Imagen que contiene Icono

Descripción generada automáticamenteLogotipo

Descripción generada automáticamente con confianza media

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Arquitectura de Computadoras

**“Simulador RARS”**

Alumno:

Malagón Baeza Alan Adrian

Profesor:

Alemán Arce Miguel Ángel

Grupo: 5CV1

**Introducción**

RARS es un simulador de arquitectura de computadora educativo y gratuito, desarrollado por Randy Bryant y Dave O'Hallaron de la Universidad Carnegie Mellon. El nombre RARS es una abreviatura de "Randy and Dave's ARSimulator".

RARS está diseñado para ayudar a los estudiantes a comprender los conceptos básicos de la arquitectura de computadoras, incluyendo la programación en lenguaje ensamblador y la simulación de programas. Es un programa de código abierto que puede ser descargado e instalado en una variedad de sistemas operativos, incluyendo Windows, macOS y Linux.

RARS es una herramienta útil para la enseñanza y el aprendizaje de la arquitectura de computadoras, y ha sido utilizado en muchos cursos de nivel universitario en todo el mundo. También cuenta con una amplia documentación y tutoriales para ayudar a los usuarios a empezar a trabajar con el simulador.

En resumen, RARS es un simulador de arquitectura de computadora educativo y gratuito, diseñado para ayudar a los estudiantes a comprender los conceptos básicos de la arquitectura de computadoras mediante la simulación de programas y la programación en lenguaje ensamblador.



**Desarrollo**

**Pantalla del RARS**

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

**Ensamblando el programa**

**Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente**

**Ejecutando el programa**

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente con confianza media

**Tarea 1**

Todavía sabemos muy poco del RISC-V. No sabemos programar ni entendemos las instrucciones... Ejecuta de nuevo el programa hola mundo, paso a paso, y trata de responder las siguientes preguntas:

* ¿Qué instrucción es la que hace que aparezca el mensaje de texto en la consola?  
    
  La instrucción que hace que aparezca el mensaje de texto en la consola es la siguiente:  
    
  Aplicación

  Descripción generada automáticamente con confianza media  
    
  En esta instrucción, el registro a0 se carga con la dirección de la cadena de caracteres "Hola mundo en RISC-V!\n" (que se encuentra en la sección .data), el registro a7 se carga con el valor 4 (que indica que se debe imprimir una cadena de caracteres) y luego se hace una llamada al sistema (ecall) para imprimir el mensaje en la consola.
* ¿Qué instrucción es la que hace que el programa termine?  
    
  La instrucción que hace que el programa termine es la siguiente:  
    
    
    
  En esta instrucción, el registro a7 se carga con el valor 10 (que indica que se debe salir del programa) y se hace una llamada al sistema (ecall) para terminar la ejecución del programa.
* ¿Sabes cuál es el código máquina de esta instrucción?  
    
  El código máquina de la instrucción li (load immediate) es 0010011 en binario.
* ¿Cuántos bits tiene esta instrucción?  
    
  Esta instrucción tiene 32 bits (4 bytes).
* ¿Cuántos instrucciones se ejecuten hasta que el programa termina?  
    
  En total se ejecutan dos instrucciones antes de que el programa termine.
* ¿Cuál crees que es la dirección donde está situada la primera instrucción?  
    
  La primera instrucción del programa está en la dirección 0x0, ya que no se especifica otra dirección para el segmento de código en el archivo de ensamblador.

**Nuestro primer programa**

**Ensamblando el programa**

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media**

**Ejecutando el programa**

**Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente**

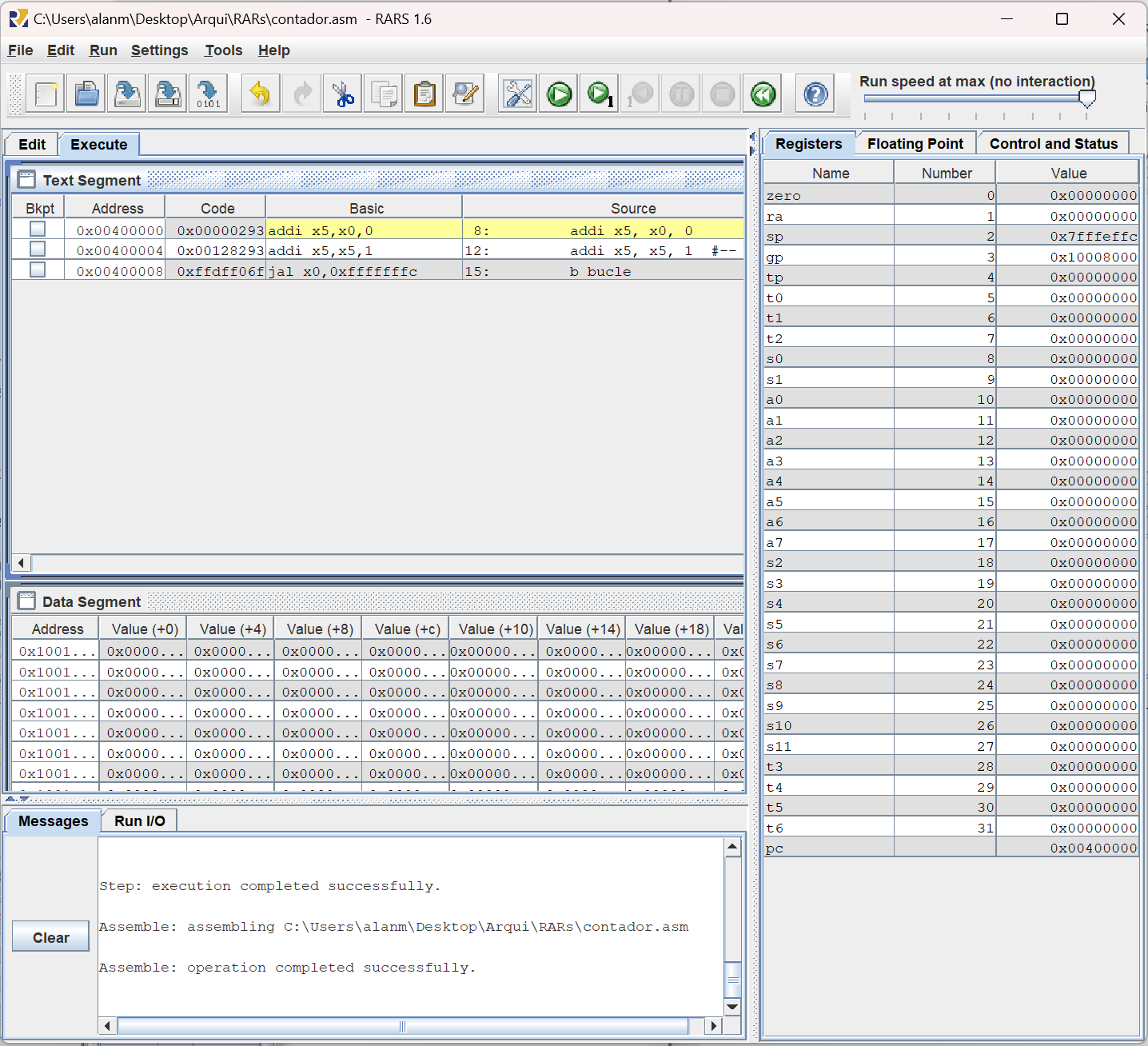
**Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media**

Comprobamos que efectivamente al ejecutar la instrucción **addi** el registro x3 cambia su valor a 30 (0x1e en hexadecimal)

**Un programa contador**

**Ensamblando el programa**

****

**Ejecutando el programa**

**Tabla

Descripción generada automáticamente**

Observamos cómo el registro 5 se incrementa cada vez.

**Tareas**

Estas actividades están pensadas para que las **hagas por tu cuenta**, sin guía del profesor. Queremos que **pienses**. Habrá muchas cosas que no sepas, o que no entiendas: Investiga, lee, prueba, practica

**Recuerda**: El objetivo es que aprendas.

**Ejercicio 1**

**Escribe** un programa para RISC-V que asigne los siguientes valores a los registros indicados: x3=3, x4=4, x5=5, x6=6, x7=7 y x8=8. **Ejecútalo paso a paso** y comprueba que funciona correctamente

**Código del programa para RISC-V**

El siguiente programa en RISC-V asignará los valores indicados a los registros correspondientes:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Ensamblando del programa**

Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**

Tabla

Descripción generada automáticamente

En este programa, se utiliza la instrucción li (load immediate) para cargar los valores 3, 4, 5, 6, 7 y 8 en los registros x3, x4, x5, x6, x7 y x8, respectivamente. Luego, se utiliza la instrucción li junto con los valores 10 y a7 para indicar que el programa debe terminar, y se hace una llamada al sistema (ecall) para salir del programa.

Como podemos observar funciona correctamente.

**Ejercicio 2**

**Modifica** el programa del contador de las actividades guiadas para que se incremente el registro x5 de dos en dos. Ejecútalo paso a paso para comprobar que funciona bien. ¿Cuál es la dirección de la primera instrucción?

**Código del programa**

Para modificar el programa del contador y que incremente el registro x5 de dos en dos, podemos cambiar la instrucción addi x5, x5, 1 por addi x5, x5, 2. De esta forma, el valor del registro x5 se incrementará en dos unidades en cada iteración del bucle.

El programa modificado quedaría así:

Texto

Descripción generada automáticamente

**Ensamblando el programa**

Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**

Tabla

Descripción generada automáticamente

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

Como podemos ver funciona correctamente incrementando de 10 a 12.

En cuanto a la dirección de la primera instrucción, si no se especifica ninguna dirección en el archivo de ensamblador, la primera instrucción se ubicará en la dirección 0x0.

**Ejercicio 3**

**Escribe** un programa para el RISC-V para que el registro x3 tome los valores 0,1,2,3,4,5..., el x4 0,3,6,9,12,15... y el x5 0,5,10,15,20,25.... indefinidamente. Ejecútalo paso a paso para comprobarlo

**Código del programa**

Este programa inicializa los registros x3, x4 y x5 en 0 y después los incrementa de forma independiente en 1, 3 y 5 unidades respectivamente en cada iteración del bucle. Como se trata de un bucle infinito, los registros seguirán incrementándose indefinidamente.

Texto

Descripción generada automáticamente

**Ensamblando el programa**

Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**

Tabla

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Como podemos observar funciona correctamente en la segunda iteración obtienen los vales 1,3 y 5 respectivamente y en la tercera iteración 2,6 y 10 que son los valores esperados.

**Ejercicio 4**

**Escribe** un programa que ejecute las siguientes acciones:

* Inicializar el registro 6 a 0
* Restarle 1
* Terminar ¿Cuál es el valor hexadecimal del registro 6 al terminar el programa?

**Código del programa**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Ensamblando el programa**

Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**

Inicializa x6 en 0

Tabla

Descripción generada automáticamente

Resta 1

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

Termina

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

El valor hexadecimal del registro 6 al terminar el programa será 0xFFFFFFFFFFFFFFF, debido a que se inicializa en cero y luego se le resta 1, lo que resulta en un valor en complemento a dos que tiene todos los bits en 1. Este valor se corresponde con el número decimal -1 en un sistema de 64 bits, que es el tamaño de los registros en RISC-V.

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

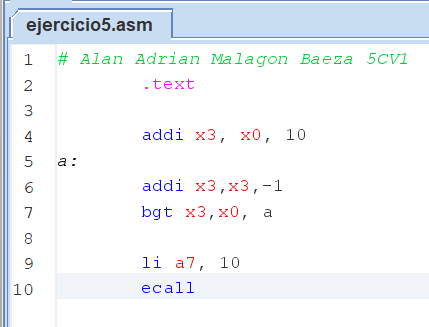
**Ejercicio 5**

**Ejecuta** este código paso a paso. ¿Qué es lo que hace?

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Código del programa**



**Ensamblando el programa**

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**Tabla

Descripción generada automáticamenteTabla

Descripción generada automáticamente

Este programa en RISC-V realiza un bucle que decrementa el registro x3 en una unidad en cada iteración, hasta que el registro x3 alcanza el valor cero. Luego, el programa finaliza con la instrucción li a7, 10 seguida de ecall, que termina la ejecución del programa.

El bucle se implementa con la etiqueta a, que se coloca en la instrucción inmediatamente después de la inicialización del registro x3 en 10. La instrucción bgt x3,x0, a salta a la etiqueta a si el registro x3 es mayor que cero. En cada iteración del bucle, se decrementa el registro x3 mediante la instrucción addi x3,x3,-1, y luego se evalúa si es mayor que cero para decidir si se continúa con la siguiente iteración o si se sale del bucle.

En resumen, este programa realiza un bucle que decrementa el registro x3 desde 10 hasta 0, y luego termina la ejecución del programa.

**Ejercicio 6**

**Ejecuta** este código paso a paso. ¿Qué es lo que hace?

Texto

Descripción generada automáticamente

**Código del programa**

Tabla

Descripción generada automáticamente con confianza baja

**Ensamblando el programa**

Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**

Tabla

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Este programa en RISC-V realiza un bucle que suma el valor del registro x6 en el registro x7 x5 veces. Luego, termina la ejecución del programa.

En la primera sección, se inicializan los registros x5 y x6 con los valores 5 y 6 respectivamente, y el registro x7 se inicializa en cero.

Luego, se implementa el bucle con la etiqueta a. La instrucción beq x5,x0,fin salta a la etiqueta fin si el registro x5 es igual a cero, lo que indica que se han realizado las sumas x5 veces. Si x5 es distinto de cero, se ejecutan las siguientes dos instrucciones:

add x7, x7, x6 suma el valor del registro x6 al registro x7.

addi x5, x5, -1 decrementa el registro x5 en una unidad.

Luego, se salta a la etiqueta a para continuar con la siguiente iteración del bucle.

Finalmente, cuando se han realizado las sumas x5 veces y x5 es igual a cero, se ejecuta la instrucción li a7, 10 seguida de ecall para terminar la ejecución del programa.

En resumen, este programa realiza un bucle que suma el valor del registro x6 en el registro x7 x5 veces, y luego termina la ejecución del programa.

**Ejercicio 7**

El siguiente código **NO** se ensambla correctamente porque tiene **errores**. Introdúcelo en el simulador y soluciona los programas para que ensamble correctamente y se pueda ejecutar paso a paso. ¿Qué hace?

Texto

Descripción generada automáticamente

**Código del programa con errores**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Ensamblando el programa con errores**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Código del programa sin errores**

Texto

Descripción generada automáticamente

**Ensamblando el programa sin errores**

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

**Ejecutando paso a paso**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

Este código es un bucle que se ejecuta 10 veces. En cada iteración, se suma el contenido del registro x12 al registro x7, y luego se decrementa el registro x17 en una unidad. Cuando el registro x17 llega a cero, el programa salta a la etiqueta "fin", donde se realiza una llamada al sistema para terminar el programa.

El bucle en este código está implementado usando la instrucción "jal ra, a", que es un salto incondicional a la etiqueta "a" y al mismo tiempo guarda la dirección de retorno en el registro ra. Al finalizar la ejecución del código en la etiqueta "a", la instrucción "jalr ra, x0, 0" es ejecutada, la cual carga la dirección de retorno desde el registro ra y salta a esa dirección. De esta manera, el programa se repite indefinidamente hasta que se cumpla la condición de terminación.

**Conclusión**

Se realizaron varios programas de RISC-V que realizan diferentes operaciones, como la inicialización de registros, la asignación de valores, la suma y resta de registros y la repetición de bucles. Estos programas fueron escritos en lenguaje ensamblador de RISC-V y se centraron en el uso de instrucciones básicas, como addi, add, sub, beq, bne, j y jal.

En general, estos programas son útiles para entender cómo funciona la arquitectura de RISC-V y cómo se pueden escribir programas simples en lenguaje ensamblador para esta arquitectura. Además, es importante destacar que el conocimiento de lenguaje ensamblador de RISC-V es útil para el desarrollo de software de bajo nivel y para la optimización de código.

En conclusión, los programas de RISC-V realizados son una introducción útil a la programación de bajo nivel y a la arquitectura de RISC-V, y pueden ser un buen punto de partida para aquellos interesados en aprender más sobre el tema.

**Referencias**

* myTeachingURJC. (2020). 2019\_20 LAB AO/wiki/L1: Practica 1. GitHub. https://github.com/myTeachingURJC/2019\_20\_LAB\_AO/wiki/L1:\_Practica\_1
* TheThirdOne. (2017, July 16). Rars [Computer software]. GitHub. https://github.com/TheThirdOne/rars
* Raul, J. (2015, November 9). ¿Qué es RISC y CISC? ¿Como funciona? [Video]. YouTube. https://www.youtube.com/watch?v=GYvBAHdkRwk&t=298s