



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.

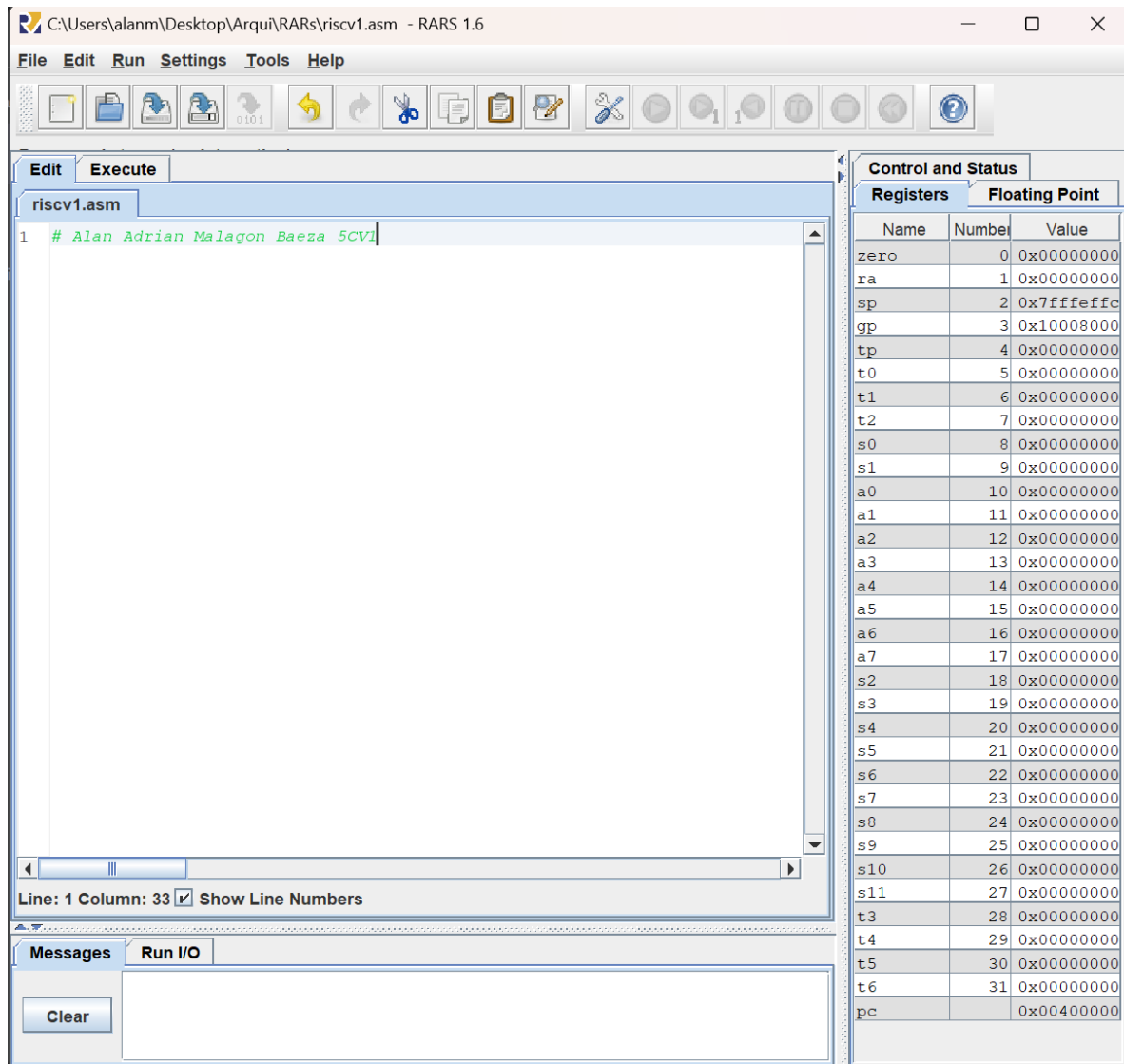
RARS es una herramienta útil para la enseñanza y el aprendizaje de la arquitectura de computadoras, y ha sido utilizado en muchos cursos de nivel universitario en todo el mundo. También cuenta con una amplia documentación y tutoriales para ayudar a los usuarios a empezar a trabajar con el simulador.

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



Ensamblando el programa

C:\Users\alanm\Desktop\Arqui\RARS\hola_mundo.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x0fc10517	auipc x10,0x0000fc10	11: la a0, str
	0x00400004	0x00050513	addi x10,x10,0	
	0x00400008	0x00400893	addi x17,x0,4	12: li a7, 4
	0x0040000c	0x00000073	ecall	13: ecall
	0x00400010	0x00a00893	addi x17,x0,10	15: li a7, 10
	0x00400014	0x00000073	ecall	16: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Val
0x1001...	0x616c...	0x6e75...	0x6520...	0x4952...	0x562d4...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x00000...	0x0000...	0x00000...	0x0

Registers

Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x10010000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x0000000a
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400018

Messages

Run I/O

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\hola_mundo.asm

Assemble: operation completed successfully.

Go: running hola_mundo.asm

Go: execution completed successfully.

Ejecutando el programa

C:\Users\alanm\Desktop\Arqui\RARS\hola_mundo.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x0fc10517	auipc x10,0x0000fc10	11: la a0, str
	0x00400004	0x00050513	addi x10,x10,0	
	0x00400008	0x00400893	addi x17,x0,4	12: li a7, 4
	0x0040000c	0x00000073	ecall	13: ecall
	0x00400010	0x00a00893	addi x17,x0,10	15: li a7, 10
	0x00400014	0x00000073	ecall	16: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Val
0x1001...	0x616c...	0x6e75...	0x6520...	0x4952...	0x562d4...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0

Registers

Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffeff
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x10010000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x0000000a
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400018

Messages **Run I/O**

Hola mundo en RISC-V!

-- program is finished running (0) --

Clear

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:

```
11      la a0, str
12      li a7, 4
13      ecall
```

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:

```
15      li a7, 10
16      ecall
```

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).

- ¿Cuántas 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

C:\Users\alanm\Desktop\Arqui\RARs\primerprograma.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit **Execute**

hola_mundo.asm **primerprograma.asm**

```
1 # Alan Adrian Malagon Baeza 5CV1
2 #-- Nuestro primer programa escrito desde cero
3 #-- Esto que sale en verde son COMENTARIOS
4 #-- y siempre es lo primero que tenemos que hacer
5 #-- ;Somos ingenieros! ;Hay que documentar!
6
7 #-- Programa para asignar el valor 30 al registro x3,
8 #-- y terminar
9
10 #-- Esto NO es una instruccion. Lo veremos mas
11 #-- adelante. Es una Directiva. Se usa para
12 #-- indicar al ensamblador que lo que va a
13 #-- continuacion es codigo
14 .text
15
16 #-- Mi primera instruccion
17 addi x3, x0, 30 #-- x3 = 0 + 30 = 30
18
19 #-- Terminar. Estas instrucciones no saben
20 #-- que significan. Lo veremos mas adelante
21 #-- Las usamos para terminar el programa y
22 #-- devolver el control al sistema operativo
23 li a7, 10
24 ecall
```

Line: 1 Column: 33 ☒ Show Line Numbers

Registers **Floating Point** **Control and Status**

Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400000

Messages **Run I/O**

Step: execution completed successfully.

Clear

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARs\primerprograma.a

Assemble: operation completed successfully.

Ejecutando el programa

C:\Users\alanm\Desktop\Arqui\RARs\primerprograma.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

0 100

Edit Execute

hola_mundo.asm primerprograma.asm

```
1 # Alan Adrian Malagon Baeza 5CV1
2 #-- Nuestro primer programa escrito desde cero
3 #-- Esto que sale en verde son COMENTARIOS
4 #-- y siempre es lo primero que tenemos que hacer
5 #-- ;Somos ingenieros! ;Hay que documentar!
6
7 #-- Programa para asignar el valor 30 al registro x3,
8 #-- y terminar
9
10 #-- Esto NO es una instruccion. Lo veremos mas
11 #-- adelante. Es una Directiva. Se usa para
12 #-- indicar al ensamblador que lo que va a
13 #-- continuacion es codigo
14 .text
15
16 #-- Mi primera instruccion
17 addi x3, x0, 30 #-- x3 = 0 + 30 = 30
18
19 #-- Terminar. Estas instrucciones no saben
20 #-- que significan. Lo veremos mas adelante
21 #-- Las usamos para terminar el programa y
22 #-- devolver el control al sistema operativo
23 li a7, 10
24 ecall
```

Line: 14 Column: 7 ☒ Show Line Numbers

Messages Run I/O

Reset: reset completed.

Reset: reset completed.

Reset: reset completed.

Clear

Registers		
Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffeff
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400000

C:\Users\alanm\Desktop\Arqui\RARS\primerprograma.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit **Execute**

hola_mundo.asm View and control assembly language program execution. Enabled upon successful assemble.

```

1 # Alan Adrian Malagon Baeza 5CV1
2 #-- Nuestro primer programa escrito desde cero
3 #-- Esto que sale en verde son COMENTARIOS
4 #-- y siempre es lo primero que tenemos que hacer
5 #-- ;Somos ingenieros! ;Hay que documentar!
6
7 #-- Programa para asignar el valor 30 al registro x3,
8 #-- y terminar
9
10 #-- Esto NO es una instruccion. Lo veremos mas
11 #-- adelante. Es una Directiva. Se usa para
12 #-- indicar al ensamblador que lo que va a
13 #-- continuacion es codigo
14 .text
15
16 #-- Mi primera instruccion
17 addi x3, x0, 30 #-- x3 = 0 + 30 = 30
18
19 #-- Terminar. Estas instrucciones no saben
20 #-- que significan. Lo veremos mas adelante
21 #-- Las usamos para terminar el programa y
22 #-- devolver el control al sistema operativo
23 li a7, 10
24 ecall

```

Line: 1 Column: 33 ☒ Show Line Numbers

Messages **Run I/O**

Reset: reset completed.

Clear

-- program is finished running (0) --

Registers **Floating Point** **Control and Status**

Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffeff
gp	3	0x0000001e
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x0000000a
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x0040000c

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

C:\Users\alanm\Desktop\Arqui\RARS\contador.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00000293	addi x5,x0,0	8: addi x5, x0, 0
<input type="checkbox"/>	0x00400004	0x00128293	addi x5,x5,1	12: addi x5, x5, 1 #--
<input type="checkbox"/>	0x00400008	0xffdf06f	jal x0,0xffffffffc	15: b bucle

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Val
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0

Registers

Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400000

Messages

Run I/O

Step: execution completed successfully.

Clear

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\contador.asm

Assemble: operation completed successfully.

Ejecutando el programa

The screenshot shows the RARS 1.6 emulator window. The title bar indicates the file path: C:\Users\alanm\Desktop\Arqui\RARS\contador.asm - RARS 1.6. The menu bar includes File, Edit, Run, Settings, Tools, and Help. The toolbar contains various icons for file operations and execution. A 'Run speed at max (no interaction)' slider is visible.

The main window is divided into several panes:

- Text Segment:** Displays assembly code with columns for Bkpt, Address, Code, Basic, and Source. The code includes instructions like `addi x5, x0, 0`, `addi x5, x5, 1`, and `j al x0, -4`.
- Data Segment:** A table showing memory addresses and their values for different data types (Value (+0), Value (+4), Value (+8), Value (+c), Value (+10), Value (+14), Value (+18), Value (+20)).
- Registers:** A table showing the current values of various registers. The 't0' register is highlighted in green, showing a value of 8.
- Messages:** A log of messages, including 'Reset: reset completed.' and a 'Clear' button.

The 'Registers' pane shows the following data:

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	8
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194308

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

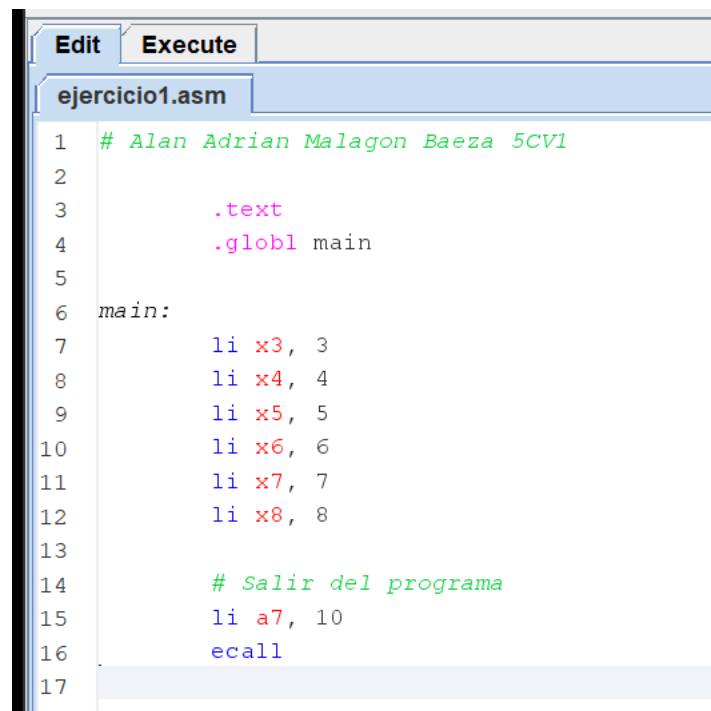
Recuerda: El objetivo es que aprendas.

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:



```
1  # Alan Adrian Malagon Baeza 5CV1
2
3      .text
4      .globl main
5
6  main:
7      li x3, 3
8      li x4, 4
9      li x5, 5
10     li x6, 6
11     li x7, 7
12     li x8, 8
13
14     # Salir del programa
15     li a7, 10
16     ecall
17
```

Ensamblando del programa

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio1.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Assemble the current file and clear breakpoints

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00300193	addi x3,x0,3	7: li x3, 3
	0x00400004	0x00400213	addi x4,x0,4	8: li x4, 4
	0x00400008	0x00500293	addi x5,x0,5	9: li x5, 5
	0x0040000c	0x00600313	addi x6,x0,6	10: li x6, 6
	0x00400010	0x00700393	addi x7,x0,7	11: li x7, 7
	0x00400014	0x00800413	addi x8,x0,8	12: li x8, 8
	0x00400018	0x00a00893	addi x17,x0,10	15: li a7, 10
	0x0040001c	0x00000073	ecall	16: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194304

Messages

Run I/O

Step: execution completed successfully.

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\ejercicio1.asm

Assemble: operation completed successfully.

Clear

Ejecutando paso a paso

The screenshot shows the RARS 1.6 emulator window. The title bar indicates the file path: C:\Users\alanm\Desktop\Arqui\RARS\ejercicio1.asm - RARS 1.6. The menu bar includes File, Edit, Run, Settings, Tools, and Help. The toolbar contains various icons for file operations and execution. The main window is divided into several panes:

- Text Segment:** A table showing assembly instructions. The last instruction, `ecall` at address `0x0040001c`, is highlighted in yellow.
- Data Segment:** A table showing memory addresses and their values, all currently set to 0.
- Registers:** A table showing the state of various registers. The `a7` register is highlighted in green and contains the value 10. The `pc` (program counter) register at the bottom shows the value 4194336.
- Messages / Run I/O:** A text area showing the execution log. It contains the message "Reset: reset completed." and two lines indicating the program is finished running.

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:

```
contador.asm
1  # Alan Adrian Malagon Baeza 5CV1
2  #-- Programa contador
3  #-- El registro x5 se incremeta indefinidamente
4
5      .text
6
7      #-- Inicializar el registro x5 a 0
8      addi x5, x0, 0
9
10     bucle:
11         #-- Incrementar el registro x5 en dos unidades
12         addi x5, x5, 2  #-- x5 = x5 + 2
13
14         #-- Repetir indefinidamente
15         b bucle
```


Ensamblando el programa

C:\Users\alanm\Desktop\Arqui\RARS\contador.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00000293	addi x5,x0,0	8: addi x5, x0, 0
	0x00400004	0x00228293	addi x5,x5,2	12: addi x5, x5, 2 #--
	0x00400008	0xffdff06f	jal x0,-4	15: b bucle

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Val
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194304

Messages

Run I/O

Step: execution completed successfully.

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\contador.asm

Assemble: operation completed successfully.

Clear

Ejecutando paso a paso

C:\Users\alanm\Desktop\Arqui\RARS\contador.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00000293	addi x5,x0,0	8: addi x5, x0, 0
<input type="checkbox"/>	0x00400004	0x00228293	addi x5,x5,2	12: addi x5, x5, 2 #--
<input type="checkbox"/>	0x00400008	0xffdff06f	jal x0,-4	15: b bucle

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Val
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Registers Floating Point Control and Status

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	10
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194312

Messages Run I/O

Reset: reset completed.

Clear

-- program is finished running (0) --

-- program is finished running (0) --

C:\Users\alanm\Desktop\Arqui\RARS\contador.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00000293	addi x5,x0,0	8: addi x5, x0, 0
	0x00400004	0x00228293	addi x5,x5,2	12: addi x5, x5, 2 #--
	0x00400008	0xffdff06f	jal x0,-4	15: b bucle

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Val
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	12
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194312

Messages **Run I/O**

Reset: reset completed.

-- program is finished running (0) --

-- program is finished running (0) --

Clear

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.

```
ejercicio3.asm
1  # Alan Adrian Malagon Baeza 5CV1
2  #-- Programa sumador
3  #-- Los registros x3, x4 y x5 se incrementan de forma independiente
4
5      .text
6
7      #-- Inicializar los registros x3, x4 y x5 a 0
8      addi x3, x0, 0
9      addi x4, x0, 0
10     addi x5, x0, 0
11
12     bucle:
13         #-- Incrementar el registro x3 en una unidad
14         addi x3, x3, 1  #-- x3 = x3 + 1
15
16         #-- Incrementar el registro x4 en tres unidades
17         addi x4, x4, 3  #-- x4 = x4 + 3
18
19         #-- Incrementar el registro x5 en cinco unidades
20         addi x5, x5, 5  #-- x5 = x5 + 5
21
22         #-- Repetir indefinidamente
23         b bucle
24
```

Ensamblando el programa

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio3.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00000193	addi x3,x0,0	8: addi x3, x0, 0
	0x00400004	0x00000213	addi x4,x0,0	9: addi x4, x0, 0
	0x00400008	0x00000293	addi x5,x0,0	10: addi x5, x0, 0
	0x0040000c	0x00118193	addi x3,x3,1	14: addi x3, x3, 1 #-- x3 =..
	0x00400010	0x00320213	addi x4,x4,3	17: addi x4, x4, 3 #-- x4 =..
	0x00400014	0x00528293	addi x5,x5,5	20: addi x5, x5, 5 #-- x5 =..
	0x00400018	0xff5ff06f	jal x0,-12	23: b bucle

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Control and Status

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194304

Messages

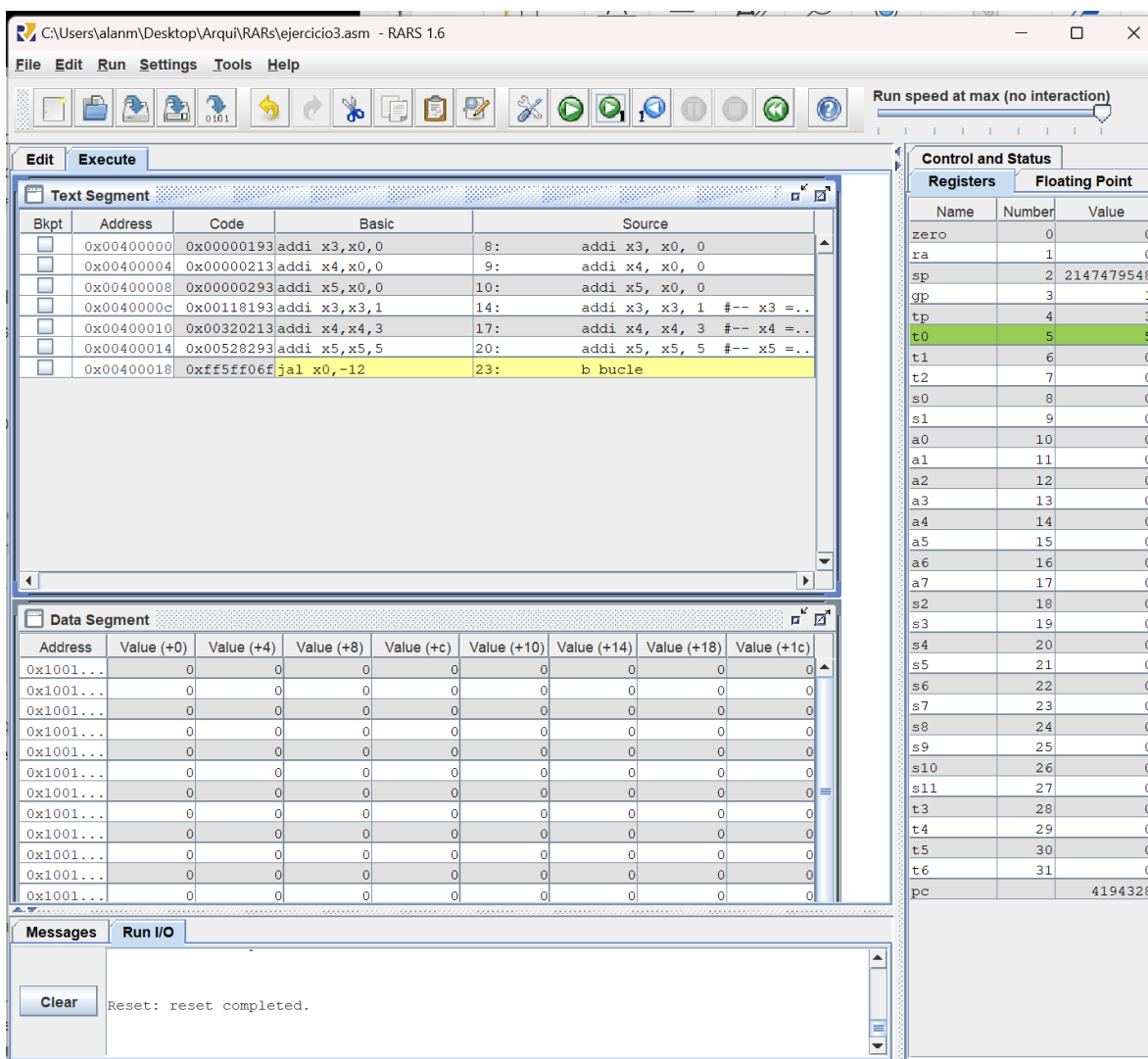
Run I/O

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\ejercicio3.asm

Clear

Assemble: operation completed successfully.

Ejecutando paso a paso

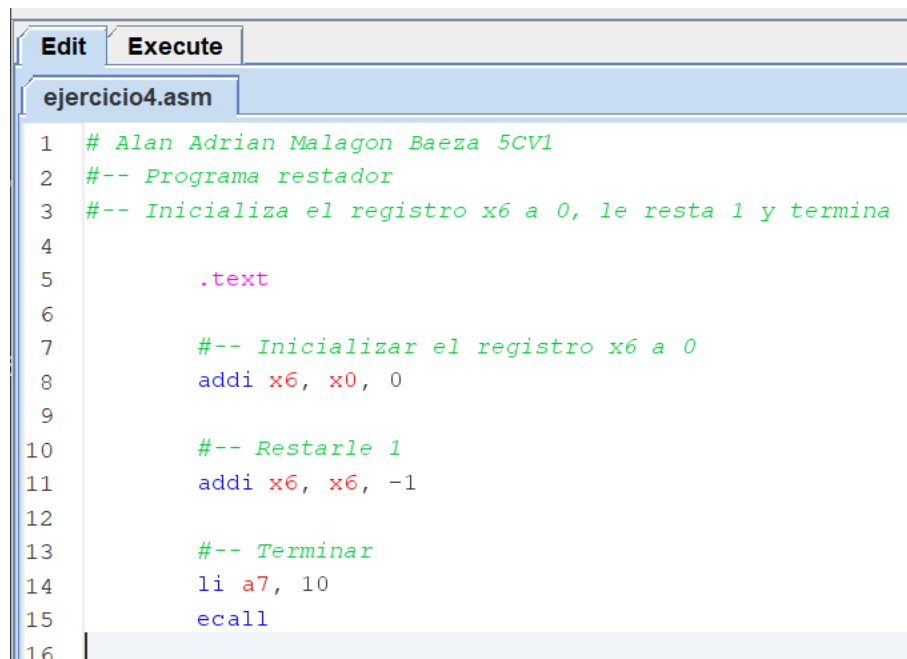


Ejercicio 4

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



The screenshot shows an assembly editor window with two tabs: 'Edit' and 'Execute'. The 'Edit' tab is active, showing the file 'ejercicio4.asm'. The code is as follows:

```
1  # Alan Adrian Malagon Baeza 5CV1
2  |-- Programa restador
3  |-- Inicializa el registro x6 a 0, le resta 1 y termina
4
5      .text
6
7      |-- Inicializar el registro x6 a 0
8      addi x6, x0, 0
9
10     |-- Restarle 1
11     addi x6, x6, -1
12
13     |-- Terminar
14     li a7, 10
15     ecall
16
```


Ensamblando el programa

C:\Users\alanm\Desktop\Arqui\RARs\ejercicio4.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00000313	addi x6,x0,0	8: addi x6, x0, 0
<input type="checkbox"/>	0x00400004	0xffff30313	addi x6,x6,-1	11: addi x6, x6, -1
<input type="checkbox"/>	0x00400008	0x00a00893	addi x17,x0,10	14: li a7, 10
<input type="checkbox"/>	0x0040000c	0x00000073	ecall	15: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Control and Status

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194304

Messages Run I/O

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARs\ejercicio4.asm

Clear Assemble: operation completed successfully.

Ejecutando paso a paso

Inicializa x6 en 0

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio4.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute Run the current program

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00000313	addi x6,x0,0	8: addi x6, x0, 0
<input type="checkbox"/>	0x00400004	0xffff30313	addi x6,x6,-1	11: addi x6, x6, -1
<input type="checkbox"/>	0x00400008	0x00a00893	addi x17,x0,10	14: li a7, 10
<input type="checkbox"/>	0x0040000c	0x00000073	ecall	15: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Control and Status

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194308

Messages Run I/O

Clear Reset: reset completed.

Resta 1

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio4.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Control and Status

Registers Floating Point

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	-1
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194312

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00000313	addi x6,x0,0	8: addi x6, x0, 0
	0x00400004	0xffff30313	addi x6,x6,-1	11: addi x6, x6, -1
	0x00400008	0x00a00893	addi x17,x0,10	14: li a7, 10
	0x0040000c	0x00000073	ecall	15: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Messages Run I/O

Clear Reset: reset completed.

Termina

The screenshot shows the RARS 1.6 emulator window. The main window displays assembly code in the 'Text Segment' and 'Data Segment' tabs. The 'Text Segment' tab is active, showing the following code:

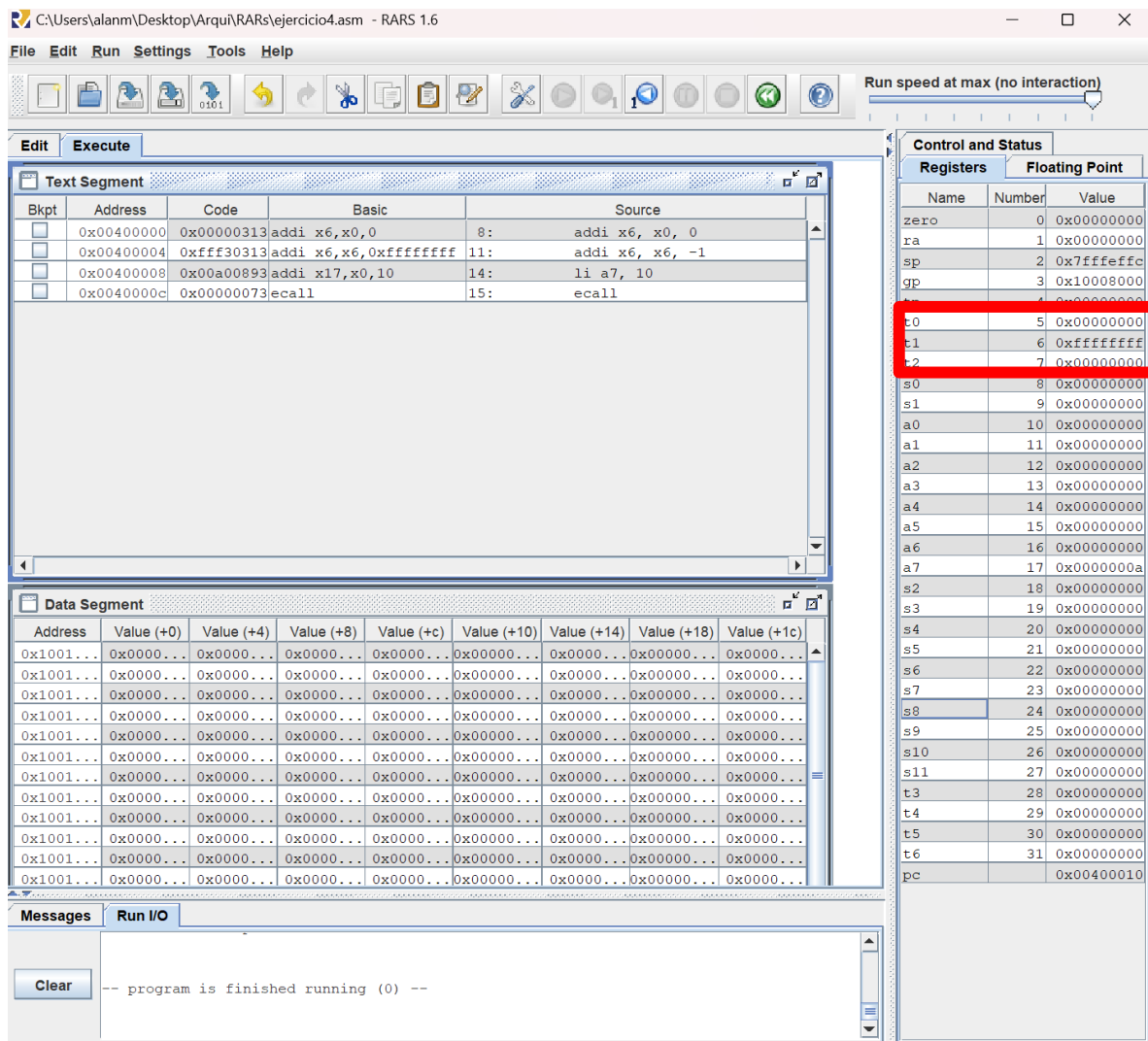
Bkpt	Address	Code	Basic	Source
	0x00400000	0x00000313	addi x6,x0,0	8: addi x6, x0, 0
	0x00400004	0xffff30313	addi x6,x6,-1	11: addi x6, x6, -1
	0x00400008	0x00a00893	addi x17,x0,10	14: li a7, 10
	0x0040000c	0x00000073	ecall	15: ecall

The 'Data Segment' tab shows a memory dump with addresses and values. The 'Control and Status' panel on the right shows the state of registers. The 'Registers' tab is active, showing the following values:

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	-1
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	10
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194320

The 'Messages' panel at the bottom shows the message: -- program is finished running (0) --

El valor hexadecimal del registro 6 al terminar el programa será 0xFFFFFFFFFFFFFFFF, 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.



Ejercicio 5

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

```
.text

addi x3, x0, 10

a:
    addi x3, x3, -1
    bgt x3, x0, a

    li a7, 10
    ecall
```

Código del programa

```
ejercicio5.asm
1  # Alan Adrian Malagon Baeza 5CV1
2      .text
3
4      addi x3, x0, 10
5  a:
6      addi x3, x3, -1
7      bgt x3, x0, a
8
9      li a7, 10
10     ecall
```

Ensamblando el programa

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio5.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00a00193	addi x3,x0,10	4: addi x3, x0, 10
	0x00400004	0xffff18193	addi x3,x3,0xffffffff	6: addi x3,x3,-1
	0x00400008	0xfe304ee3	blt x0,x3,0xffffffffc	7: bgt x3,x0, a
	0x0040000c	0x00a00893	addi x17,x0,10	9: li a7, 10
	0x00400010	0x00000073	ecall	10: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...
0x1001...	0x0000...	0x0000...	0x0000...	0x0000...	0x000000...	0x0000...	0x000000...	0x0000...

Control and Status

Registers

Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7ffffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400000

Messages Run I/O

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\ejercicio5.asm

Clear

Assemble: operation completed successfully.

Ejecutando paso a paso

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio5.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00a00193	addi x3,x0,10	4: addi x3, x0, 10
<input type="checkbox"/>	0x00400004	0xfff18193	addi x3,x3,-1	6: addi x3,x3,-1
<input type="checkbox"/>	0x00400008	0xfe304ee3	blt x0,x3,-4	7: bgt x3,x0, a
<input type="checkbox"/>	0x0040000c	0x00a00893	addi x17,x0,10	9: li a7, 10
<input type="checkbox"/>	0x00400010	0x00000073	ecall	10: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

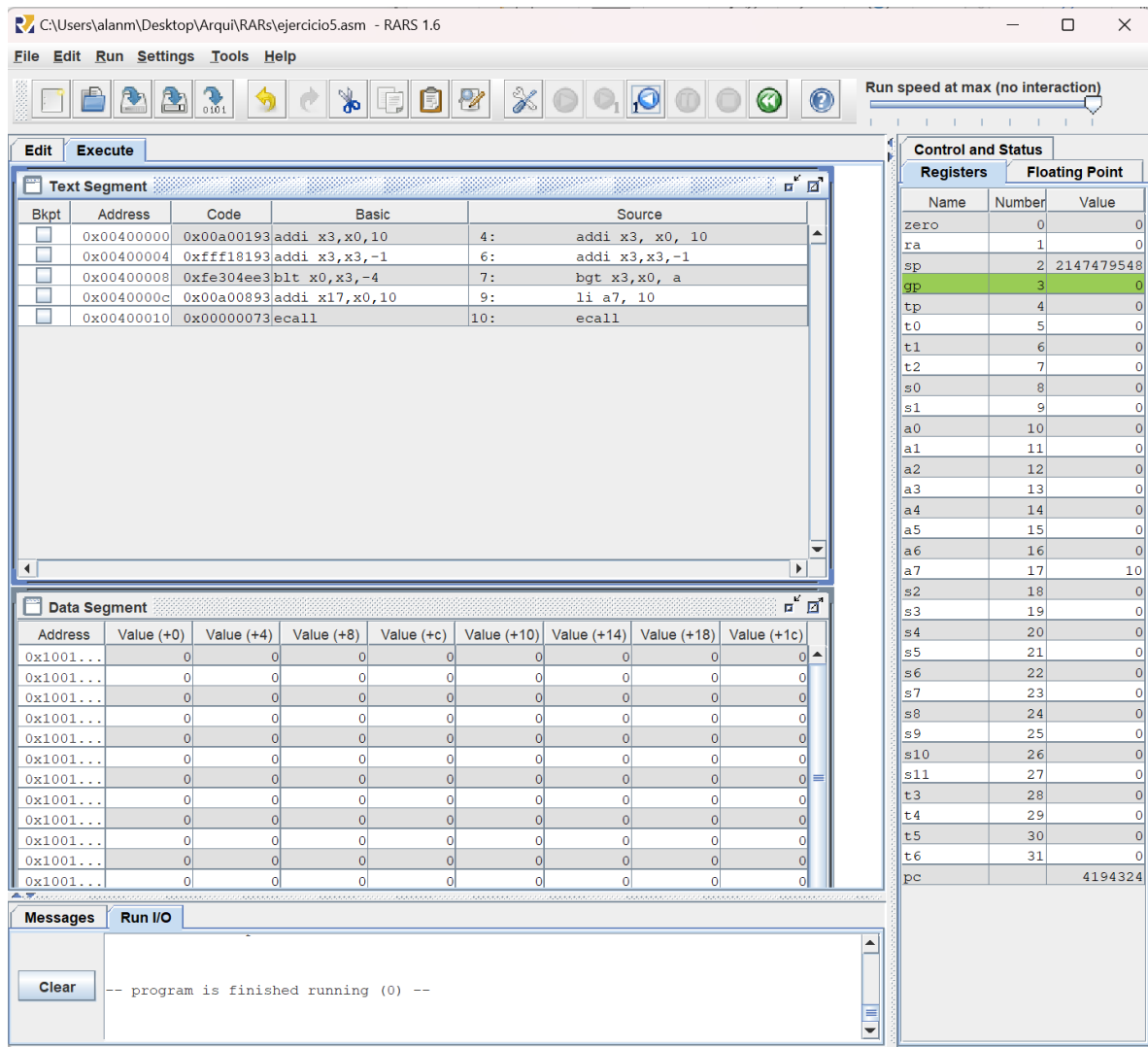
Control and Status

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	10
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194308

Messages **Run I/O**

Clear Reset: reset completed.



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?

```
.text

addi x5, x0, 5
addi x6, x0, 6
addi x7, x0, 0

a:
    beq x5, x0, fin
    add x7, x7, x6
    addi x5, x5, -1
    b a
fin:
    li a7, 10
    ecall
```

Código del programa

```
ejercicio6.asm
1  # Alan Adrian Malagon Baeza 5CV1
2      .text
3
4      addi x5, x0, 5
5      addi x6, x0, 6
6      addi x7, x0, 0
7  a:
8      beq x5, x0, fin
9      add x7, x7, x6
10     addi x5, x5, -1
11     b a
12 fin:
13     li a7, 10
14     ecall
```

Ensamblando el programa

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio6.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00500293	addi x5,x0,5	4: addi x5, x0, 5
	0x00400004	0x00600313	addi x6,x0,6	5: addi x6, x0, 6
	0x00400008	0x00000393	addi x7,x0,0	6: addi x7, x0, 0
	0x0040000c	0x00028863	beq x5,x0,16	8: beq x5,x0,fin
	0x00400010	0x006383b3	add x7,x7,x6	9: add x7, x7, x6
	0x00400014	0xffff28293	addi x5,x5,-1	10: addi x5, x5, -1
	0x00400018	0xff5ff06f	jal x0,-12	11: b a
	0x0040001c	0x00a00893	addi x17,x0,10	13: li a7, 10
	0x00400020	0x00000073	ecall	14: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Control and Status

Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194304

Messages Run I/O

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\ejercicio6.asm

Clear

Assemble: operation completed successfully.

Ejecutando paso a paso

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio6.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00500293	addi x5,x0,5	4: addi x5, x0, 5
<input type="checkbox"/>	0x00400004	0x00600313	addi x6,x0,6	5: addi x6, x0, 6
<input type="checkbox"/>	0x00400008	0x00000393	addi x7,x0,0	6: addi x7, x0, 0
<input type="checkbox"/>	0x0040000c	0x00028863	beq x5,x0,16	8: beq x5,x0,fin
<input type="checkbox"/>	0x00400010	0x006383b3	add x7,x7,x6	9: add x7, x7, x6
<input type="checkbox"/>	0x00400014	0xffff28293	addi x5,x5,-1	10: addi x5, x5, -1
<input type="checkbox"/>	0x00400018	0xff5ff06f	jal x0,-12	11: b a
<input type="checkbox"/>	0x0040001c	0x00a00893	addi x17,x0,10	13: li a7, 10
<input type="checkbox"/>	0x00400020	0x00000073	ecall	14: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Control and Status

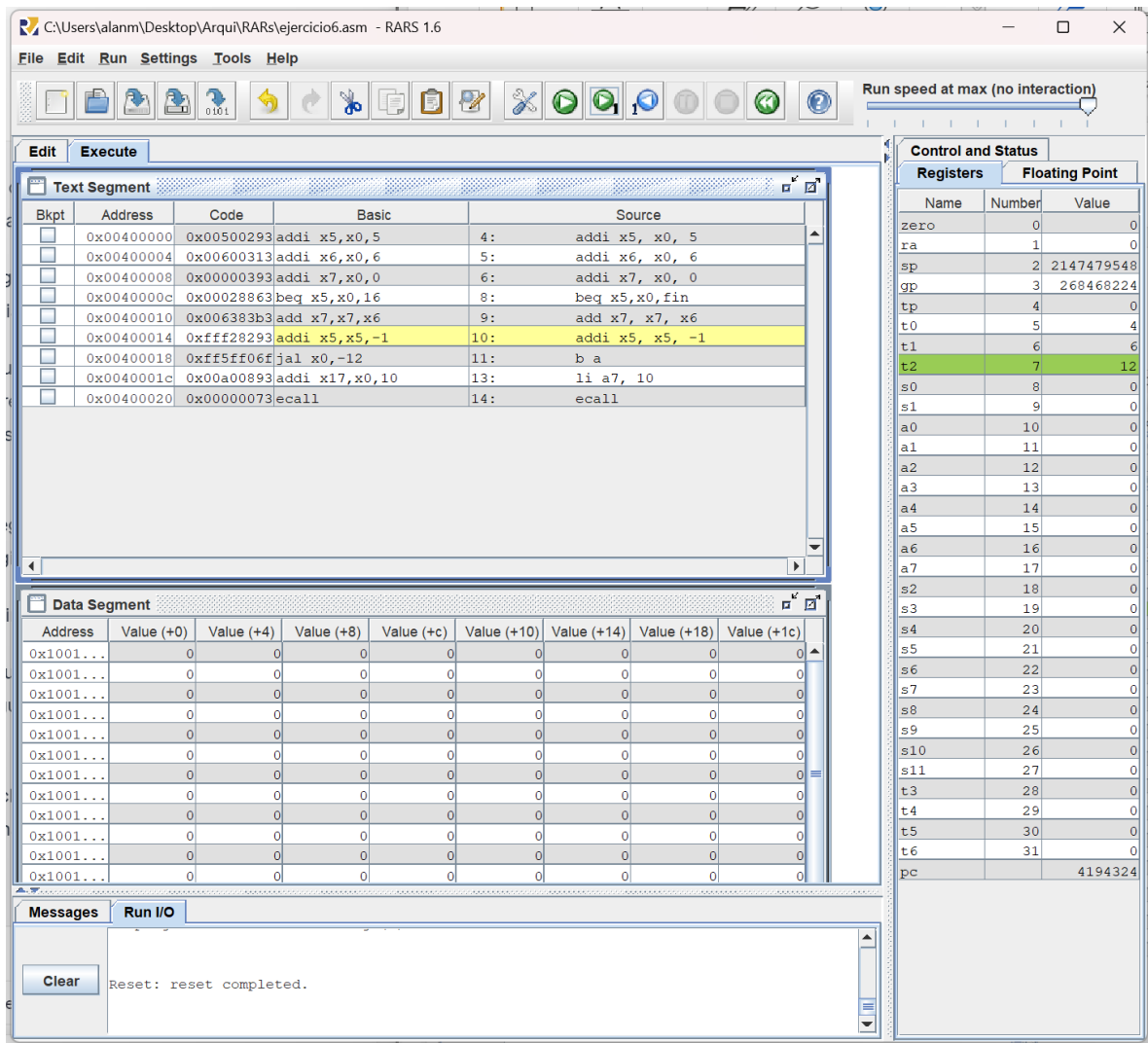
Registers

Name	Number	Value
zero	0	0
ra	1	0
sp	2	2147479548
gp	3	268468224
tp	4	0
t0	5	5
t1	6	6
t2	7	6
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194324

Messages **Run I/O**

Clear

Reset: reset completed.



Este programa en RISC-V realiza un bucle que suma el valor del registro x6 en el registro x5 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?

```
.text

    addi x17, x0, 10
    addi x12, x0, 8
    addi x7, x0, 0
a:
    beq x17,x0,fin ;-- Comprueba terminacion
    add x7, x7, x12
    addi x17, x17, -1

    ;-- repetir

    b hola

fin:
    li a7, 10
    ecall
```

Código del programa con errores

```
ejercicio7.asm
1  # Alan Adrian Malagon Baeza 5CV1
2  .text
3
4      addi x17, x0, 10
5      addi x12, x0, 8
6      addii x7, x0, 0
7  a:
8      beq x17,x0,fin  ;-- Comprueba terminacion
9      add x7, xx7, x12
10     addi x17, x17, -1
11
12     ;-- repetir
13
14     b hola
15
16  fin:
17     li a7, 10
18     ecall
```

Ensamblando el programa con errores

Messages	Run I/O
Error in C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm line 8 column 18: beq x17,x0,fin ;-- Comprueba terminacion	
Invalid language element: ;	
Error in C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm line 12 column 2: ;-- repetir	
Invalid language element: ;	
Error in C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm line 8 column 18: beq x17,x0,fin ;-- Comprueba terminacion	
Invalid language element: ;	
Error in C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm line 12 column 2: ;-- repetir	
Invalid language element: ;	
Assemble: operation completed with errors.	

Clear

Código del programa sin errores

```
ejercicio7.asm
1  # Alan Adrian Malagon Baeza 5CV1
2      .text
3
4      addi x17, x0, 10
5      addi x12, x0, 8
6      addi x7, x0, 0
7
8  a:
9      beq x17,x0,fin  #-- Comprueba terminacion
10     add x7, x7, x12
11     addi x17, x17, -1
12
13     #-- repetir
14
15     jal ra, a
16
17 fin:
18     li a7, 10
19     ecall
20
```


Ensamblando el programa sin errores

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x00a00893	addi x17,x0,10	4: addi x17, x0, 10
<input type="checkbox"/>	0x00400004	0x00800613	addi x12,x0,8	5: addi x12, x0, 8
<input type="checkbox"/>	0x00400008	0x00000393	addi x7,x0,0	6: addi x7, x0, 0
<input type="checkbox"/>	0x0040000c	0x00088863	beq x17,x0,16	9: beq x17,x0,fin #-- Comp..
<input type="checkbox"/>	0x00400010	0x00c383b3	add x7,x7,x12	10: add x7, x7, x12
<input type="checkbox"/>	0x00400014	0xffff8893	addi x17,x17,-1	11: addi x17, x17, -1
<input type="checkbox"/>	0x00400018	0xff5ff0ef	jal x1,-12	15: jal ra, a
<input type="checkbox"/>	0x0040001c	0x00a00893	addi x17,x0,10	18: li a7, 10
<input type="checkbox"/>	0x00400020	0x00000073	ecall	19: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Messages Run I/O

Assemble: assembling C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm

Assemble: operation completed successfully.

Clear

Control and Status

Floating Point

Registers

Name	N...	Value
zero	0	0
ra	1	0
sp	2	2147...
gp	3	2684...
tp	4	0
t0	5	0
t1	6	0
t2	7	0
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	0
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	0
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194304

Ejecutando paso a paso

C:\Users\alanm\Desktop\Arqui\RARS\ejercicio7.asm - RARS 1.6

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x00a00893	addi x17,x0,10	4: addi x17, x0, 10
	0x00400004	0x00800613	addi x12,x0,8	5: addi x12, x0, 8
	0x00400008	0x00000393	addi x7,x0,0	6: addi x7, x0, 0
	0x0040000c	0x00088863	beq x17,x0,16	9: beq x17,x0,fin #-- Comp..
	0x00400010	0x00c383b3	add x7,x7,x12	10: add x7, x7, x12
	0x00400014	0xffff8893	addi x17,x17,-1	11: addi x17, x17, -1
	0x00400018	0xff5ff0ef	jal x1,-12	15: jal ra, a
	0x0040001c	0x00a00893	addi x17,x0,10	18: li a7, 10
	0x00400020	0x00000073	ecall	19: ecall

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0
0x1001...	0	0	0	0	0	0	0	0

Control and Status

Floating Point

Registers

Name	N...	Value
zero	0	0
ra	1	4194332
sp	2	2147...
gp	3	2684...
tp	4	0
t0	5	0
t1	6	0
t2	7	56
s0	8	0
s1	9	0
a0	10	0
a1	11	0
a2	12	8
a3	13	0
a4	14	0
a5	15	0
a6	16	0
a7	17	4
s2	18	0
s3	19	0
s4	20	0
s5	21	0
s6	22	0
s7	23	0
s8	24	0
s9	25	0
s10	26	0
s11	27	0
t3	28	0
t4	29	0
t5	30	0
t6	31	0
pc		4194324

Messages **Run I/O**

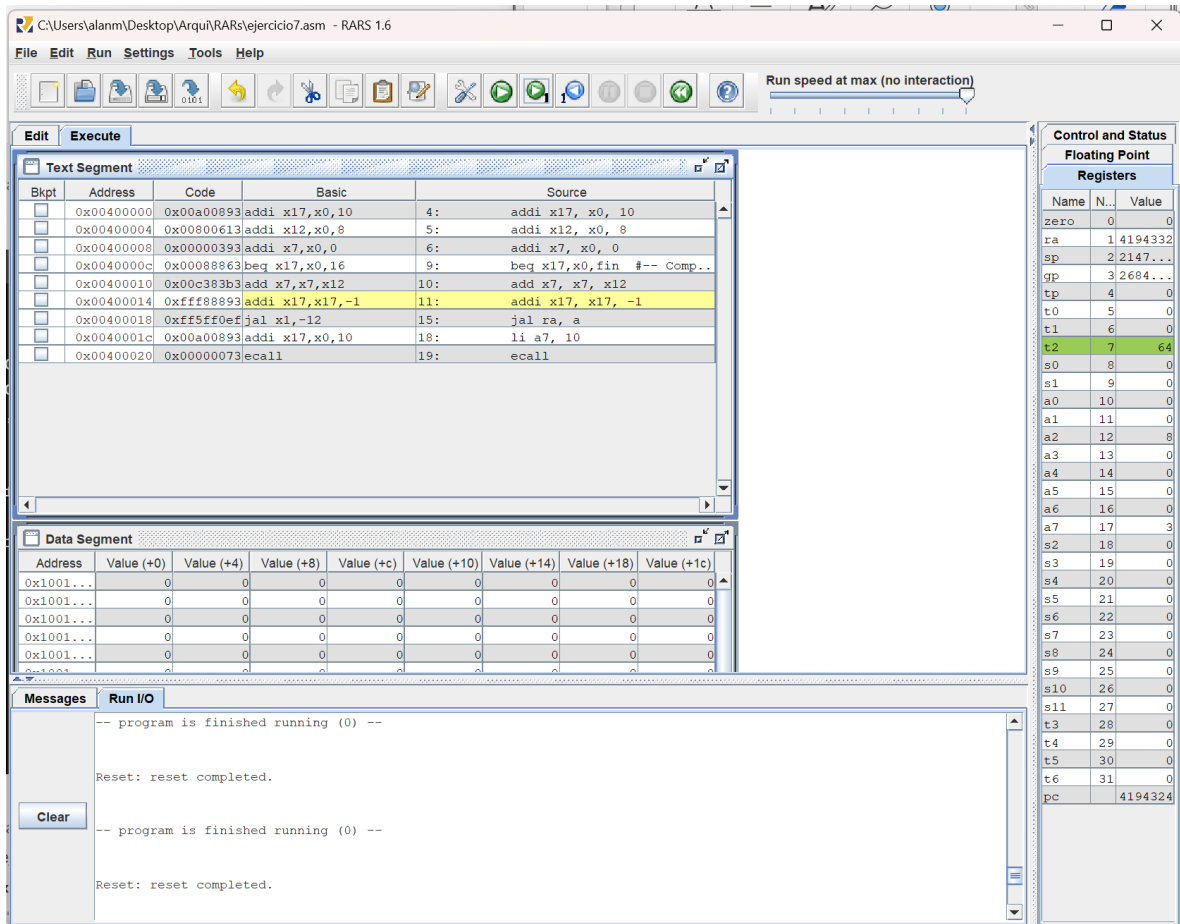
-- program is finished running (0) --

Reset: reset completed.

Clear

-- program is finished running (0) --

Reset: reset completed.



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>