



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

ARQUITECTURAS GRÁFICAS

Grado en Diseño y Desarrollo de Videojuegos

Actividades prácticas

Programación en ensamblador de MIPS:
modos de direccionamiento y llamadas a sistema

CURSO 2017-18

Luis Rincón Córcoles

Área de Arquitectura y Tecnología de Computadores

Introducción

En el presente cuaderno de actividades prácticas se analizarán y simularán diversos programas en ensamblador de MIPS, utilizando para ello la herramienta **MARS**, realizada por Peter Sanderson y Kenneth Vollmar de la Missouri State University. Será preciso descargar la última versión del mismo (que actualmente es la 4.5). Puesto que **MARS** está escrito en Java, para poder ejecutarlo será preciso tener instalada una versión reciente del Java Runtime Environment. Los programas que deben simularse se proporcionarán como ficheros anexos al enunciado.

Las actividades realizadas serán las siguientes:

- a) Un test objetivo a través de la herramienta de Aula Virtual, cuyas preguntas se desgranarán a lo largo de los ejercicios del 1 al 6. Para realizarlo, será preciso realizar varias simulaciones con **MARS**. En este documento se indican los pasos necesarios, así como los enunciados de las preguntas y las respuestas alternativas. Es preciso tener en cuenta que en ciertas ocasiones se requerirá introducir ciertos valores en los elementos de **MARS**. En este documento se indica cómo hacerlo. En ocasiones, los valores concretos que deben ser introducidos se indican en el test de Aula Virtual, no en este documento.
- b) Una traza de la ejecución de un programa sencillo que realiza una operación de interpolación lineal.

AMBAS ACTIVIDADES SE REALIZARÁN DE FORMA INDIVIDUAL.

Trabajo preparatorio

Antes de acudir a la sesión de laboratorio, es preciso realizar un trabajo de preparación previo.

En primer lugar, es imprescindible estudiar los contenidos teóricos de los apartados 6, 7 y 8 del tema 3. Esta actividad consiste en la simulación de los ejemplos que aparecen en dichos apartados, con objeto de afianzar los conceptos explicados.

En segundo lugar, es altamente recomendable utilizar el simulador **MARS** y analizar su funcionamiento antes de acudir a la sesión de prácticas. Esto permitirá aprovechar mejor el tiempo, y detectar con antelación puntos concretos en los que puede surgir alguna duda.

Antes de comenzar


El material necesario para realizar las actividades de esta sesión se encuentra en el archivo comprimido **AG-actividad0301-fuentes.zip**. Este archivo deberá ser copiado en una carpeta de un medio de almacenamiento, junto con el archivo **MARS.jar**. A continuación se deberá descomprimir el archivo ZIP, con lo que aparecerán los siguientes ficheros:

- **suma1.asm**, **suma2.asm**, **suma3.asm**, **suma4.asm**, **suma5.asm** y **suma6.asm**, necesarios para realizar la parte 1.
- **interpolacion_lineal.asm**, que es el archivo fuente para la realización de la parte 2.
- **traza.xlsx**, en el que se realizará la traza pedida en la parte 2.

Parte 1: realización de un test objetivo

Para cumplimentar las preguntas de los ejercicios correspondientes a esta actividad, será preciso entrar en el Aula Virtual y arrancar el test objetivo asociado.

Ejercicio 1

Arranca el simulador MARS y carga el archivo *suma1.asm* mediante las opciones *File* ▷ *Open* y buscando el archivo correspondiente. A continuación, ensambla el código con la opción *Run* ▷ *Assemble*, o bien pulsando F3 o pinchando el botón .

1. ¿Cuál es el mensaje que aparece en la zona de mensajes de MARS (*Mars Messages*) al ensamblar el código fuente del archivo *suma1.asm*? (los puntos suspensivos representan la ruta completa hasta el archivo)

- a) Assemble: assembling C:\...\suma1.asm
Assemble: operation completed successfully.
- b)** Assemble: assembling C:\...\suma1.asm
Error in C:\...\suma1.asm line 7 column 6: "A": operand is of incorrect type
Assemble: operation completed with errors.
- c) – program is finished running –
- d) Ninguna de las restantes respuestas es cierta.

2. ¿Por qué sale dicho mensaje?

- a) Porque el ensamblaje y la ejecución son correctos.
- b)** Porque el código de operación es incorrecto.
- c) Porque hay un error en el direccionamiento de A.
- d) Ninguna de las restantes respuestas es cierta.

3. ¿Como arreglarías el error producido?

- a) Cambiando el código de operación de la instrucción.
- b) Cambiando el direccionamiento de A, y así la instrucción sería correcta.
- c)** Cambiando el direccionamiento de A, B y C, ya que cambiar sólo A no es suficiente para que la instrucción esté correcta.
- d) Ninguna de las restantes respuestas es cierta.

Ejercicio 2

Pulsa el botón *Clear* de la zona de mensajes. Ahora carga el archivo *suma2.asm* y ensámblalo.


4. ¿Cuál es el mensaje que aparece en la zona de mensajes de MARS (*Mars Messages*) al ensamblar el código fuente del archivo *suma2.asm*? (los puntos suspensivos representan la ruta completa hasta el archivo)

- a)** Assemble: assembling C:\...\suma2.asm
Assemble: operation completed successfully.
- b) Assemble: assembling C:\...\suma2.asm

Error in C:\...\suma1.asm line 7 column 6: "\$s1": operand is of incorrect type

Assemble: operation completed with errors.

- c) – program is finished running –
- d) Ninguna de las restantes respuestas es cierta.


Al ensamblar el programa se activa automáticamente la pestaña de ejecución. Ahora ejecuta el programa mediante la opción *Run* ▷ *Go*, o bien pulsando F5 o pinchando el botón .

5. ¿Cuál es el mensaje que aparece en la zona de mensajes *Run I/O* de MARS al ejecutar el código del archivo suma2.asm?

- a) – program is finished running (dropped off bottom) –
- b) Assemble: assembling C:\...\suma2.asm
– program is finished running –
- c) – program is finished running –
- d) Ninguna de las restantes respuestas es cierta.

6. ¿Por qué aparece dicho mensaje?

- a) Porque la ejecución ha terminado completamente con una llamada a la rutina de sistema para terminación de programa.
- b) Porque la ejecución ha sido incorrecta.
- c) Porque no hay instrucciones en el programa.
- d) Porque el contador de programa ha superado la última instrucción del programa.

Vamos a simular de nuevo el programa. Para ello, reinicia la simulación mediante la opción *Run* ▷ *Reset*, o bien pulsando F12 o pinchando el botón . Ahora modifica el contenido de los registros que actúan como operandos fuente. Para ello, haz *click* sobre el contenido del registro \$s2, escribe el valor indicado en la pregunta 7 del test del Aula Virtual y pulsa INTRO. Haz lo mismo con \$s3. Asegúrate de que la opción *Hexadecimal Values* de la zona *Data Segment* está desmarcada y de que aparecen los contenidos de ambos registros expresados en base 10 (es decir, que no están precedidos de “0x”).

7. Indica cuál es el valor final contenido en \$s1 tras ejecutar de nuevo el programa, introduciendo en \$s2 y en \$s3 los valores que se indican en el enunciado de la pregunta de test del Aula Virtual, expresados ambos en base 10.

Modifica de nuevo el contenido de los registros \$s2 y \$s3. Coloca sobre ambos el valor hexadecimal 0x7FFFFFFF. Para ello, si es preciso marca la opción *Hexadecimal Values* de la zona *Data Segment*.

8. ¿Cuál es el mensaje que aparece en la zona *Mars Messages* tras ejecutar de nuevo el código con los valores antedichos? (los puntos suspensivos representan la ruta completa hasta el archivo fuente)

- a) – program is finished running (dropped off bottom) –
- b) – program is finished running –

- c)** Error in C:\...\suma02.asm line 7: Runtime exception at 0x00400000: arithmetic overflow
Go: execution terminated with errors.
- d) Ninguna de las restantes respuestas es cierta.

9. ¿Qué significa dicho mensaje?

- a) Que el programa ha terminado correctamente.
- b)** Que el programa ha terminado con errores, puesto que los datos que hemos sumado producen un desbordamiento aritmético, ya que el resultado no se puede representar con 32 bits.
- c) Que se ha producido un error en la ejecución debido a que los operandos eran de tipo incorrecto por haberlos escrito en hexadecimal.
- d) Ninguna de las restantes respuestas es cierta.

Ejercicio 3

Pulsa el botón *Clear* de la zona de mensajes. Ahora carga el archivo *suma3.asm* y ensámblalo. Modifica el contenido de **\$s2** según lo indicado en la pregunta 10 del test del Aula Virtual y ejecuta el programa.

10. Indica cuál es el valor final contenido en \$s1 tras ejecutar el programa suma3.asm, habiendo introducido previamente en \$s2 el valor indicado en el enunciado de la pregunta de test del Aula Virtual (expresado en base 10).

11. ¿Se puede colocar un operando inmediato negativo en la instrucción addi?

- a)** Sí.
- b) No, porque sería una resta y no una suma.
- c) No, porque el operando tiene que ser siempre positivo.
- d) Ninguna de las restantes respuestas es cierta.

Ejercicio 4

Pulsa el botón *Clear* de la zona de mensajes. Ahora carga el archivo *suma4.asm* y ensámblalo. Modifica el contenido de **\$s2** y de **\$s3** tal como indica la pregunta 12 del test del Aula Virtual y ejecuta el programa.

12. Indica cuál es el valor final contenido en \$s1, \$s2 y \$s3 tras ejecutar el programa, habiendo introducido previamente en \$s2 y \$s3 los valores indicados en el enunciado de la pregunta 12 de test del Aula Virtual (expresados en base 10)

13. ¿Por qué ha salido este resultado?

- a) Porque me he equivocado al meter los valores en los registros.
- b) Porque al cargar B y C sobre los registros se han machacado los valores que había puesto antes.

- c) Porque la carga no funciona bien.
- d) Ninguna de las restantes respuestas es cierta.

14. ¿Cuáles son las direcciones en memoria de las variables A, B y C, expresadas en hexadecimal?

- a) Dirección(A) = 0x10010000; Dirección(B) = 0x10010004; Dirección(C) = 0x10010008
- b) Dirección(A) = 0x10010000; Dirección(B) = 0x10010001; Dirección(C) = 0x10010002
- c) Dirección(A) = 0x10010000; Dirección(B) = 0x10010002; Dirección(C) = 0x10010004
- d) Ninguna de las restantes respuestas es cierta.

Reinicia la simulación, modifica los valores de las variables A, B y C tal como indica la pregunta 15 del test del Aula Virtual y ejecuta de nuevo el programa.

15. ¿Cuál es el contenido final en memoria de la variable A tras ejecutar de nuevo el programa, habiendo introducido previamente en las variables B y C los valores indicados en el enunciado de la pregunta 15 de test del Aula Virtual (expresados en base 10)?

Ejercicio 5

Pulsa el botón *Clear* de la zona de mensajes. Ahora carga el archivo *suma5.asm* y ensámblalo. Modifica el contenido de las variables A, B y C tal como dice la pregunta 16 del test del Aula Virtual y ejecuta el programa.

16. Indica cuál es el valor final contenido en A tras ejecutar el programa, habiendo introducido previamente en las variables A, B y C los valores indicados en el enunciado de la pregunta 16 de test del Camps Virtual (expresados en base 10).

17. ¿El programa ha entrado por la rama then o por la rama else?

- a) Por la rama then.
- b) Por la rama else.
- c) Por las dos ramas, primero por la rama else y después por la rama then.
- d) Ninguna de las restantes respuestas es cierta.

Reinicia la simulación, modifica los valores de las variables A, B y C tal como dice la pregunta 18 del test del Aula Virtual y ejecuta el programa.

18. Indica cuál es el valor final contenido en A tras ejecutar el programa, habiendo introducido previamente en las variables A, B y C los valores indicados en el enunciado de la pregunta 18 de test del Camps Virtual (expresados en base 10).

19. ¿El programa ha entrado por la rama then o por la rama else?

- a) Por la rama then.
- b) Por la rama else.

- c) Por las dos ramas, primero por la rama else y después por la rama then.
- d) Ninguna de las restantes respuestas es cierta.

Ejercicio 6

Pulsa el botón *Clear* de la zona de mensajes. Ahora carga el archivo *suma6.asm* y ensámblalo. Ejecuta el programa, e introduce en las variables B y C los valores indicados en la pregunta 20 del test.

20. Indica cuál es el valor final contenido en A tras ejecutar el programa, habiendo introducido los valores indicados en el enunciado de la pregunta 20 de test del Aula Virtual en B y C cuando el programa lo ha solicitado.

21. ¿Cuál es la dirección de comienzo de la tira de caracteres tiraB?

- a) 0x10010000
- b) 0x10010004
- c) 0x10010008
- d) 0x1001000c

22. ¿Cuál es la longitud en bytes de tiraB? (Pista: mira dónde empieza tiraC y haz una simple resta)

- a) 1 byte.
- b) 14 bytes.
- c) 16 bytes.
- d) Ninguna de las restantes respuestas es cierta.

23. Fíjate en la pseudoinstrucción `li $v0,4` de la línea 17 del código fuente. ¿Cuál es la instrucción ensamblador en la que se traduce?

- a) `addiu $2,$0,0x0004`
- b) `addi $2,$0,0x0004`
- c) `li $2,0x0004`
- d) Ninguna de las restantes respuestas es cierta.

24. Fíjate ahora en la pseudoinstrucción `la $a0,tiraB`. ¿Cuál es su traducción a ensamblador?

- a) `lui $1,0x1001`
`ori $4,$1,0x000c`
- b) `addiu $2,$0,0x000c`
- c) `lui $1,0x1001`
`ori $4,$1,0x001a`
- d) Ninguna de las restantes respuestas es cierta.

25. Si tuviésemos la pseudoinstrucción `li $v0,50000`, ¿se traduciría con la misma instrucción ensamblador que `li $v0,4` sin más que cambiar el inmediato, o la traducción sería diferente?

- a) Sería igual que la traducción de `li $v0,4`, pero poniendo el nuevo valor inmediato en vez del valor 4.
- b) Habría que usar dos instrucciones como mínimo para realizar la traducción.
- c) Se puede hacer con una instrucción `ori`.
- d) Ninguna de las restantes respuestas es cierta.

Una vez respondida esta pregunta, se revisarán las respuestas del test y se efectuará la entrega del mismo.

Parte 2

A continuación se realizará una traza de ejecución de un programa sencillo en ensamblador de MIPS, utilizando para ello la herramienta **MARS**. El programa que se debe simular está en el archivo `interpolacion_lineal.asm`.

Las instrucciones para realizar la traza se describen en el manual de uso de **MARS**, disponible en el curso virtual de la asignatura. Los detalles se encuentran en el apartado 5 y subapartados de dicho manual, particularmente en el 5.1. En el apéndice A figura una descripción del código fuente del programa completo.

La traza se escribirá en el archivo `traza.xlsx`. Este archivo se entregará a través del Aula Virtual de la asignatura.

Adicionalmente, se recomienda:

- a) Realizar más trazas del programa con otros datos.
- b) Analizar detenidamente el efecto producido por todas las instrucciones, haciendo especial hincapié en la instrucción `lui`.
- c) Estudiar cómo se realiza la traducción de las pseudoinstrucciones a instrucciones reales soportadas por el hardware.

Como ayuda para la realización de esta actividad, se recomienda recurrir al manual de usuario de **MARS** que puede encontrarse en el Aula Virtual, en el apartado de Recursos generales. En él se describen las acciones necesarias para generar una traza usando este mismo código como modelo.