

Universidad de La Laguna

PRINCIPIOS DE COMPUTADORES

Utilización del emulador QTSpim y representación de la información

Aday Padilla Amaya

Índice

1 Introducción	2
2 Ejercicios de la práctica	
a) Identificación en el emulador	
I El segmento de Datos	2
II El segmento de Instrucciones	4
III Registros enteros	6
IV Registros en punto flotante	6
V La consola del sistema	8
b) Editar con un editor de texto plano	8
c) Explorar el segmento de datos	
I Apartado I.	9
II Apartado II.	10
III Apartado III.	11
IV Apartado IV.	12
V Apartado V.	13
VI Apartado VI.	13
VII Apartado VII.	13
d) Reinicia la máquina y vuelve a cargar el p	rograma
en el QtSpim.	
I Apartado I.	13
II Apartado II.	14
III Apartado III.	14
IV Apartado IV.	15
3 Bibliografía	16

1 Introducción

En esta práctica aprenderemos a interpretar y manejar un simulador independiente que ejecuta programas en MIPS32. Lee y ejecuta programas en lenguaje ensamblador escritos para su procesador. Además, contamos con un debugger y un conjunto mínimo de servicios de sistema operativo.

Utilizaremos anotaciones como IEEE-754 tanto en 32 bits como en 64 bits, además de la anotación hexadecimal.

2 Ejercicios de la práctica

- a) Identificar en el emulador los siguientes elementos.
 - I. El segmento de Datos (Usuario, Kernel y Pila)

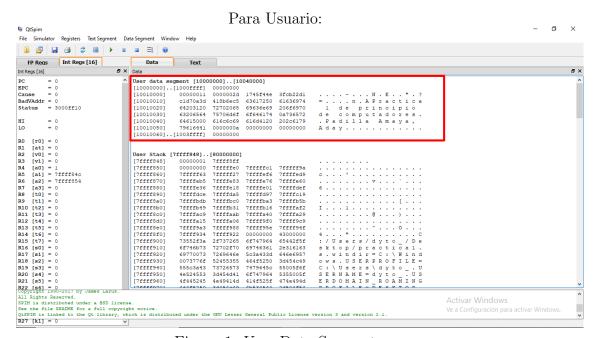


Figura 1: User Data Segment

¿Porqué tiene esa dirección?

Los sistemas basados en procesadores MIPS comúnmente dividen la memoria en tres partes, como podemos ver en la imagen siguiente.

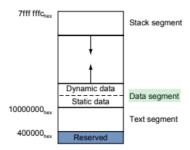


Figura 2:Disposición de la memoria

Ahora entendemos por qué comienza en la posición $10000000_{\rm HEX}$ los datos de Usuario en el simulador. Además, también podemos ver que en la Figura 1 hay varias columnas donde se indica: la dirección en hexadecimal en el registro, el contenido en hexadecimal de la celda y la representación de los datos en ASCII (de izquierda a derecha respectivamente).

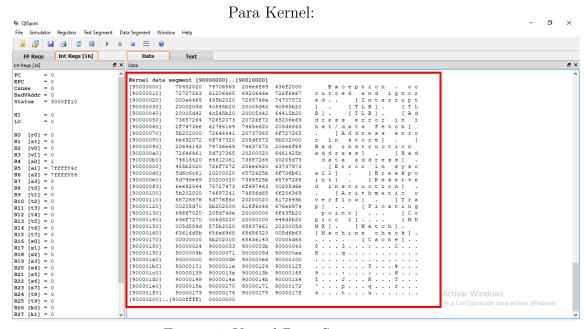


Figura 3: Kernel Data Segment

En el recuadro rojo podemos ver los datos en el kernel. Su bloque de datos acapara desde la posición $90000000_{\rm HEX}$ hasta la $90010000_{\rm HEX}$.

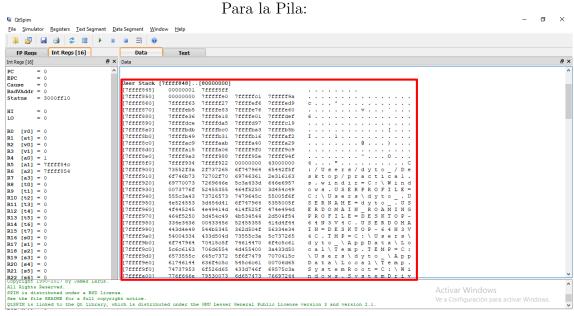


Figura 4: Stack Data Segment

El bloque de la Pila es grande (en comparación con los otros dos) y requeriría de mas de tres imágenes para ilustrar todo su contenido. Sin embargo, podemos ver que comienza en la dirección 7ffff848 $_{\rm HEX}$ y acaba en la posición $8000000_{\rm HEX}$. Si restamos esas cantidades, obtenemos como resultado que nos ocupa $7B8_{\rm HEX}$ (1976 $_{\rm DEC}$) direcciones para los bloques de memoria.

II. El segmento de Instrucciones (Usuario y Kernel)

Ahora veremos en el simulador donde se pueden observar las instrucciones cargadas en él y en qué dirección de registro se guardan. Al igual que el segmento de datos, disponemos de varias columnas las cuales indican; 1 direcciones, 2 Opcodes, 3 Mnemónicos y 4 Pseudo-Instrucciones (Figura 5). La primera instrucción se encuentra en la posición $0x00400000_{\rm HEX}$ de la memoria, siendo la primera celda de las no reservadas (Figura 2).

Para el Usuario: QtSpim File Simulator Registers Text Segment Data Segment Window Help FP Regs Int Regs [16] lw \$4, 0(\$29) addiu \$5, \$29, 4 addiu \$6, \$5, 4 sll \$2, \$4, 2 addu \$6, \$6, \$6, \$2 jal 0x00400024 [main] nop ori \$2, \$0, 10 syscall lui \$1, 4097 [titulo] ori \$4, \$1, 24 [titulo] ori \$2, \$0, 4 syscall lui \$1, 4097 [nombre] ori \$4, \$1, 65 [nombre] ori \$2, \$0, 4 syscall lui \$1, 4097 [num1] lui \$1, 4097 [num1] lui \$1, 4097 [num2] lui \$1, 4097 [num2] add \$10, \$8, \$9 add \$12, \$10, \$11 ori \$14, \$0, 555 beg \$14, \$0, 555 beg \$14, \$0, 555 beg \$14, \$0, 16 [fin_buol] addi \$11, \$11, -1 bger \$0 -12 [buolewhile-faddi \$14, \$14, -1 FIR FCSR FCCR FEXR = 9800 = 0 = 0 User Text Segment [00400000]..[00440000] User Text Segment [00400000]...[004: ! 183: 1 V 500 (/Sp) # argc ! 184: addiu 5a1 Sap 4 # argv ! 184: addiu 5a1 Sap 4 # argv ! 186: addiu 5a2 Sa1 4 # envp ! 186: sll 5v0 5a0 2 ! 187: addu 5a2 Sa2 \$v0 ! 188: jal main ! 189: nop ! 191: 1! \$v0 10 ! 182: syscall # syscall 10 (exit) ! 12: la \$a0,titulo 00400000] sfa40000 00400004] 2745004 00400040] 2460004 00400000] 0023021 00400001] 0023021 00400010] 0023021 00400010] 0023021 0040001014] 002000000 004000102] 34020004 00400020] 34240018 00400020] 34240018 00400020] 34240018 00400030] 34240018 00400030] 34240018 00400030] 34240018 00400030] 34240018 00400030] 34240018 00400030] 34240041 00400030] 34240041 Single Precision PGO = 0 PGI = 0 PGI = 0 PG2 = 0 PG3 = 0 PG4 = 0 PG5 = 0 PG6 = 0 PG7 = 0 PG9 = 0 PG9 = 0 PG11 = 0 PG11 = 0 PG12 = 0 PG13 = 0 PG14 = 0 ; 13: li \$v0,4 ; 14: syscall ; 17: la \$a0,nom ; 18: li \$v0,4 ; 19: syscall ; 21: lv \$t0,num1 # carga en el registro \$t0 el valor etiquetado como numi 0400044] 3c011001 0400048] 8c280000 040004c] 3c011001 0400050] 8c290004 ; 22: lw \$t1,num2 # carga en el registro \$t1 el valor etiquetado como num2 ; 33: add \$t2,\$t0,\$t1 \$ realiza la siguiente operacion \$t2 = \$t0 + \$t1 ; 25: add \$t4, \$t2,\$t3 \$ realiza la siguiente operacion \$t4 = \$t2 + \$t3 ; 29: li \$t6,555 clewhile-0x00400060] ; 31: add; \$t3,-1 -0x00400066, ; 39: move \$a0,\$t3 040Q054] 01095020 00400058] 014b6020 00400058] 01406022b 00400050] 340e022b 00400060] 11c00004 00400064] 21cbffff 00400060] 21ceffff 00400060] 0401fffd 00400070] 000b2021 3 4 Activar Windows

Figura 5: User Text Segment



Figura 6: Kernel Text Segment

Comienza al acabar el segmento de instrucciones del usuario.

III. El contenido de los Registros Enteros

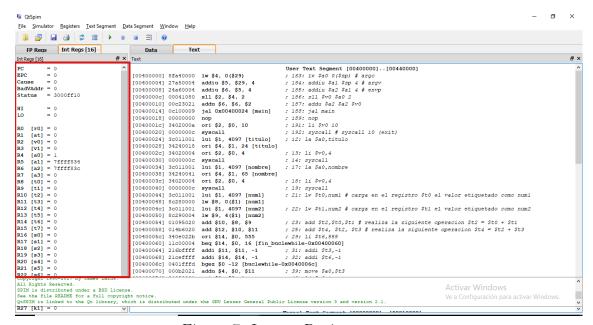


Figura 7: Integer Registers

En la pestaña podemos ver qué valores guardan las variables enteras, desde el PC (Program Counter) hasta las $k\theta$ -k1 (Reservadas para el Kernel OS) pasando por las $a\theta$ -a3, $t\theta$ -t7, $s\theta$,s7... Todos ellos pueden ser modificados directamente por nosotros en cualquier momento.

IV. El contenido de los Registros en Punto Flotante

En la Figura 8 se muestran los registros de formato en coma flotante de precisión simple y en la Figura 9 en precisión doble. No olvidar que el simulador los expresa en hexadecimal (tanto los registros, como el segmento de datos y el de instrucciones se pueden cambiar a binario o a decimal si lo prefiere).

```
<u>File Simulator Registers Text Segment Data Segment Window Help</u>
           FP Regs Int Regs [16]
                                                                                                                                                                                                                                                                                                                    Data
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             User Text Segment [0040000]..[00440000]; 183; 1v Sa0 O(Sap) # argc; 184; addin Sai Sap 4 # argv; 185; addin Sai Sap 4 # argv; 185; addin Sai Sai 4 # envp; 186; sl1 SvO SaO 2; 187; addu Sai Sai SvO; 188; jai main; 183; nop; 181; 11 SvO 10; 183; nop; 191; 11 SvO 10; 192; syecall # syecall 10 (exit); 182; la SaO,titulo
    FP Regs
  FIR
FCSR
FCCR
FEXR
                                                              = 9800
= 0
= 0
= 0
                                                                                                                                                                                                                                                                                       [00400000] 8fa40000 lw $4, 0 ($29) [
[00400004] 27a50004 addiu $5, $29, 4 [
[00400008] 24a60004 addiu $5, $5, 4 [
[00400008] 00041008 sil $2, $4, 2 [
[0040001] 00023021 addu $6, $6, $2 [
[0040001] 0010009 jal 0x000400024 [main] [
[0040001] 30000000 np [
[00400012] 30400000 ori $2, $0, 10 [
[00400012] 30501001 lii $1, 4097 [titulo] [
[00400023] 32011001 lii $1, 4097 [titulo] [
[00400023] 34240018 ori $2, $0, 12 [
[00400023] 34200004 ori $2, $0, 12 [
[00400023] 34200004 ori $2, $0, 12 [
[00400023] 3400004 ori $2, $0, 12 [
[00400023] 340004 ori $2, $0, 12 [
[00400023] 3400004 ori $2, $0, 12 [
[00400023] 340004 ori $2, $0, 12 [
[00400023] 340004 ori $2, $0, 12 [

Single Precision
FGO = 0
FG1 = 0
FG2 = 0
FG3 = 0
FG3 = 0
FG4 = 0
FG5 = 0
FG6 = 0
FG7 = 0
FG7 = 0
FG9 = 0
FG10 = 0
FG11 =
                                                                                                                                                                                                                                                                                                                                                                                                                                                          ori $2, $0, 10
syscall
lui $1, 4097 [titulo]
ori $4, $1, 24 [titulo]
ori $2, $0, 4
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  ; 13: li $v0,4
; 14: syscall
; 17: la $a0,nombre
                                                                                                                                                                                                                                                                                                                                                                                                                                                     ori $2, $3, 24 [rituto] ori $2, $5, $4 sysoall lui $1, 4097 [nombre] ori $4, $1, 65 [nombre] ori $2, $5, $0, $4 sysoall lui $1, 4097 [num1] lw $8, 0 ($1) [num1] lw $8, 0 ($1) [num2] lw $9, 4 ($1) [num2] add $10, $8, $9 add $12, $10, $11 ori $14, $0, 55 beq $14, $0, 16 [fin_buc_addi $11, $11, -1 addi $14, $14, -1 begz $0 -12 [buclewhile—addu $4, $0, $11
                                                                                                                                                                                                                                                                                                        0040002c] 34020004
00400030] 0000000c
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  ; 18: 11 $v0,4 ; 19: syscall ; 21: 10 $corps en el registro $t0 el valor etiquetado como numl
                                                                                                                                                                                                                                                                                                     00400044] 3c011001
00400048] 8c280000
                                                                                                                                                                                                                                                                                                 00400048] 8c280000

0040004c] 3c011001

00400050] 8c290004

00400054] 01095020

00400058] 014b6020

0040005c] 340e022b

00400060] 11c00004
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  ; 22: lw $t1,num2 # carga en el registro $t1 el valor etiquetado como num2
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         , 23: add $t2,$t0,$t1 f realize la siguiente operacion $t2 = $t0 + $t1
, 28: add $t4, $t2, $t3 f realize la siguiente operacion $t4 = $t2 + $t3
, 29: li $t6,555
clewhit=-0x00400060]
, 31: addi $t2,-1
, 32: addi $t2,-1
, 33: addi $t6,-1
, 39: move $40,$t3
                                                                                                                                                                                                                                                                                                             0400064] 216bffff
                                                                                                                                                                                                                                                                                                               0400068] 21ceffff
040006c] 0401fffd
0400070] 000b2021
  COSPTING 1997-2017 by James Larus.
All Rights Reserved.
SPH is distributed under a BSD license.
See the file ERANME for a full copyright notice.
QCSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

Figura 8: Single Precision Registers

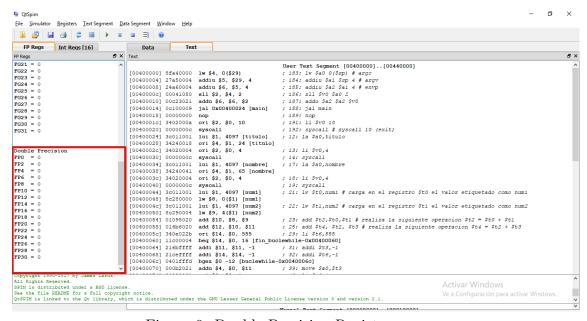


Figura 9: Double Precision Registers

V.La consola del sistema



Figura 10: Console window

Si el programa realiza una salida o entrada será dirigida a la consola.

b) Edita con un editor de textos plano.

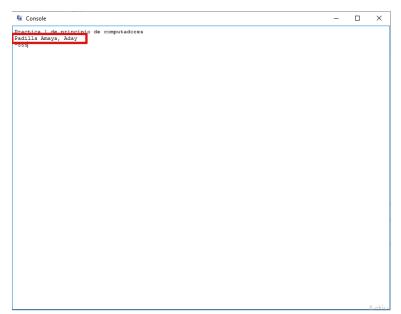


Figura 11: Console window

c) Explora el segmento de datos.

I.¿Qué dirección de memoria (expresa la dirección en hexadecimal) ocupa el último carácter de tu primer apellido (p.ej: en 'Simpson, Homer' el carácter en cuestión es 'n')?

Ahora explicaré cómo podemos interpretar el segmento de datos de usuario. En la Figura 12, en la posición $10010040_{\rm HEX}$ vemos todo el contenido de la memoria en la celda. Empezando desde la primera columna de la izquierda, con el contenido 64615000, el byte con la dirección de memoria mas baja es el primero de la derecha, es decir, el 00.

User data segment [1000	0000][10	040000]															
[10000000][1000ffff]	00000000																
[10010000] 00000011	0000002d	1745f44e	3fcb22dl				-			Ν		E			**		?
[10010010] cld70a3d	418b6ec5	63617250	61636974	=				n.	A	P	r	а	С	t	i	С	a
[10010020] 64203120	72702065	69636e69	206f6970		1	d	e	p	r	i	n	С	i	p	i	0	
[10010030] 63206564	75706d6f	6f646174	0a736572	d	e	С	0	m p	u	t	а	d	0	r	e	s	
[10010040] 64615000	616c6c69	616d4120	202c6179		Рa	d	i	1 1	a		Α	m	a	У	a	,	7
[10010050] 79616441	00000000a	00000000	00000000	A	d a	У											Ξ.
[10010060][1003ffff]	00000000																

Figura 12: Data Segment

Para una mejor visualización, la Figura 13 muestra como se distribuye cada contenido de la celda con los datos cargados al programa (el color azul representa el valor en hexadecimal y en naranja la posición). Por ejemplo, en la posición '0' vemos que esta guardado el valor $00_{\rm HEX}$ (que contiene 1 byte, es decir, 8 bits). No olvidar la posición $F_{\rm HEX}$, que aunque no muestre nada, ahí hay un byte, porque como dimos en clase, también hay caracteres no imprimibles (tabulador, enter,...). Entonces, la dirección que ocupa la ultima palabra de mi primer apellido (a) sería la 7, es decir, 0x10010047.

•	▮▮	a	u			I	a		A	6d A	a	y	a	,	
00	50	61	64	69	6c	6c	61	20	41	6d	61	79	61	2c	20
0	1	2	3	4	5	6	7	8	9	Α	В	С	D	Е	F

Figura 13: Data Segment with Address

II.¿Qué carácter es y qué representación tiene en hexadecimal?

El carácter correspondiente es la a con la representación 0x61.

```
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
[10010000] 00000011 0000002d 1745f44e 3fcb22d1
                                                  = . . . . n . A P r a c t i c a
[10010010]
            cld70a3d 418b6ec5 63617250 61636974
           64203120 72702065 69636e69 206f6970 63206564 75706d6f 6f646174 0a736572
                                                  l de principio
[10010020]
[10010030]
                                                  de computadores.
          64615000 618c6c69 616d4120 202c6179
[10010040]
                                                  . Padilla Amaya,
[100100501
            79616441
                      0000000a 00000000 00000000
                                                  Aday.........
[10010060]..[1003ffff]
                     00000000
```

Figura 14: Hexadecimal Data

Podemos comprobarla en la tabla de caracteres US-ASCII.

USASCII code chart ۰, 0 0 o 0 þ 2 6 7 2 3 4 Rowi p 0 0 NUL DLE SP 0 0 ø 0 1 SOH DC1 ļ A Q 0 0 п 1 a q 0 0 0 STX DC2 2 8 R ь r 3 DC3 o 0 ı ETX # 3 C 3 EOT T 0 4 DC4 4 D 0 1 0 ١ d ٠ 5 NAK % 5 E 0 0 ı ENG U w 0 0 6 ACK SYN 8 6 F ٧ f 1 ٧ 7 BEL ETB 7 0 G ₩ 1 1 w 8 CAN 8 0 0 0 BS. t н X. h × 9 9 нΤ ΕM) 1 Υ 0 0 0 1 10 LF SUB ٠ î. J Z z 1 П C 1 VΤ ESC ĸ k 0 0 12 FF FS € L ٨ 1 1 М ī 0 13 CR 65 3 m > ı 0 14 ı SO AS. N Α. m 15 1 0 DEL US 51 0

Figura 15: US-ASCII Table

III.Busca en el segmento de datos de QtSpim el número que se encuentra en la dirección etiquetada como num3.

Primero miramos qué valor tiene la variable num3 en nuestro código.

$$num3 = 0x1745F44E$$

Ahora pasamos el valor 0x1745F44E a los caracteres que correspondan:

 $17_{\text{HEX}} = ETB$ (END-OF-TRANSMISSION-BLOCK, No imprimible)

 $45_{\text{HEX}} = E$

 $F4_{HEX} = Out \ of \ Range \ (No \ imprimible)$

 $4E_{HEX} = N$

						US	ASCII	code	chart				
B. Deb	5 -				=_	°°,	°° ,	۰, ٥	۰,	'°°	١٥,	' _{'0}	'',
	b4+	b ₃	p 5	١,	ROW	0	-	2	3	4	5	6	7
,	0	0	0	0	0	NUL .	DLE	SP	0	@	Р	,	P
	0	0	0	ı	1	SOH	DC1	!	1	Α.	0	o	q
	0	0	1	0	2	STX	DCS		2	В	R	. b	r
	0	0	1	1	3	ETX	DC3	#	3	C	S	c	5
	0	1	0	0	4	EOT	DC4		4	D	T	d	1
	0	1	0	1	5	ENQ	NAK	%	5	E	υ	•	U
	0	1	1	0	6	ACK	SYN	8	6	F	v	1	v
	0	1	1	1	7	BEL	ETB	,	7	G	w	g	w
	1	0	0	0	8	BS	CAN	(8	н	x	h	x
	-	0	0	1	9	нТ	EM)	9	1	Y	i	у
	-	0	1	0	10	LF	SUB	*	:	J	z	j	z
	1	0	1	1	11	VT	ESC	+	:	K	C	k.	(
	1	1	0	0	12	FF	FS		<	L	١.	- 1	1
	1	1	0	1	13	CR	GS	-	-	м)	m)
	,	.1	1	0	14	so	RS		>	N	^	n	\sim
	1	1	1	1	15	\$1	υs	/	?	0	_	0	DEL

Figura 16: US-ASCII Table

Ahora buscamos el bloque que contenga el valor 1745 F44
E $_{\rm HEX}$ en el segmento de datos de usuario:

```
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
[10010000] 00000011 0000002d 1745f44e 3fcb22d1
[10010010]
           cld70a3d 418b6ec5 63617250 61636974
                                                    = . . . . n . A P r a c t i
           64203120 72702065 69636e69 206f6970 63206564 75706d6f 6f646174 0a736572
                                                   l de principi
[10010020]
[10010030]
                                                    de computador
            64615000 616c6c69 616d4120 202c6179
[100100401
                                                    . Padilla Amaya
            79616441 0000000a 00000000 00000000
[10010050]
                                                    Adav....
[10010060]..[1003ffff] 00000000
```

Figura 17: User Data Segment

Por lo que el valor num3 está entre 0x10010008 y 0x1001000B

IV.Convierte el número 1.587 a formato IEE-754 para 32 bits. Busca ahora este número en el segmento de datos.

El número 1.587 corresponde a $3 \text{fcb} 22 \text{d} 1_{\text{HEX}}$. Ahora, como el apartado anterior, los pasamos a los caracteres que corresponda de la tabla ASCII y los buscamos en el segmento de datos de usuario:

```
3F_{HEX} = ?
cb_{HEX} = Out \ of \ Range \ (No \ imprimible)
22_{HEX} = "
dl_{HEX} = Out \ of \ Range \ (No \ imprimible)
```

```
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
                                                    . . . . - . . . N . E . . " . ?
[10010000] 00000011 0000002d 1745f44e 3fcb22d1
                                                  =...n.APractic
l de principio
[10010010]
           cld70a3d 418b6ec5 63617250 61636974
            64203120 72702065 69636e69 206f6970 63206564 75706d6f 6f646174 0a736572
[10010020]
                                                    de computadores
[100100301
[10010040]
             64615000 616c6c69 616d4120 202c6179
                                                    . Padilla Amaya,
            79616441
                      0000000a
                               00000000 00000000
                                                    Aday........
[10010050]
[10010060]..[1003ffff]
                      00000000
```

Figura 18: User Data Segment

V.; En qué dirección empieza el número 1.587?

Como podemos ver en la Figura 18, comienza en la posición 0x100100C y acaba en la 0x100100F.

VI.Convierte el número 57530552.23 a formato IEE-754 para 64 bits.Busca ahora este número en el segmento de datos.

El número $57530552.23_{\rm DEC}$ equivale a $418B6EC5C1D70A3D_{\rm HEX}$. Se encuentra en:

Figura 19: User Data Segment

VII.; En qué dirección empieza el número 57530552.23?.

El número $57530552.23_{\rm DEC}$ empieza en la posición del registro 0x10010010.

d) Reinicia la máquina y vuelve a cargar el programa en el QtSpim.

I.Ejecuta paso a paso el programa.

Contiene el valor $3E_{HEX}$ y equivale a 62_{DEC} . Lo podemos corroborar observando el código fuente, ya que num3 = num2 + num1, donde num1 = 17 y num2 = 45, luego num3 = 62.

II.Cuando hayas terminado de ejecutar esta instrucción, modifica a mano el valor del registro t3.

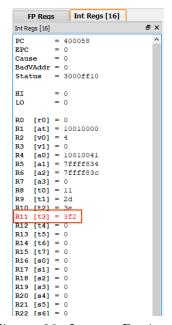


Figura 20: Integer Registers

III.A continuación sigue ejecutando paso a paso hasta terminar de ejecutar la instrucción add t4,t2,t3.

Contiene el valor $430_{\text{HEX}} = 1072_{\text{DEC}}$.

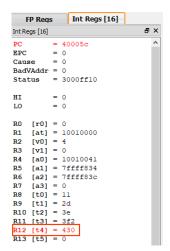


Figura 21: Integer Registers

IV.A continuación establece un punto de ruptura.

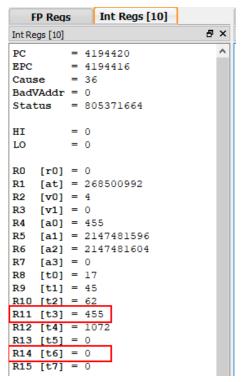


Figura 22: Integer Registers

El resultado es $t3=1c7_{HEX}=455_{DEC}$ y $t6=0_{HEX}=0_{DEC}$.

¿Porqué tienen esos valores?

Como en el apartado anterior cambiamos el valor de la variable $t3=1010_{DEC}$, el bucle se parará cuando t3 o t6 lleguen a θ . Sabiendo que $t6=555_{DEC}$, entonces llegará antes a θ , así que t3 será la diferencia de ambos.

$$t3 = 1010 - 555 = 455_{DEC}$$

3. Bibliografía

```
https: //drive.google.com/file/d/1Zye1G -
6Sri2KYBAAAze6Sjfn3yJqr_PJ/view
//drive.google.com/file/d/1_64OxIFydYLyI58HECCw1hOC_Yk50nHt/view
https:
//drive.google.com/file/d/1_69X7LQSDRbkkP5XLkl6BXQagF9EJq-
v/view
Estructura de computadores, programacion del procesador MIPS y su en samblador
Fundamentos y estructura de computadores/Jose M^a Angulo U sategui, Javier Garcia Zubia
http://spimsimulator.sourceforge.net/
http://spimsimulator.sourceforge.net/spim.pdf
http://spimsimulator.sourceforge.net/xspim.pdf
http://spimsimulator.sourceforge.net/PCSpim.pdf
http://spimsimulator.sourceforge.net/SPIM_command-line.pdf
https: //www.binaryhexconverter.com/hex-to-decimal-converter
https: //babbage.cs.qc.cuny.edu/ieee - 754.old/decimal.html
https: //www.h-schmidt.net/FloatConverter/IEEE754.html
https: //www.adobe.com/es/products/photoshopfamily.html
https: //www.latex - project.org/
https://wiki.archlinux.org/index.php/locale
https: //wiki.archlinux.org/index.php/TeX_Live
```