

Bài thực hành số 4

Lớp: 139365 – Học phần: Thực hành Kiến Trúc Máy Tính

Họ và tên: Đinh Thị Hồng Phúc

MSSV: 20215118

Bài 1.

```
1 #Assignment 1
2 #kiểm tra tràn số trong phép cộng s1 + s2
3 .text
4 start:
5     li $s1, 0xffffffff
6     li $s2, 0x80000000
7
8     li $t0, 0           #No Overflow is default status, khởi tạo t0 = 0
9     addu $s3, $s1, $s2  #s3 = s1 + s2
10    xor $t1, $s1, $s2    #Test if s1 and s2 have the same sign (có cùng dấu không, lưu kết quả vào t1)
11
12    bltz $t1, EXIT       #If not, exit (không cùng dấu -> thoát chương trình) nếu t1<0
13    slt $t2, $s3, $s1    #s3<s1 -> t2=1 (thì s1 là số âm)
14    bltz $s1, NEGATIVE  #Test if s1 and s2 is negative?
15    beq $t2, $zero, EXIT #s1 and s2 are positive (?nếu t2 = 0 thì exit)
16    #if s3 > s1 then the result is not overflow
17    j OVERFLOW
18 NEGATIVE:
19    bne $t2, $zero, EXIT #s1 and s2 are negative (nếu t2 # 0 thì exit)
20    #if s3 < s1 then the result is not overflow
21 OVERFLOW:
22    li $t0, 1           #the result is overflow
23 EXIT:
```

Thực hiện gõ chương trình vào công cụ MARS

- Thực hiện dòng lệnh 5 và 6: Các thanh ghi \$s1 và \$s2 được gán với các giá trị lần lượt là 0xffffffff và 0x80000000
- Thực hiện dòng lệnh số 8: Thanh ghi \$t0 được gán giá trị 0 (tương ứng là không bị tràn số)
- Thực hiện dòng lệnh số 9: Dòng lệnh thực hiện ghi giá trị của: $s3 = s1 + s2$
- Thực hiện dòng lệnh số 10: Dòng lệnh thực hiện: $s1 \wedge s2$, lưu giá trị vào t1, kiểm tra s1 có cùng dấu với s2 không
- Thực hiện dòng lệnh số 12: Nếu $t1 < 0$ thì 2 số không cùng dấu, thoát chương trình (EXIT)
- Thực hiện dòng lệnh số 13, 15: So sánh s3 có nhỏ hơn s1 hay không (tổng nhỏ hơn số hạng đối với số dương)
- Thực hiện dòng lệnh số 14: Kiểm tra s1, s2 có là số âm hay không, nếu là số âm thì tổng nhỏ hơn số hạng thì EXIT không thì OVERFLOW
- Thực hiện dòng lệnh số 17: chuyển đến phần OVERFLOW

- Thực hiện dòng lệnh số 19: Nếu $t2 \neq 0$ thì EXIT
- Thực hiện dòng lệnh số 22: gán $t0 = 1$ (trần số)

Byte	Address	Code	Basic	Source
	0x00400000	0x241fffff	addiu \$t0, \$0, 0xffff...	5: li \$s1, 0xffffffff
	0x00400004	0x3c018000	lui \$t1, 0xfffff000	6: li \$s2, 0x80000000
	0x00400008	0x34320000	ori \$t8, \$t1, 0x00000000	
	0x0040000c	0x24080000	addiu \$s, \$0, 0x00000000	8: li \$t0, 0
	0x00400010	0x0232821	addu \$s3, \$s1, \$s2	9: addu \$s3, \$s1, \$s2
	0x00400014	0x0032482e	xor \$t1, \$s1, \$s2	10: xor \$t1, \$s1, \$s2
	0x00400018	0x0520000e	bltz \$s, 0x0000000e	12: bltz \$t1, EXIT
	0x0040001c	0x0271502a	sllt \$t2, \$s3, \$s1	13: sllt \$t2, \$s3, \$s1
	0x00400020	0x04200002	bltz \$s1, 0x00000002	14: bltz \$s1, NEGATIVE
	0x00400024	0x1140001b	beg \$t2, \$zero, EXIT	15: beg \$t2, \$zero, EXIT
	0x00400028	0x08100003	j OVERFLOW	17: j OVERFLOW
	0x0040002c	0x1540001b	bne \$t2, \$zero, EXIT	19: bne \$t2, \$zero, EXIT
	0x00400030	0x24080001	addiu \$s2, \$0, 0x00000001	22: li \$t0, 1

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
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

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	0x00000001
\$t1	9	0x7fffffff
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0xffffffff
\$s2	18	0x80000000
\$s3	19	0x7fffffff
\$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	0x7ffffefc
\$fp	30	0x00000000
\$ra	31	0x00000000
\$pc		0x00400034
\$hi		0x00000000
\$lo		0x00000000

Thực hiện chạy chương trình với MARS

Bài 2.

```

1  #Assignment 2
2  #trích xuất các bit cụ thể từ giá trị lưu trong thanh ghi s0
3  .text
4      li $s0, 0x56342131          #load test value for these function -> sử dụng cho việc kiểm tra các bit
5      addi $t0, $s0, 0xffff00000  #extract MSB of s0
6      andi $t1, $s0, 0xffffffff00 #clear LSB of s0
7      ori $t2, $s0, 0xff          #set LSB of s0
8      xor $t3, $s0, $s0          #clear s0

```

Thực hiện gõ chương trình vào công cụ MARS

- Dòng 4: Thanh ghi s0 được gán giá trị 0x56342131
- Dòng 5: Lấy giá trị MSB của s0 bằng phép and với 0xffff00000. Khi đó t0 = 0x55342131
- Dòng 6: Xóa giá trị LSB của s0 bằng phép and với 0xffffffff00. Khi đó t1 = 0x56342100
- Dòng 7: Set LSB của s0 bằng phép or với 0xff. Khi đó t2 = 0x563421ff
- Dòng 8: Xóa giá trị s0 bằng cách xor s0 với chính nó. Khi đó t3 = 0x00000000

Offset	Address	Code	Basic	Source
0x04000000	0x3c015634	lui \$1, 0x0005634	4:	li \$a0, 0x56342131 #load test value for these function -> sử dụng cho việc kiểm tra...
0x04000004	0x34302131	ori \$16, \$1, 0x0002131		
0x04000008	0x3c01ff00	lui \$1, 0xffffff00	5:	addi \$t0, \$a0, 0xffff0000 #extract MSB of a0
0x0400000c	0x34210000	ori \$1, \$1, 0x00000000		
0x04000010	0x02014020	add \$9, \$16, \$1		
0x04000014	0x3c01ffff	lui \$1, 0xfffffff	6:	andi \$t1, \$a0, 0xfffffff #clear LSB of a0
0x04000018	0x3421ff00	ori \$1, \$1, 0x0000ff00		
0x0400001c	0x02014024	and \$9, \$16, \$1		
0x04000020	0x360a00ff	ori \$10, \$16, 0x000000ff	7:	ori \$t2, \$a0, 0xff #set LSB of a0
0x04000024	0x02105826	xor \$11, \$16, \$16	8:	xor \$t3, \$a0, \$a0 #clear a0

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	0x55342131
\$t1	9	0x56342100
\$t2	10	0x563421ff
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x56342131
\$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
\$t0	26	0x00000000
\$t1	27	0x00000000
\$fp	28	0x10008000
\$sp	29	0xffffffc
\$f0	30	0x00000000
\$f1	31	0x00000000
\$f2		0x00000000
\$f3		0x00000000
\$f4		0x00000000
\$f5		0x00000000
\$f6		0x00000000
\$f7		0x00000000
\$f8		0x00000000
\$f9		0x00000000
\$f10		0x00000000
\$f11		0x00000000
\$f12		0x00000000
\$f13		0x00000000
\$f14		0x00000000
\$f15		0x00000000
\$f16		0x00000000
\$f17		0x00000000
\$f18		0x00000000
\$f19		0x00000000
\$f20		0x00000000
\$f21		0x00000000
\$f22		0x00000000
\$f23		0x00000000
\$f24		0x00000000
\$f25		0x00000000
\$f26		0x00000000
\$f27		0x00000000
\$f28		0x00000000
\$f29		0x00000000
\$f30		0x00000000
\$f31		0x00000000

Thực hiện chạy chương trình với MARS

Bài 3.

a. abs

```

1  #Assignment 3a
2  .text
3      li $s1, 5
4  start:
5      slt $t0, $s1, $zero
6      bne $t0, $zero, else
7      add $s0, $s1, $zero
8      j endif
9  else:
10     sub $s0, $zero, $s1
11  endif:

```

Thực hiện gõ chương trình vào công cụ MARS

- Dòng 3: Thanh ghi s1 được gán với giá trị 5
- Dòng 5: So sánh s1 với 0, gán kết quả vào t0 (kiểm tra s1 là số âm hay dương)
- Dòng 6: Nếu t0 ≠ 0 thì chuyển đến else
- Dòng 7: Thực hiện s0 = s1
- Dòng 8: Thực hiện chuyển đến endif
- Dòng 10: Thực hiện s0 = 0 – s1

Bkpt	Address	Code	Basic	Source
0x00400000	0x24110005	addiu \$17,\$0,0x0000...	3:	li \$s1, 5
0x00400004	0x2204002a	slt \$0,\$s1,\$0	5:	slt \$t0, \$s1, \$zero
0x00400008	0x15000002	bne \$0,\$0,0x00000002	6:	bne \$t0, \$zero, else
0x0040000c	0x22000020	add \$16,\$17,\$0	7:	add \$s0, \$s1, \$zero
0x00400010	0x00100006	j 0x00400018	8:	j endif
0x00400014	0x00118022	sub \$s0,\$0,\$17	10:	sub \$s0, \$zero, \$s1

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
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

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	0x00000005
\$s1	17	0x00000005
\$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	0xffffffff
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400018
hi		0x00000000
lo		0x00000000

Thực hiện chạy chương trình với MARS

b. move

```

1  #Assignment 3b
2  .text
3      li $s1, 2
4      add $s0, $s1, $zero

```

Thực hiện gõ chương trình vào công cụ MARS

- Dòng 3: Thanh ghi s1 được gán giá trị 2
- Dòng 4: Thực hiện phép toán s0 = s1

Bkpt	Address	Code	Basic	Source
0x00400000	0x24110002	addiu \$17,\$0,0x0000...	3:	li \$s1, 2
0x00400004	0x22050020	add \$16,\$17,\$0	4:	add \$s0, \$s1, \$zero

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
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

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	0x00000002
\$s1	17	0x00000002
\$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	0xffffffff
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400018
hi		0x00000000
lo		0x00000000

Thực hiện chạy chương trình với MARS

c. not

```

1  #Assignment 3c
2  .text
3      li $s0, 6
4      xori $t0, $s0, 0xffffffff

```

Thực hiện gõ chương trình vào công cụ MARS

- Dòng 3: Thanh ghi s0 được gán giá trị 6

- Dòng 4: Thực hiện đảo dấu s0 bằng phép xor với 0xffffffff. Khi đó t0 = 0xffffffff9

The screenshot shows the MARS MIPS simulator. The assembly window displays the following code:

```

1: li $s0, 6
2: xori $t0, $s0, 0xffffffff
3:
4:
5:
6:

```

The memory dump window shows the state of registers and memory. The register file is as follows:

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	0xffffffff9
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000006
\$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
\$k1	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10000000
\$fp	29	0xffffffff
\$fp	30	0x00000000
\$ra	31	0x00000000
PC		0x00400010
hi		0x00000000
lo		0x00000000

Thực hiện chạy chương trình với MARS

d. ble

```

1: #Assignment 3d
2: .text
3:     li $s0, 2
4:     li $s1, 3
5:     slt $t0, $s0, $s1      #s0<s1 -> t0=1
6:     bne $t0, $zero, L
7: L:

```

Thực hiện gõ chương trình vào công cụ MARS

- Dòng 3, 4: Thanh ghi s0, s1 được gán giá trị lần lượt là 2 và 3
- Dòng 5: So sánh s0 có nhỏ hơn s1 hay không cho kết quả vào t0
- Dòng 6: Nếu t0 ≠ 0 (s0 ≤ s1) thì chuyển sang L

The screenshot shows the MARS MIPS simulator. The assembly window displays the following code:

```

1: li $s0, 2
2: li $s1, 3
3: slt $t0, $s0, $s1      #s0<s1 -> t0=1
4: bne $t0, $zero, L
5:
6:

```

The memory dump window shows the state of registers and memory. The register file is as follows:

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	0x00000001
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000002
\$s1	17	0x00000003
\$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	0x10000000
\$fp	29	0xffffffff
\$fp	30	0x00000000
\$ra	31	0x00000000
PC		0x00400010
hi		0x00000000
lo		0x00000000

Thực hiện chạy chương trình với MARS

Bài 4.

```

1  #Assignment 4
2  .text
3      li $s1, 0xffffffff
4      li $s2, 0x80000000
5  start:
6      li $t0, 0
7      addu $s3, $s1, $s2
8      xor $t1, $s1, $s2
9      bltz $t1, EXIT
10
11     xor $t2, $s1, $s3
12     bltz $t2, OVERFLOW
13     j EXIT
14 OVERFLOW:
15     li $t0, 1
16 EXIT:

```

Thực hiện gõ chương trình vào công cụ MARS

- Dòng 3, 4: Thanh ghi s1, s2 được gán giá trị lần lượt là 0xffffffff và 0x80000000
- Dòng 6: Thanh ghi t0 được gán giá trị 0 (không tràn số)
- Dòng 7: Thực hiện phép toán $s3 = s1 + s2$
- Dòng 8: Thực hiện phép toán $s1 \wedge s2$ gán kết quả vào t1
- Dòng 9: Nếu $t1 < 0$ ($s1, s2$ trái dấu) thì EXIT (không tràn số)
- Dòng 11: Thực hiện phép toán $s1 \wedge s3$ gán kết quả vào t2
- Dòng 12: Nếu $t2 < 0$ ($s1, s3$ trái dấu) thì OVERFLOW
- Dòng 13: chuyển đến phần EXIT
- Dòng 15: Thanh ghi t0 được gán giá trị 1 (tràn số)

Bkpt	Address	Code	Basic	Source
	0x00400000	0x2411ffff	addiu \$t1,\$0,0xffff...	3: li \$a1, 0xffffffff
	0x00400004	0x3c010000	lui \$t1,0xffff8000	4: li \$a2, 0x80000000
	0x00400008	0x34320000	ori \$t0,\$0,0x00000000	6: li \$t0, 0
	0x0040000c	0x40800000	addiu \$s3,\$0,0x00000000	7: addu \$s3, \$s1, \$s2
	0x00400010	0x02329f21	addu \$t0,\$t1,\$t1	8: xor \$t1, \$s1, \$s2
	0x00400014	0x02324e26	xor \$t1,\$t1,\$t1	9: bltz \$t1, EXIT
	0x00400018	0x05200004	bltz \$t1,0x00000004	11: xor \$t2, \$s1, \$s3
	0x0040001c	0x02335026	xor \$t0,\$t0,\$t0	12: bltz \$t2, OVERFLOW
	0x00400020	0x05400001	bltz \$t0,0x00000001	13: j EXIT
	0x00400024	0x0510000b	j EXIT	15: li \$t0, 1
	0x00400028	0x24080001	addiu \$t0,\$0,0x00000001	

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
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

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	0x00000001
\$t1	9	0xffffffff
\$t2	10	0x80000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$w0	16	0x00000000
\$a1	17	0xffffffff
\$a2	18	0x80000000
\$a3	19	0xffffffff
\$a4	20	0x00000000
\$a5	21	0x00000000
\$a6	22	0x00000000
\$a7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$t0	26	0x00000000
\$t1	27	0x00000000
\$gp	28	0x10000000
\$fp	29	0xffffffff
\$fp	30	0x00000000
\$ra	31	0x00000000
\$c		0x0040002c
\$h1		0x00000000
\$t0		0x00000000

Thực hiện chạy chương trình với MARS

Bài 5.

```
1  #Assignment 5
2  .text
3      li $s0, 2
4      li $s1, -1
5      li $s2, 1
6      li $s3, 5
7  loop:
8      add $s1, $s1, $s2
9      sllv $s4, $s0, $s1
10     slt $t0, $s1, $s3
11     bne $t0, $zero, loop
```

Thực hiện gõ chương trình vào công cụ **MARS**

- Dòng 3, 4, 5, 6: Thanh ghi s0, s1, s2, s3 được gán giá trị lần lượt 2, -1, 1, 10 (s0: giá trị ban đầu, s1 = i = -1, s2 = step = 1, s3 = n = 10)
- Dòng 8: Thực hiện phép toán $s1 = s1 + s2$ ($i = i + \text{step}$)
- Dòng 9: Thực hiện phép dịch bit sang trái: $s4 = s0 * 2^{s1}$
- Dòng 10, 11: Kiểm tra s1 (i) đã đếm đến s3 (n = 5) hay chưa. Nếu chưa thì tiếp tục vòng lặp

The screenshot displays the MARS MIPS simulator interface. The main window shows the assembly code from the previous block. Below the code, the 'Data Segment' is visible, showing memory addresses and their corresponding values. To the right, a table lists the registers and their current values.

Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	0
\$t2	10	0
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	2
\$s1	17	-1
\$s2	18	1
\$s3	19	5
\$s4	20	64
\$s5	21	0
\$s6	22	0
\$s7	23	0
\$t8	24	0
\$t9	25	0
\$k0	26	0
\$k1	27	0
\$gp	28	268468224
\$sp	29	2147479548
\$fp	30	0
\$ra	31	0
pc		4194336
hi		0
lo		0

The 'Mars Messages' window at the bottom shows the message: "-- program is finished running (dropped off bottom) --".

Thực hiện chạy chương trình với **MARS**