

THỰC HÀNH KIẾN TRÚC MÁY TÍNH

IT-3280

Họ tên	Thân Cát Ngọc Lan
MSSV	20225646

Assignment 1:

- Đặt giá trị thanh ghi \$s1 = 23 , \$s2 = 25
- Chạy lệnh:

Bước chạy	Thanh ghi	Giá trị ban đầu	Giá trị sau
1,2	\$s1, \$s2	0x00000000	0x00000017,0x00000019
3	\$t0	0x00000000	0x00000000
4	\$s3	0x00000000	0x00000030
5	\$t1	0x00000000	0x0000000e
7	\$t2	0x00000000	0x00000000

The screenshot displays the MARS MIPS simulator interface. The main window shows assembly code with columns for Address, Code, Basic, and Source. The code includes instructions like `addi $s1,$s1,23`, `addi $s2,$s2,25`, `li $t0,0`, `addu $s3,$s1,$s2`, `xor $s3,$s1,$s2`, `bltz $t1,EXIT`, `slt $t2,$s3,$s1`, `bltz $s1,NEGATIVE`, `beq $t1,$zero,EXIT`, `bne $t2,$zero,EXIT`, and `addiu $s8,$s0,0x00000019`. The Labels window on the right lists labels like `start`, `NEGATIVE`, `OVERFLOW`, and `EXIT` with their corresponding addresses. The Registers window on the right shows the state of registers, with `$s1` at 0x00000017 and `$s2` at 0x00000019. The Data Segment window at the bottom shows memory addresses from 0x10010000 to 0x10010120, all containing 0x00000000.

Assignment 2:

```

#Laboratory Exercise 4, Home Assignment 2
.text
li $s0, 0x12345678 #load test value for these function
#Extract the MSB of $s0
andi $t0, $s0, 0xff000000
srl $t0, $t0, 24
#clear LSB
andi $t1, $s0, 0xfffffff0
# Set LSB of $s0 (bits 7 to 0 are set to 1)
ori $t2, $s0, 0x000000ff
#clear $s0
andi $s0, $s0, 0x00000000

```

Page	Address	Code	Basic	Source
0x00400000	0x3e01234	lui \$t, 0x00001234	3: li \$s0, 0x12345678 #load test value for these function	
0x00400004	0x3405678	ori \$t6, \$t, 0x00005678		
0x00400008	0x3e01fff0	lui \$t, 0xffffffff	5: andi \$t0, \$s0, 0xff000000	
0x0040000c	0x3421000	ori \$t, \$t, 0x00000000		
0x00400010	0x0201400	and \$s, \$t6, \$t		
0x00400014	0x0008402	srl \$t0, \$t0, 24	6: srl \$t0, \$t0, 24	
0x00400018	0x3e01fff0	lui \$t, 0xffffffff	8: andi \$t1, \$s0, 0xfffffff0	
0x0040001c	0x3421fff0	ori \$t, \$t, 0x0000fff0		
0x00400020	0x0201400	and \$s, \$t6, \$t		
0x00400024	0x360a0fff	ori \$t0, \$t0, 0x000000ff	10: ori \$t2, \$s0, 0x000000ff	
0x00400028	0x3210000	andi \$t6, \$t6, 0x00000012	12: andi \$s0, \$s0, 0x00000000	

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0xffffffff
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000012
\$t1	9	0x12345600
\$t2	10	0x123456ff
\$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
\$s8	24	0x00000000
\$s9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10000000
\$sp	29	0x7fffffc0
\$fp	30	0x00000000
\$ra	31	0x0040002c
\$s		
\$t		
\$a		
\$v		

Assignment 3:

a. abs

```

imp30.asm
1 .text
2 li $s1, 0xffffffff
3 add $s0, $zero, $s1 # copy s1 into s2
4 slt $t0, $s1, $zero # s1 < 0 ?
5 beq $t0, $zero, EXIT # if s1 > 0, s0 = |s1| = s1
6 sub $s0, $zero, $s1 # if s1 < 0, s0 = |s1| = -s1
7 EXIT:

```

The screenshot shows the Mars MIPS simulator interface. The main window displays assembly code for a program named `mips5.asm`. The code includes instructions like `li $s1, 0xffffffff`, `add $s0, $zero, $s1`, `slt $t0, $s1, $zero`, `beq $t0, $zero, EXIT`, and `sub $s0, $zero, $s1`. The `Labels` window shows the `EXIT` label at address `0x00400014`. The `Registers` window on the right shows the state of all MIPS registers, with `$s0` and `$s1` both containing `0xffffffff`.

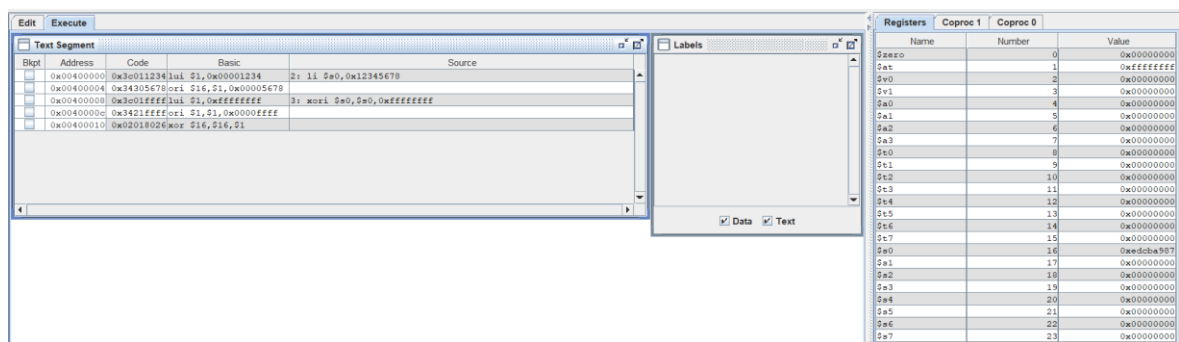
b. move

The screenshot shows the Mars MIPS simulator interface with the assembly code for a program named `mips5.asm`. The code includes instructions like `.text`, `li $s1, 0x12345678`, and `addu $s0, $0, $s1`. The `Registers` window on the right shows the state of all MIPS registers, with `$s1` containing `0x12345678` and `$s0` containing `0x00000000`.

The screenshot shows the Mars MIPS simulator interface with the assembly code for a program named `mips5.asm`. The code includes instructions like `.text`, `li $s1, 0x12345678`, and `addu $s0, $0, $s1`. The `Registers` window on the right shows the state of all MIPS registers, with `$s1` containing `0x12345678` and `$s0` containing `0x12345678`.

c. not (đảo bit)

The screenshot shows the Mars MIPS simulator interface with the assembly code for a program named `mips5.asm`. The code includes instructions like `.text`, `li $s0, 0x12345678`, and `xori $s0, $s0, 0xffffffff`. The `Registers` window on the right shows the state of all MIPS registers, with `$s0` containing `0x12345678`.

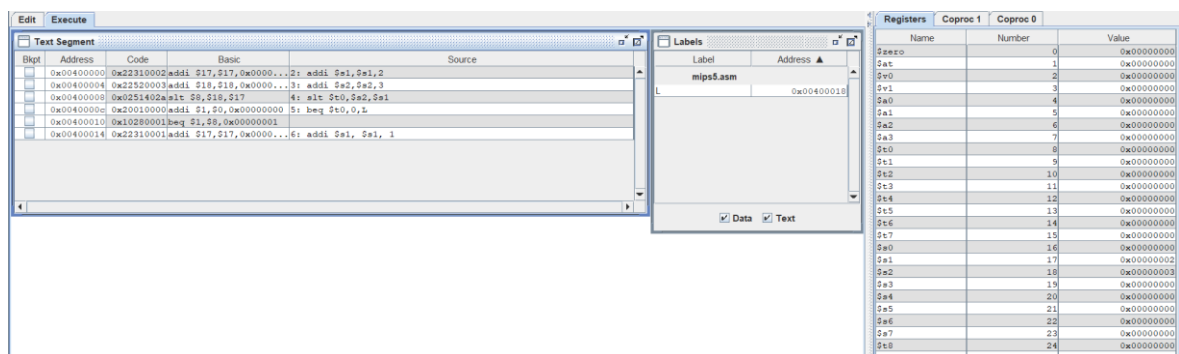


d. $\$s1 \leq \$s2$

```

mips5.asm
1  .text
2  addi $s1, $s1, 2
3  addi $s2, $s2, 3
4  slt $t0, $s2, $s1
5  beq $t0, 0, L
6  addi $s1, $s1, 1
7  L:

```



Assignment 4:

Text Segment

Bkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x24110004	addiu \$17,\$0,4	3: li \$a1, 4
<input type="checkbox"/>	0x00400004	0x24120030	addiu \$18,\$0,32	4: li \$a2, 32
<input type="checkbox"/>	0x00400008	0x24130003	addiu \$16,\$0,1	5: li \$a0, 1
<input type="checkbox"/>	0x0040000c	0x12500003	beq \$18,\$16,9	7: beq \$a2,\$a0,EXIT
<input type="checkbox"/>	0x00400010	0x00129042	srl \$18,\$18,1	8: srl \$a2,\$a2,1
<input type="checkbox"/>	0x00400014	0x00118840	sl \$17,\$17,1	9: sl \$a1,\$a1,1 # ket qua phiep nhan
<input type="checkbox"/>	0x00400018	0x08100003	0x0040000c	10: j loop

Labels

Label	Address
mips5.asm	
loop	0x0040000c
EXIT	0x0040001c

Data Segment

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

Registers

Name	Coproc 1	Coproc 0	Value
\$zero	0		0
\$a0	1		0
\$a1	2		0
\$a2	3		0
\$a3	4		0
\$a4	5		0
\$a5	6		0
\$a6	7		0
\$a7	8		0
\$t0	9		0
\$t1	10		0
\$t2	11		0
\$t3	12		0
\$t4	13		0
\$t5	14		0
\$t6	15		0
\$t7	16		1
\$s0	17		128
\$s1	18		0
\$s2	19		0
\$s3	20		0
\$s4	21		0
\$s5	22		0
\$s6	23		0
\$s7	24		0