industria espacial. Para su realización, se aplicaron conocimientos adquiridos en Arquitectura de Microprocesadores, Programación de Microcontroladores y Protocolos de Comunicación.

Agradecimientos

Esta sección es para agradecimientos personales y es totalmente **OPCIONAL**.

Índice general

Índice de figuras

Índice de tablas

Dedicado a... [OPCIONAL]

Capítulo 1

Introducción general

En este capítulo se presenta una introducción general al trabajo final, en el cual se describen los antecedentes, la motivación, los objetivos y alcance del trabajo. Además, se presenta el estado del arte en emuladores de sistemas espaciales.

1.1. Desarrollo de simuladores espaciales y emuladores

Para productos de ámbito espacial, como lo son los satélites, muchas veces es difícil, y en ocasiones imposible, generar escenarios realistas para pruebas de los elementos que los componen. Ya sea por no poder generar las mismas condiciones ambientales, o porque la naturaleza de la maniobra que se busca probar implicaría un daño a los equipos bajo revisión.

Otro desafío que se presenta es la escasez de hardware disponible para realizar dichas pruebas, ya que los componentes espaciales no se suelen tener en grandes cantidades, y suelen ser caros. Haciendo que el acceso a dichos recursos sea limitado.

En este contexto, es común replicar los elementos de interés de manera programada, cumpliendo con cierto grado de representatividad. De manera tal que se comporten de la manera más similar posible a su contraparte física. Estos elementos, íntegramente desarrollados en software, se los denominan emulados o simulados. Uno de los componentes que se suele tener mayor interés en simular es el procesador de la computadora a bordo.

El término computadora a bordo (OBC, por las siglas en inglés de *On Board Computer*) suele referirse a la unidad en la que se ejecuta el Software a bordo (OBSW, por sus siglas en inglés de *On Board Software*) y su rol principal es la orquestación del resto de subsistemas en el satélite. Esto incluye recolectar información de los periféricos conectados, analizarla y tomar las decisiones y acciones apropiadas cuando sea requerido.

En la figura 1.1 se muestra la relación entre simuladores, emuladores y software de vuelo. Los simuladores satelitales son más abarcativos e incluye todos los subsistemas simulados del satélite, entre ellos suele encontrarse el emulador de microprocesador. El emulador, por otro lado, se enfocan en reproducir el comportamiento del microprocesador específico a la misión. Por último, el software de vuelo es el software que se ejecuta en la OBC del satélite, y es el software que se busca probar en los emuladores de microprocesadores.

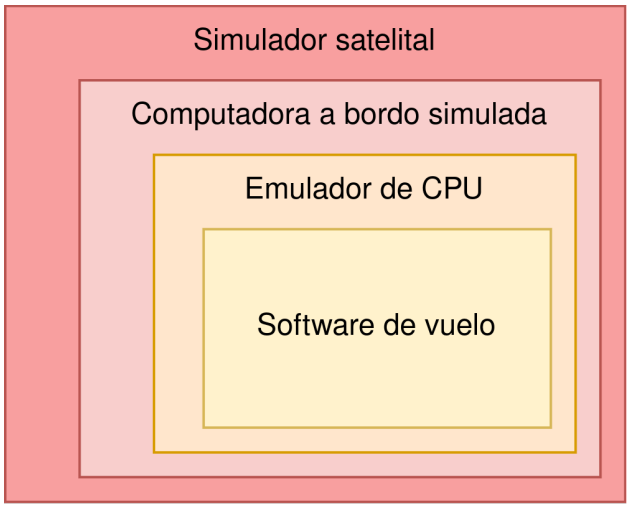
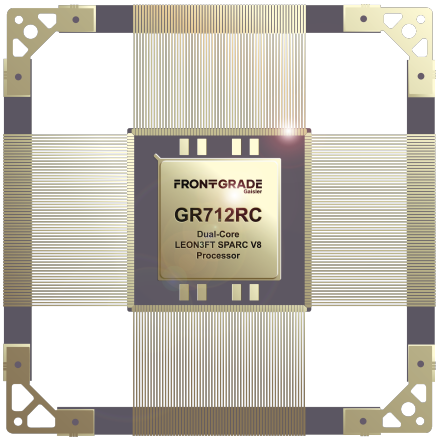


FIGURA 1.1. Relación entre simuladores, emuladores y software de vuelo.

En el caso de los satélites, la OBC suele estar compuesta por un microprocesador, memoria y periféricos, los últimos suelen variar dependiendo de la misión. El microprocesador es el encargado de ejecutar el OBSW, y es el componente que se busca emular en este trabajo. En la figura 1.2 se muestran dos microprocesadores de uso espacial, el GR712RC [1] y el UT700 [2]. Ambos utilizan la arquitectura SPARC V8[3], que es la arquitectura que se desarrolló en este trabajo.



(A) Procesador GR712RC.

(B) Procesador UT700.

FIGURA 1.2. Ejemplo de procesadores de uso espacial.[1]

[1]Imágenes tomadas de https://www.gaisler.com/

Cabe aclarar que el desarrollo de un emulador de microprocesador no es trivial, ya que se debe replicar el comportamiento del microprocesador físico ciclo a ciclo, incluyendo sus instrucciones, registros, y comunicación con los periféricos. Además, se debe tener en cuenta que el microprocesador real puede tener características específicas que no se encuentran en otros microprocesadores, lo que puede dificultar la tarea de emulación.

Dado que el presente desarrollo se realiza en el marco de un posgrado, no se buscó replicar todas las funcionalidades del microprocesador, sino que un subconjunto de las mismas. En particular, el desarrollo se enfocó en la emulación de las instrucciones básicas del microprocesador GR712RC, omitiendo múltiples instrucciones y periféricos.

1.2. Estado del arte

En la actualidad, existen dos grandes grupos de emuladores de microprocesadores: los emuladores de microprocesadores de propósito general y los de propósito específico.

Los emuladores de propósito general son aquellos que buscan emular múltiple microprocesadores de manera genérica, es decir, sin tener en cuenta las características específicas de cada microprocesador. Estos emuladores suelen ser muy complejos y suelen tener un rendimiento inferior a los emuladores de propósito específico, ya que su implementación debe ser lo más abarcativa posible. En particular, los emuladores de propósito general suelen ser utilizados en el desarrollo de en el cual rendimiento no es un factor crítico.

Un ejemplo de este tipo de emuladores es QEMU [4], un emulador de código abierto que permite emular múltiples arquitecturas de microprocesadores, incluyendo x86, ARM, SPARC y otros. QEMU es un emulador muy popular y es utilizado en múltiples proyectos de software libre y de código abierto. Teniendo una comunidad activa de desarrolladores y usuarios. Haciendo que sea una opción muy atractiva para su uso.

Por otro lado, los emuladores de propósito específico son aquellos que buscan emular un microprocesador específico. Estos emuladores suelen ser más simples que los emuladores de propósito general, ya que no buscan generalidad, sino que buscan emular un microprocesador específico de la manera más precisa y eficiente posible. Pudiendo utilizar optimizaciones específicas que podrían romper con la compatibilidad con otros microprocesadores.

Una clara ventaja de este tipo de emuladores, además de su rendimiento, es que suelen ser más fáciles de utilizar y de integrar con otros sistemas, porque están diseñados desde un inicio para ser embebidos en otros sistemas. Lo que es de suma importancia en el desarrollo de sistemas espaciales, ya que gran parte del desarrollo suele hacerse dentro de la misma organización.

A continuación, en la Tabla 1.1 se muestra una tabla comparativa entre emuladores de propósito general y de propósito específico.

TABLA 1.1. Comparación entre emuladores de propósito general y de propósito específico.

Emuladores específicos	Emuladores genéricos
Usualmente privativos y pagos	Solidas opciones gratis disponibles
Alto rendimiento	No enfocado en el rendimiento
Alta integrabilidad	Baja integrabilidad
Soporte limitado	Comunidades activas y código abierto

1.3. Objetivos y alcance

En esta sección se presentan los objetivos y alcance del trabajo final. Además, se presenta el alcance del trabajo, en el cual se describen las limitaciones y restricciones del trabajo.

1.3.1. Objetivos

El objetivo de este trabajo es el diseño e implementación de un emulador de microprocesador Leon3 desarrollado para la empresa INVAP. SE, pensado para su uso en simulaciones espaciales. El trabajo abarca las etapas de diseño, planificación, desarrollo y documentación, con el objetivo de ofrecer una herramienta robusta que se adapte a las necesidades de la industria espacial.

1.3.2. Alcance

El alcance del trabajo se limita a la emulación de un subconjunto de instrucciones del microprocesador Leon3, omitiendo múltiples instrucciones y periféricos. El proyecto no busca ser una herramienta finalizada, sino que busca ser una base sólida para futuros desarrollos. Dadas estas condiciones, el alcance del trabajo se limita a los siguientes puntos:

- Desarrollo del software de emulación.
- Desarrollo de operaciones esperables de un emulador de microprocesador. Tales como la carga de binarios en RAM para su ejecución, la ejecución de instrucciones, y la interacción con periféricos.
- Desarrollo de modelo de memoria RAM.
- Creación de API para la interacción con el emulador.
- Desarrollo de mecanismo de depuración de la herramienta.
- Manual de usuario de la herramienta.

Capítulo 2

Introducción específica

El presente capítulo describe los elementos componentes del sistema, el principio de funcionamiento y los requerimientos acordados con el cliente sobre el sistema.

2.1. Elementos componentes del sistema

Para entender el sistema que se busca desarrollar, es necesario describir los elementos que lo componen. Es muy importante saber que rol cumple un *firmware* y un *bootloader* en un sistema embebido, ya que son los encargados de iniciar el sistema y cargar el software de vuelo en la memoria RAM.

El *firmware* es un software que se encuentra almacenado en la memoria no volátil del sistema, y es el encargado de inicializar los periféricos del sistema. Usualmente, este componente es desarrollado por un equipo especializado tanto en la misión para la que se está desarrollando el sistema, como en la arquitectura del microprocesador que se está utilizando. En el caso de aplicaciones espaciales, el *firmware* es llamado software de vuelo o *Fly Software* (FSW).

Por otro lado, el *bootloader* es un software que se encarga de cargar el software de vuelo en la memoria RAM del sistema. Una vez cargado el software de vuelo, el *bootloader* le cede el control al software de vuelo, el cual se encarga de ejecutar la misión del sistema.

En el presente trabajo, no se ha modelado una memoria persistente, por lo que se ha agregado una función que recibe un archivo binario (FSW) y lo carga en la memoria RAM del sistema. Dicha funciona posicionará al binario en la dirección correcta de memoria, donde el CPU lo pueda ejecutar.

La siguiente figura muestra un diagrama de flujo del procedimiento de carga y ejecución de un software en el emulador desarrollado. El diagrama asume que ya se posee un archivo binario valido para la arquitectura SPARC V8.

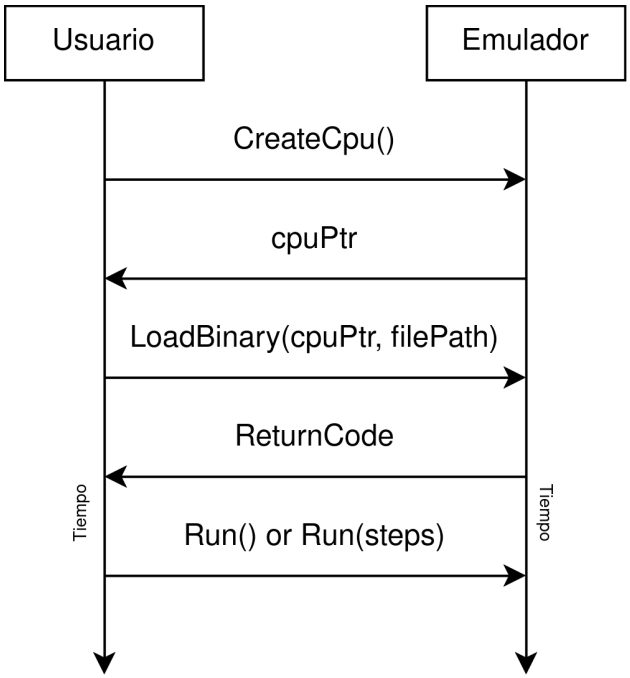


FIGURA 2.1. Proceso de carga de binarios en emulador.

Todo el procedimiento descrito anteriormente se lleva a cabo en la computadora a bordo (OBC) del sistema. La OBC es el componente que se encarga de orquestar el resto de los subsistemas del sistema. En un escenario real, la OBC tendría más componentes que los que se han modelado en este trabajo, tales como periféricos de comunicación UART, CAN y SpaceWire, entre otros. La figura 2.2, tomada de la página de Gaisler [1] y editada, muestra en detalle todos los componentes de la OBC GR712RC resaltando en rojo los componentes que se han modelado en el presente trabajo trabajo.

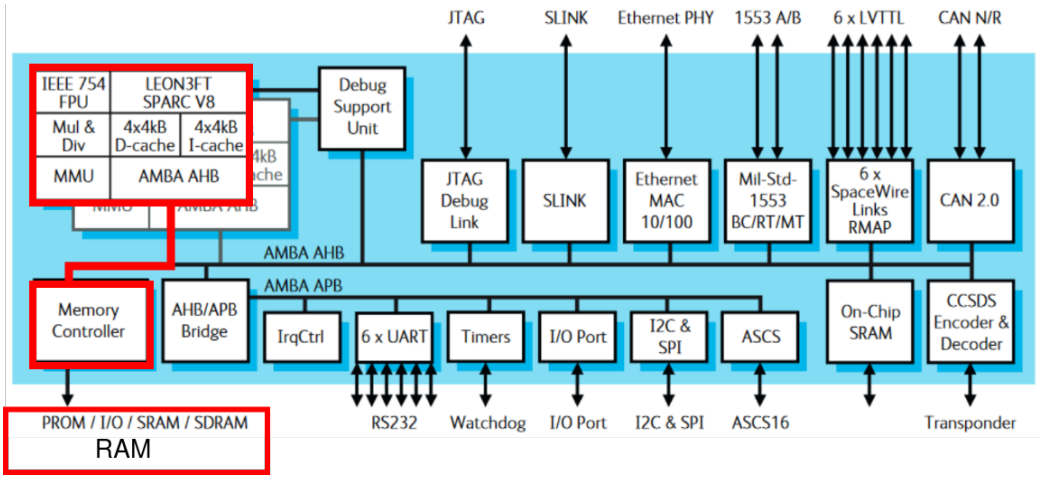


FIGURA 2.2. Proceso de carga de binarios en emulador.

Cabe destacar que se utilizó la placa de desarrollo GR712RC como referencia para el desarrollo del emulador, ya que es una placa de desarrollo de uso común en aplicaciones espaciales. Y podemos destacar las siguientes simplificaciones realizadas:

- Se ha modelado únicamente un CPU. Esta simplificación permite omitir todas las instrucciones de sincronización y manejo de interrupciones que se deberían implementar en un sistema multi-core.
- Se ha modelado el controlador de memoria (MCU por sus siglas en inglés *Memory Controller Unit*) correctamente.
- Las memorias PROM, I/O, SRAM y SDRAM se ha modelado como un único bloque de memoria RAM. Dicha simplificación reduce significativamente el número de modelos a implementar. Cabe aclarar que la presente simplificación no afecta el funcionamiento del emulador.

2.2. Principio de funcionamiento

El funcionamiento de un CPU, y por lo tanto de un emulador, se puede mostrar de manera simplificada de forma gráfica:

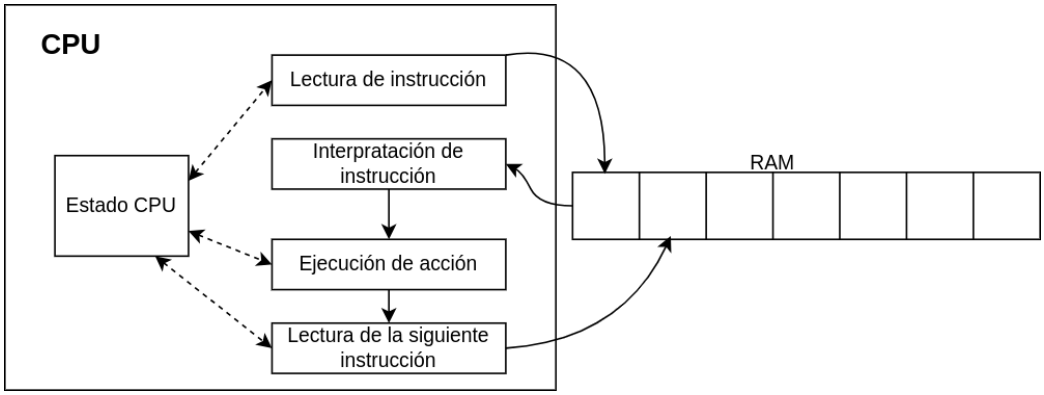


FIGURA 2.3. Funcionamiento de alto nivel de un CPU o emulador.

Dicho flujo se ejecuta de manera continua, y no necesariamente las instrucciones se ejecutan en el orden en el que se encuentran en la memoria. Esto se debe a que el CPU tiene instrucciones de salto condicional, instrucciones de salto incondicional, y puede recibir interrupciones que cambien el flujo de ejecución del programa.

Notas al revisor: *Hasta acá logré escribir, a esta sección le falta un poco de desarrollo. (Un párrafo o dos + explicación de endianess).*

Relacionado a la corrección previa, se pidió una referencia para los tipos de emulador (Yo dije que existen dos grandes grupos) pero no encontré una referencia válida. Espero consejo o guía en como accionar respecto a eso.

Desde ya muchas gracias por su tiempo!

2.3. Requerimientos

Capítulo 3

Diseño e implementación

Capítulo 4

Ensayos y resultados

4.1. Ambiente de pruebas

4.2. Pruebas unitarias

4.3. Pruebas funcionales

4.4. Pruebas de integración

4.5. Caso de uso

Capítulo 5

Conclusiones

5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.

Bibliografía

[1] FrontGrade. *Microprocesador GR712RC*. https://www.gaisler.com/index.php/products/components/gr712rc. Ene. de 2023. (Visitado 13-09-2024).

[2] FrontGrade. *Microprocesador UT700*. https://www.frontgrade.com/product/ut700. Ene. de 2024. (Visitado 13-09-2024).

[3] SPARC. *Documentos técnicos*. https://sparc.org/technical-documents/. Sep. de 2024. (Visitado 13-09-2024).

[4] QEMU. *Sitio web oficial de QEMU*. https://www.qemu.org/. Ene. de 2024. (Visitado 13-09-2024).