Homework 1

Alex Palumbo
CISC 340 Computer Architecture

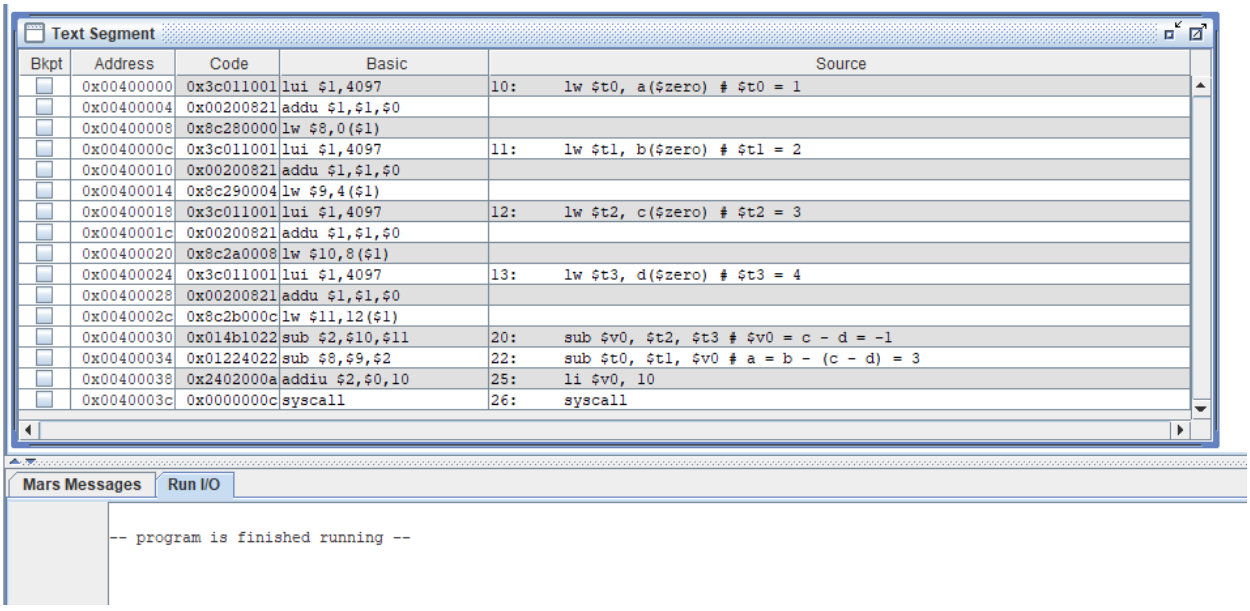
March 16, 2020

1

MIPS instructions that compute $a = b - (c–d)$

```
1  .data
2      a: .word 1
3      b: .word 2
4      c: .word 3
5      d: .word 4
6  .text
7      lw $t0, a($zero)
8      lw $t1, b($zero)
9      lw $t2, c($zero)
10     lw $t3, d($zero)
11     # Problem 1 specific:
12     sub $v0, $t2, $t3
13     sub $t0, $t1, $v0
```

Successful execution in MARS:

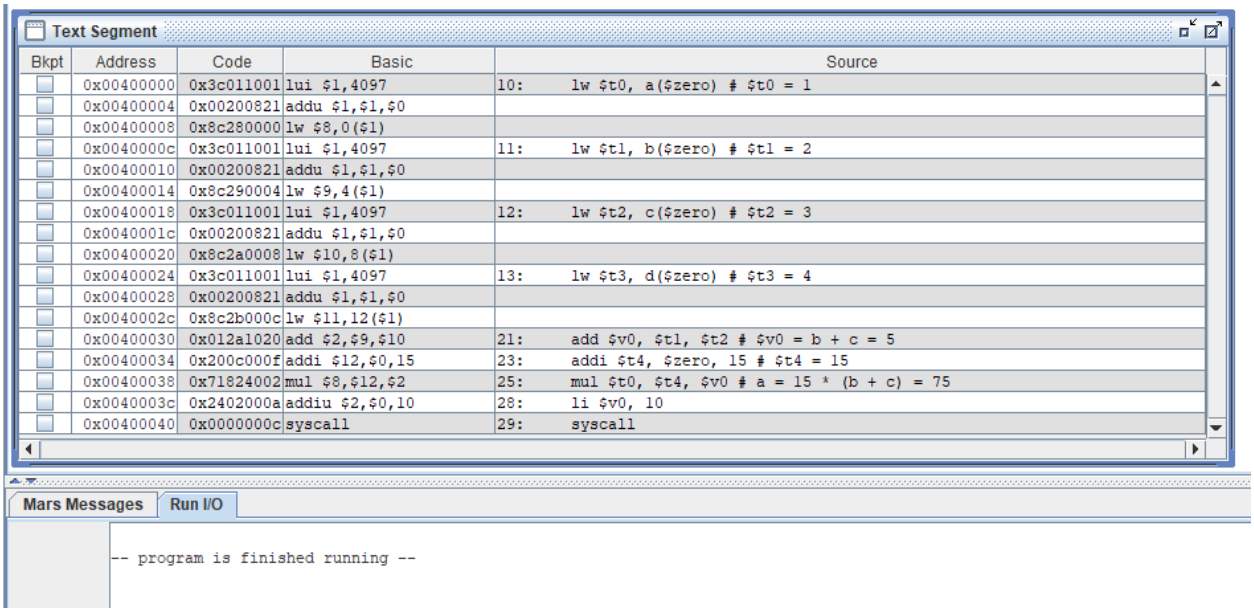


2

MIPS instructions that compute $a = 15 * (b + c)$

```
1  .data
2      a: .word 1
3      b: .word 2
4      c: .word 3
5      d: .word 4
6  .text
7      lw $t0, a($zero)
8      lw $t1, b($zero)
9      lw $t2, c($zero)
10     lw $t3, d($zero)
11     # Problem 2 specific:
12     add $v0, $t1, $t2
13     addi $t4, $zero, 15
14     mul $t0, $t4, $v0
```

I couldn't find a way to multiply the expression without using the mul instruction. So I used the mul instruction to do the multiplication with three total instructions.
Successful execution in MARS:



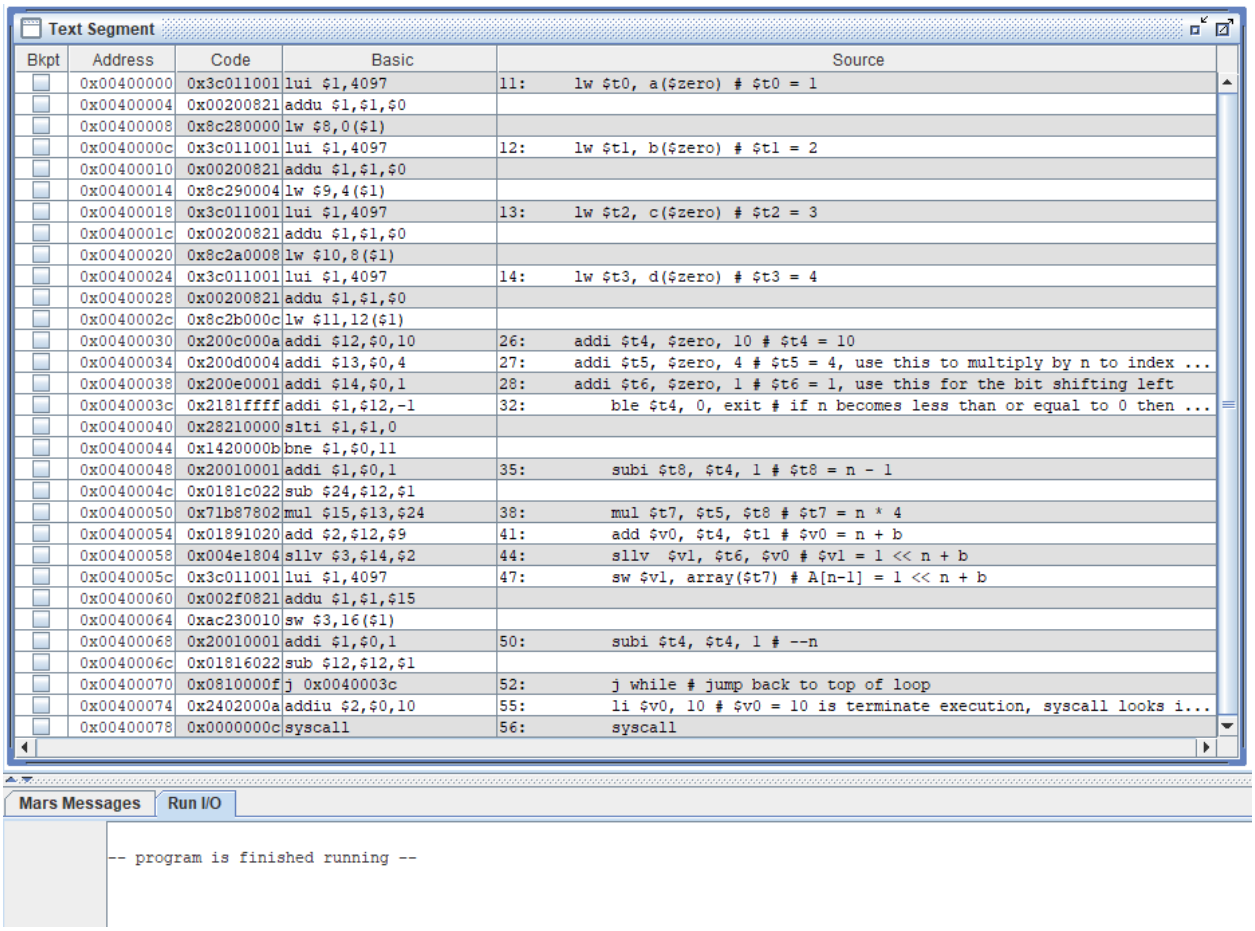
3

MIPS instructions for provided code

```
1  .data
2      a: .word 1
```

```
3       b: .word 2
4       c: .word 3
5       d: .word 4
6       array: .space 40 # Problem 3 specific
7   .text
8       lw $t0, a($zero)
9       lw $t1, b($zero)
10      lw $t2, c($zero)
11      lw $t3, d($zero)
12      # Problem 3 specific:
13      addi $t4, $zero, 10
14      addi $t5, $zero, 4
15      addi $t6, $zero, 1
16      while:
17          ble $t4, 0, exit
18          subi $t8, $t4, 1
19          mul $t7, $t5, $t8
20          add $v0, $t4, $t1
21          sllv  $v1, $t6, $v0
22          sw $v1, array($t7)
23          subi $t4, $t4, 1
24          j while
25      exit:
26          li $v0, 10
27          syscall
```

Successful execution in MARS:



Text Segment

Bkpt	Address	Code	Basic	Source
	0x00400000	0x3c011001	lui $1,4097	11: lw $t0, a($zero) # $t0 = 1
	0x00400004	0x00200821	addu $1,$1,$0	
	0x00400008	0x8c280000	lw $8,0($1)	
	0x0040000c	0x3c011001	lui $1,4097	12: lw $t1, b($zero) # $t1 = 2
	0x00400010	0x00200821	addu $1,$1,$0	
	0x00400014	0x8c290004	lw $9,4($1)	
	0x00400018	0x3c011001	lui $1,4097	13: lw $t2, c($zero) # $t2 = 3
	0x0040001c	0x00200821	addu $1,$1,$0	
	0x00400020	0x8c2a0008	lw $10,8($1)	
	0x00400024	0x3c011001	lui $1,4097	14: lw $t3, d($zero) # $t3 = 4
	0x00400028	0x00200821	addu $1,$1,$0	
	0x0040002c	0x8c2b000c	lw $11,12($1)	
	0x00400030	0x200c000a	addi $12,$0,10	26: addi $t4, $zero, 10 # $t4 = 10
	0x00400034	0x200d0004	addi $13,$0,4	27: addi $t5, $zero, 4 # $t5 = 4, use this to multiply by n to index ...
	0x00400038	0x200e0001	addi $14,$0,1	28: addi $t6, $zero, 1 # $t6 = 1, use this for the bit shifting left
	0x0040003c	0x2181ffff	addi $1,$12,-1	32: ble $t4, 0, exit # if n becomes less than or equal to 0 then ...
	0x00400040	0x28210000	slti $1,$1,0	
	0x00400044	0x1420000b	bne $1,$0,11	
	0x00400048	0x20010001	addi $1,$0,1	35: subi $t8, $t4, 1 # $t8 = n - 1
	0x0040004c	0x0181c022	sub $24,$12,$1	
	0x00400050	0x71b87802	mul $15,$13,$24	38: mul $t7, $t5, $t8 # $t7 = n * 4
	0x00400054	0x01891020	add $2,$12,$9	41: add $v0, $t4, $t1 # $v0 = n + b
	0x00400058	0x004e1804	sllv $3,$14,$2	44: sllv $v1, $t6, $v0 # $v1 = 1 << n + b
	0x0040005c	0x3c011001	lui $1,4097	47: sw $v1, array($t7) # A[n-1] = 1 << n + b
	0x00400060	0x002f0821	addu $1,$1,$15	
	0x00400064	0xac230010	sw $3,16($1)	
	0x00400068	0x20010001	addi $1,$0,1	50: subi $t4, $t4, 1 # --n
	0x0040006c	0x01816022	sub $12,$12,$1	
	0x00400070	0x0810000f	j 0x0040003c	52: j while # jump back to top of loop
	0x00400074	0x2402000a	addiu $2,$0,10	55: li $v0, 10 # $v0 = 10 is terminate execution, syscall looks i...
	0x00400078	0x0000000c	syscall	56: syscall

Mars Messages | Run I/O

-- program is finished running --

4

MIPS instructions for C code from problem 2.27 in textbook

```
1  .data
2      a: .word 3
3      b: .word 4
4      D: .space 80
5  .text
6      lw $s0, a($zero)
7      lw $s1, b($zero)
8      la $s2, D
9
10     addi $t0, $zero, 0
11     outerloop:
12         bge $t0, $s0, exitouter
13
14         addi $t1, $zero, 0
15         innerloop:
16             bge $t1, $s1, exitinner
```

```
17
18              add $v0, $t0, $t1
19              mul $v1, $t1, 4
20              mul $v1, $v1, 4
21              sw $v0, D($v1)
22
23              addi $t1, $t1, 1
24              j innerloop
25
26          exitinner:
27              addi $t0, $t0, 1
28              j outerloop
29
30      exitouter:
31          li $v0, 10
32          syscall
```

Successful execution in MARS: