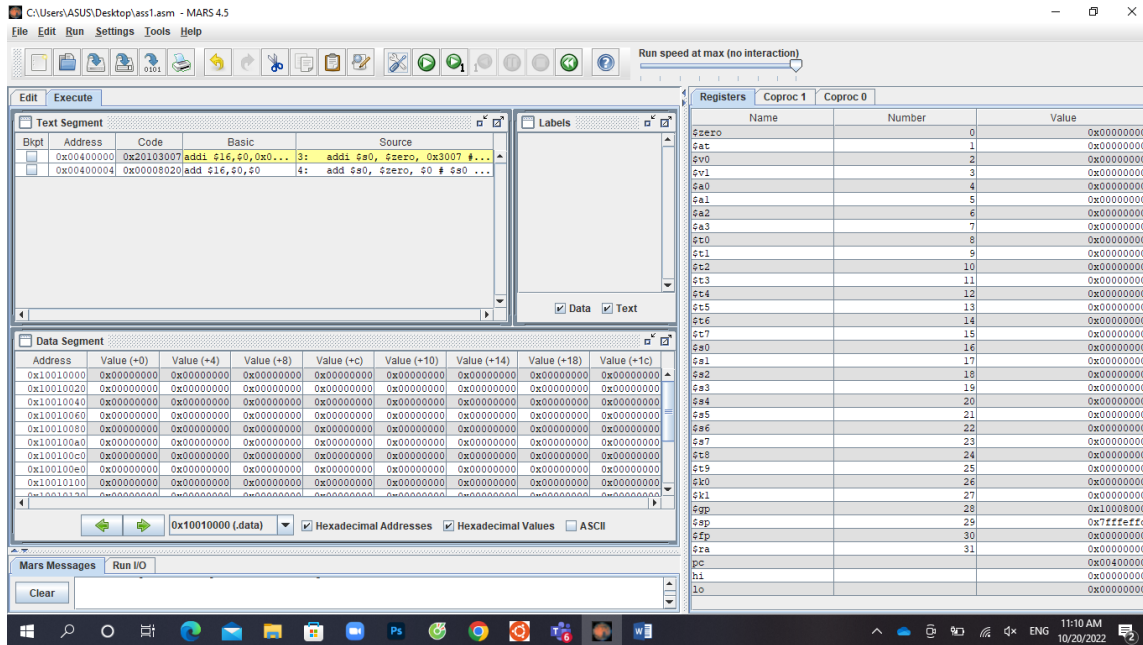


Assignment 1:

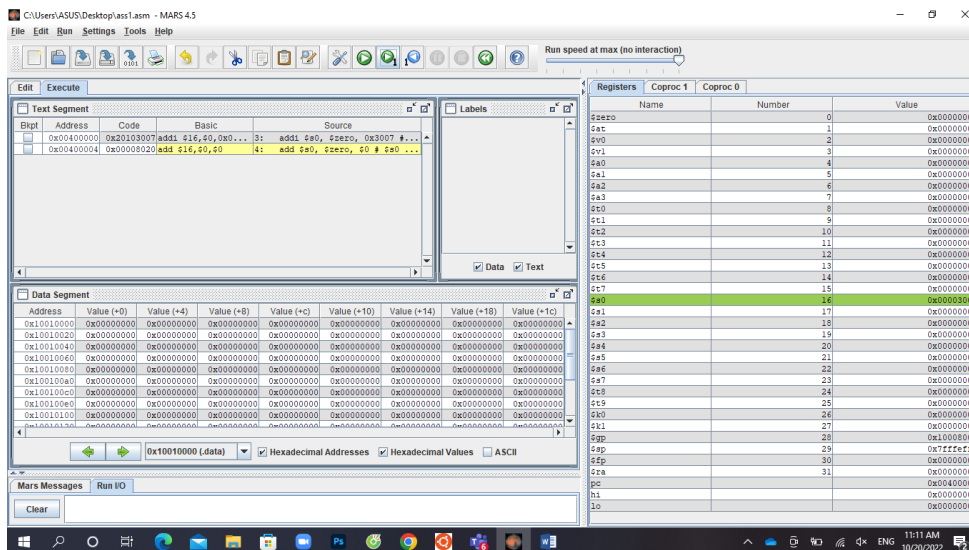
Start:



\$s0	16	0x00000000
------	----	------------

pc		0x00400000
----	--	------------

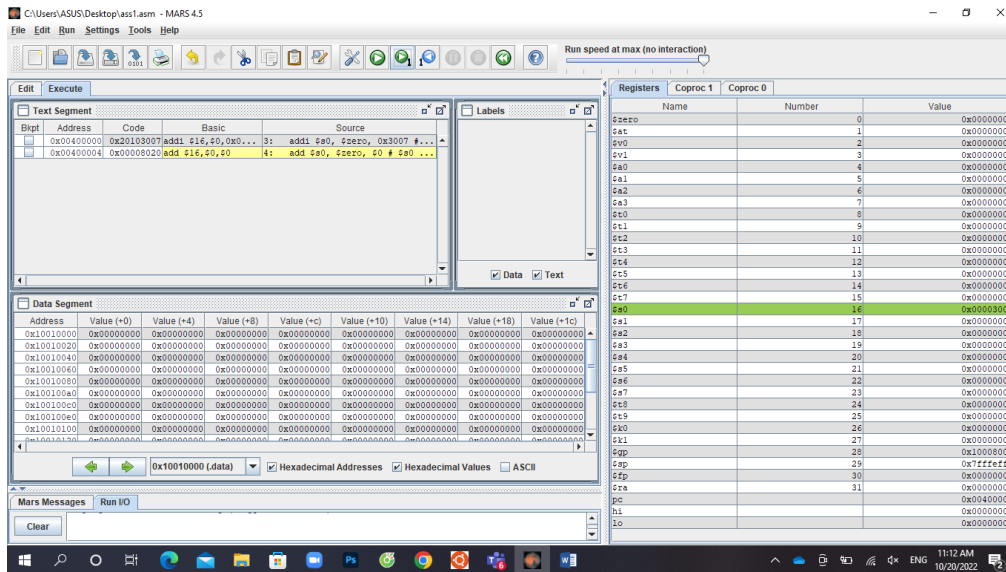
Step 1:



\$s0	16	0x00003007
------	----	------------

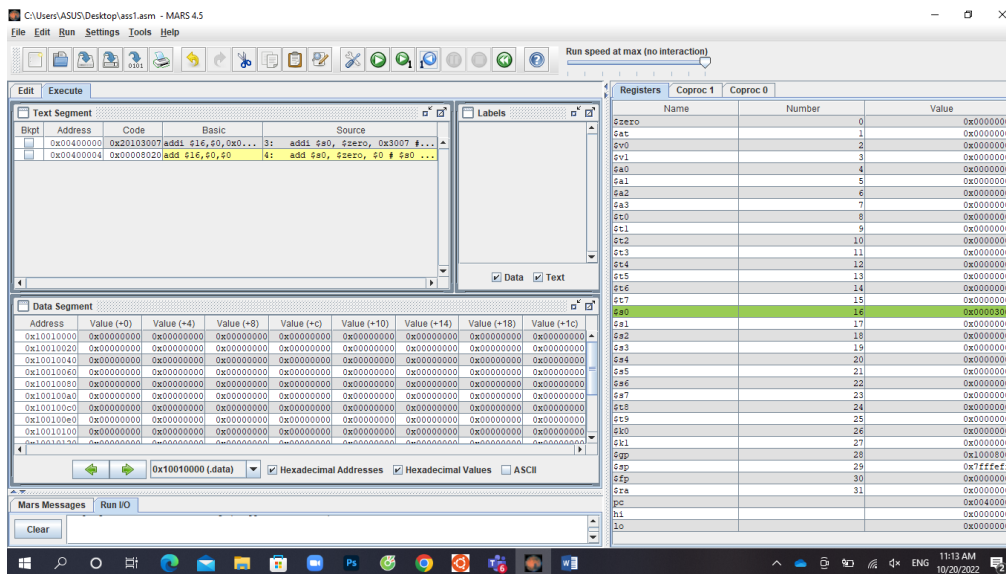
pc		0x00400004
----	--	------------

Step 2:



\$s0	16	0x00000000
pc		0x00400008

Step 3:



\$s0	16	0x00000000
pc		0x00400008

- Ô cửa sổ Text Segment:

Lệnh đầu tiên là 0x20103007: 0010 0000 0001 0000 0011 0000 0000 0111
001000 00000 10000 0011000000000111

Op: 8 (I)(addi)

Rs: 0 (\$zero)

Rt: 16 (\$s0)

Imm: 12295

Đúng với: addi \$s0, \$zero, 0x3007

- Khi sửa lại lệnh lui:

Sau khi sửa lệnh addi \$s0, \$zero, 0x3007 thành addi \$s0,\$zero, 0x2110003d có nghĩa là chúng ta đang đòi hỏi việc cộng s0 với số 32bit. Tuy nhiên, lệnh addi chỉ phục vụ cộng số 16bit do đó sau khi chạy chương trình hiện lên như sau:

Text Segment				
Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x3c012110	lui \$1,0x00002110	3: addi \$s0, \$zero, 0x2110003d # \$s0 = 0 + 0x...
<input type="checkbox"/>	0x00400004	0x3421003d	ori \$1,\$1,0x0000003d	
<input type="checkbox"/>	0x00400008	0x00018020	add \$16,\$0,\$1	
<input type="checkbox"/>	0x0040000c	0x00008020	add \$16,\$0,\$0	4: add \$s0, \$zero, \$0 # \$s0 = 0 + 0 = 0 ;R-type

Assignment 2:

Start

C:\Users\ASUS\OneDrive - Hanoi University of Science and Technology\KMT\Assignment_Lab\week2\ass2.asm - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c102110	lui \$t6,0x00002110	3: lui \$s0,0x2110 #put upper half of pattern ...
	0x00400004	0x3610003d	ori \$t6,\$t6,0x0000003d	4: ori \$s0,0x003d #put lower half of pattern ...

Labels

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Mars Messages

Run I/O

Assemble: assembling C:\Users\ASUS\OneDrive - Hanoi University of Science and Technology\KMT\Assignment_Lab\week2\ass2.asm

Clear

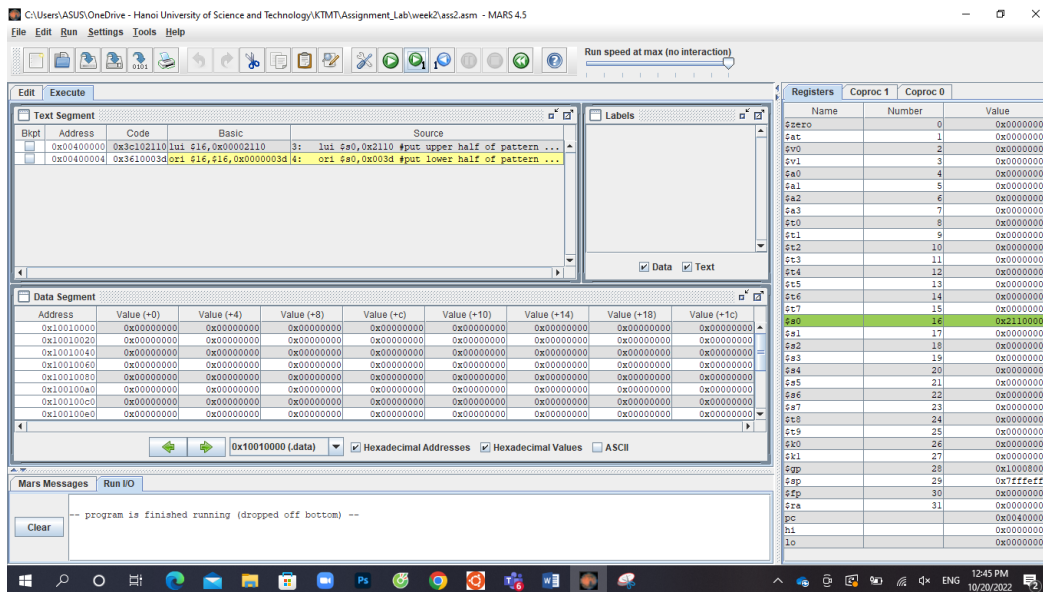
Assemble: operation completed successfully.

Registers

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffcfc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400000
hi		0x00000000
lo		0x00000000

\$s0	16	0x00000000
pc		0x00400000

Step 1:



\$s0	16	0x21100000
pc		0x00400004

Step 2:

\$s0	16	0x2110003d
pc		0x00400008

Step 3:

\$s0	16	0x2110003d
pc		0x00400008

2 lệnh `lui` và `ori` sẽ giúp cộng một số 32 bit dần dần vào biến `$s0` bằng cách cộng 16bit đầu rồi cộng tiếp 16 bit cuối.

- Ở cửa sổ Data Segment, sự thay đổi của các byte trong cung lệnh .text giống với cột Code trong cửa sổ Text Segment.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00400000	0x3c102110	0x3610003d	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00400020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00400040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00400060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00400080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x004000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x004000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x004000e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

Assignment 3:

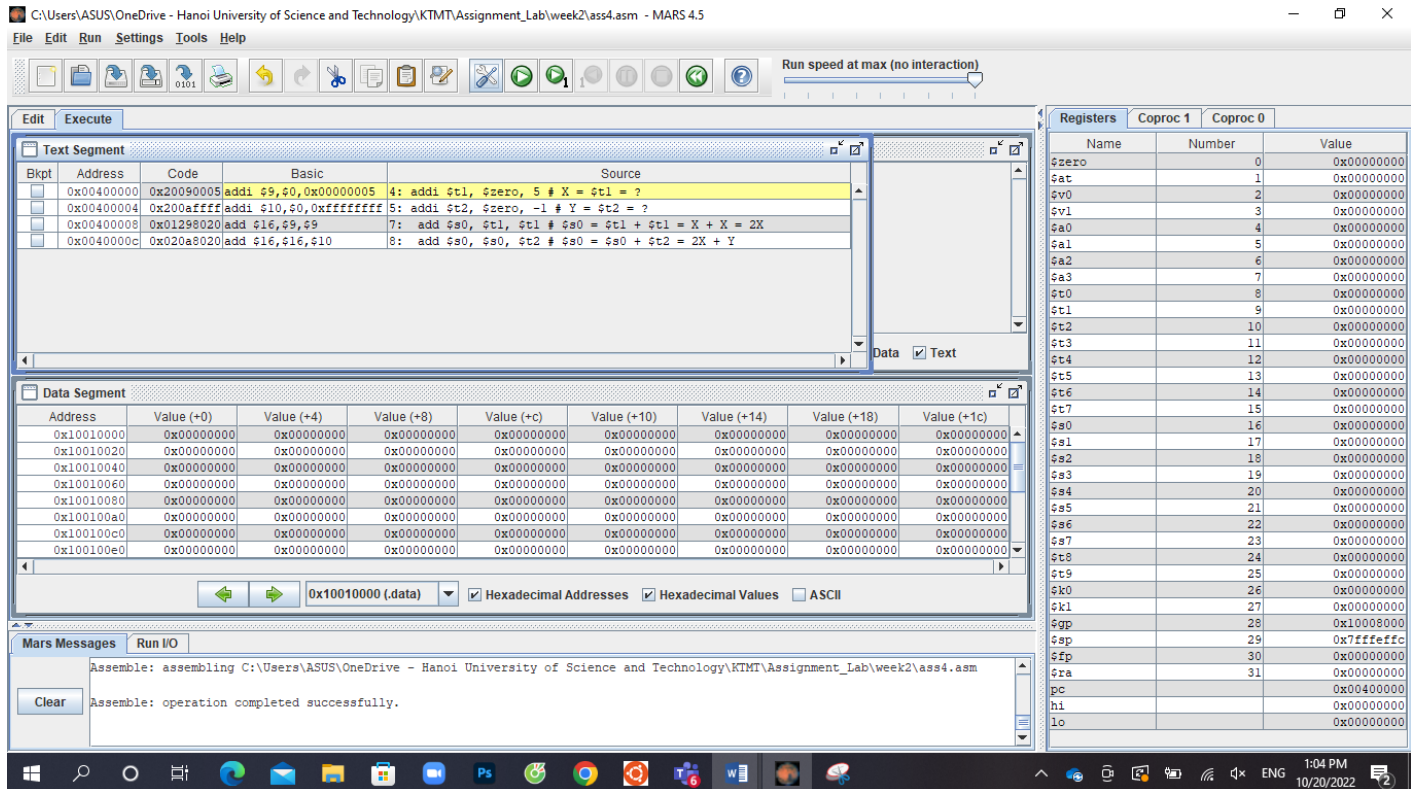
Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c012110	lui \$1, 0x00002110	3: li \$s0, 0x2110003d #pseudo instruction=2 basic instructions
	0x00400004	0x3430003d	ori \$16, \$1, 0x0000003d	
	0x00400008	0x24110002	addiu \$17, \$0, 0x0000...	4: li \$s1, 0x2 #but if the immediate value is small, one ins

Li là lệnh gán ngay lập tức 1 số 16bit hoặc 32 bit vào biến .

Ở câu lệnh li đầu tiên, lệnh li muốn gán số 32bit 0x2110003d vào biến \$s0 nên đã tách ra làm 2 lệnh là lui và ori.

Còn lệnh thứ 2 là li \$s1, 0x2 là chỉ gán 1 số <32bit nên nó sẽ gán luôn và tương đương với addiu \$s1, \$s0, 0x2

Assignment 4:



Khi debug, gán $\$t1 = 5$, $\$t2 = -1$, thì kết quả là

- Ở lần add đầu tiên, tương ứng với 10 và lần add thứ hai là tương ứng với 9
- \Rightarrow Kết quả đúng: $5 * 2 - 1 = 9$

Addi $\$t2, \$zero, -1$

0x200affff : 0010 0000 0000 1010 1111 1111 1111 1111

001000 00000 01010 1111111111111111

Op: addi

Rs: \$0

Rt: 10 ($\$t2$)

Imm: -1

Trùng với khuôn mẫu kiểu lệnh I

Add $\$s0, \$t1, \$t1$: 0x01298020

0000 0001 0010 1001 1000 0000 0010 0000
000000 01001 01001 10000 00000 100000

Op: R

Rs: 9 (\$t1)

Rt: 9 (\$t1)

Rd: 16 (\$s0)

Shift amount: 0

Function: 32 (add)

Trùng với khuôn mẫu kiểu lệnh R

Assignment 5:

Text Segment				
Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x20090004	addi \$9,\$0,0x00000004	4: addi \$t1, \$zero, 4 # X = \$t1 = ?
<input type="checkbox"/>	0x00400004	0x200a0005	addi \$10,\$0,0x00000005	5: addi \$t2, \$zero, 5 # Y = \$t2 = ?
<input type="checkbox"/>	0x00400008	0x712a8002	mul \$16,\$9,\$10	7: mul \$s0, \$t1, \$t2 # HI-LO = \$t1 * \$t2 = X * Y ; \$s0 = LO
<input type="checkbox"/>	0x0040000c	0x20010003	addi \$1,\$0,0x00000003	8: mul \$s0, \$s0, 3 # \$s0 = \$s0 * 3 = 3 * X * Y
<input type="checkbox"/>	0x00400010	0x72018002	mul \$16,\$16,\$1	
<input type="checkbox"/>	0x00400014	0x00008812	mflo \$17	10: mflo \$s1

2 lệnh addi để gán giá trị cho 2 biến t1 và t2

Thao tác nhân có kết quả chứa hai thanh ghi HI <lưu 32 bit sau> và LO<lưu 32 bit đầu>

Điều bất thường là trong lệnh mul, 3 không phải là 1 biến nên khi debug xuất hiện biến at lưu giá trị của 3

Text Segment				
Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x20090004	addi \$9,\$0,0x00000004	4: addi \$t1, \$zero, 4 # X = \$t1 = ?
<input type="checkbox"/>	0x00400004	0x200a0005	addi \$10,\$0,0x00000005	5: addi \$t2, \$zero, 5 # Y = \$t2 = ?
<input type="checkbox"/>	0x00400008	0x712a8002	mul \$16,\$9,\$10	7: mul \$s0, \$t1, \$t2 # HI-LO = \$t1 * \$t2 = X * Y ; \$s0 = LO
<input type="checkbox"/>	0x0040000c	0x20010003	addi \$1,\$0,0x00000003	8: mul \$s0, \$s0, 3 # \$s0 = \$s0 * 3 = 3 * X * Y
<input type="checkbox"/>	0x00400010	0x72018002	mul \$16,\$16,\$1	
<input type="checkbox"/>	0x00400014	0x00008812	mflo \$17	10: mflo \$s1

Registers			Coproc 1	Coproc 0
Name	Number	Value		
\$zero	0	0x00000000		
\$at	1	0x00000003		
\$v0	2	0x00000000		
\$v1	3	0x00000000		
\$a0	4	0x00000000		
\$a1	5	0x00000000		
\$a2	6	0x00000000		
\$a3	7	0x00000000		
\$t0	8	0x00000000		
\$t1	9	0x00000004		
\$t2	10	0x00000005		
\$t3	11	0x00000000		
\$t4	12	0x00000000		
\$t5	13	0x00000000		

Lệnh mflo giúp sao lưu giá trị của biến s0 sang biến s1.

C:\Users\ASUS\OneDrive - Hanoi University of Science and Technology\KTM\Assignment_Lab\week2\ass5.asm - MARS 4.5

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x20090004	addi \$s9,\$0,0x00000004	4: addi \$t1, \$zero, 4 # X = \$t1 = 2
	0x00400004	0x200a0005	addi \$t0,\$0,0x00000005	5: addi \$t2, \$zero, 5 # Y = \$t2 = ?
	0x00400008	0x712a8002	mul \$t6,\$9,\$t0	7: mul \$s0, \$t1, \$t2 # HI-LO = \$t1 * \$t2 = X * Y ; \$s0 = LO
	0x0040000c	0x20010003	addi \$t1,\$0,0x00000003	8: mul \$s0, \$s0, 3 # \$s0 = \$s0 * 3 = 3 * X * Y
	0x00400010	0x72018002	mul \$t6,\$t6,\$t1	
	0x00400014	0x00008812	mflo \$s1	10: mflo \$s1

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

0x10010000 (.data) Hexadecimal Addresses Hexadecimal Values ASCII

Mars Messages Run I/O

Clear -- program is finished running (dropped off bottom) --

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000003
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000004
\$t2	10	0x00000005
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x0000003c
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10008000
\$sp	29	0x7ffffeffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400014
hi		0x00000000
lo		0x0000003c

1:38 PM 10/20/2022

Assignment 6:

Text Segment					
Bkpt	Address	Code	Basic	Source	Address
<input type="checkbox"/>	0x00400000	0x3c011001	lui \$1,0x00001001	8: la \$t8, X # Get the address of X in Data Segment	
<input type="checkbox"/>	0x00400004	0x34380000	ori \$24,\$1,0x00000000		0x10010000
<input type="checkbox"/>	0x00400008	0x3c011001	lui \$1,0x00001001	9: la \$t9, Y # Get the address of Y in Data Segment	0x10010004
<input type="checkbox"/>	0x0040000c	0x34390004	ori \$25,\$1,0x00000004		0x10010008
<input type="checkbox"/>	0x00400010	0x8f090000	lw \$9,0x00000000(\$24)	10: lw \$t1, 0(\$t8) # \$t1 = X	
<input type="checkbox"/>	0x00400014	0x8f2a0000	lw \$10,0x00000000(\$25)	11: lw \$t2, 0(\$t9) # \$t2 = Y	
<input type="checkbox"/>	0x00400018	0x01298020	add \$16,\$9,\$9	13: add \$s0, \$t1, \$t1 # \$s0 = \$t1 + \$t1 = X + X = 2X	
<input type="checkbox"/>	0x0040001c	0x020a8020	add \$16,\$16,\$10	14: add \$s0, \$s0, \$t2 # \$s0 = \$s0 + \$t2 = 2X + Y	
<input type="checkbox"/>	0x00400020	0x3c011001	lui \$1,0x00001001	16: la \$t7, Z # Get the address of Z in Data Segment	
<input type="checkbox"/>	0x00400024	0x342f0008	ori \$15,\$1,0x00000008		

- Lệnh la được biên dịch thành 2 lệnh là lui và ori nhằm lưu địa chỉ của các biến X,Y vào biến t8,t9.
- Giá trị của X và Y tương ứng vs giá trị khởi tạo, được hiện thị trong Data Segment lần lượt là 0x00000005 và 0xffffffff
- Ở cửa sổ Register,

Câu lệnh lw lưu giá trị của biến X vào biến t1 bằng việc trở tới địa chỉ của X qua 0(\$t8).

Câu lệnh sw lưu giá trị của thanh ghi s0 vào biến có địa chỉ t7 (là Z).