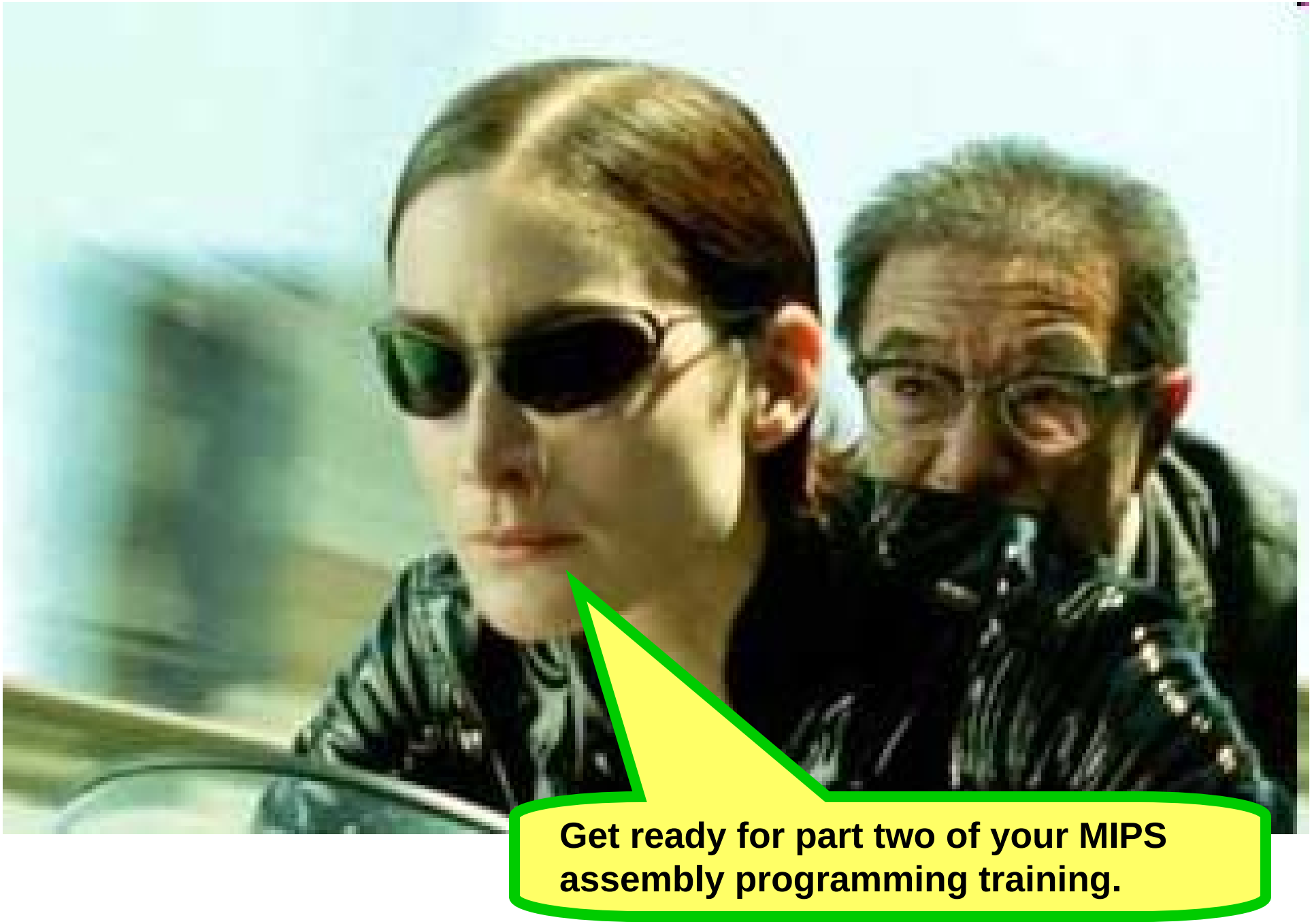
MIPS and SPIM tutorial

Part Two: load and store

November 2008

karl.marklund@it.uu.se


Get ready for part two of your MIPS assembly programming training.

first_try.s

In the first part of the tutorial we examinded the following program in detail.

```
        .text

        .globl main

main:
        addi    $t0, $zero, 3     # a = 3
        addi    $t1, $zero, 2     # b = 2

        add     $t2, $t0, $t1     # c = a + b

        seq     $t3, $t0, $t1     # d = 1 iff a == b else d = 0
        seq     $t4, $t0, $t0     # e = 1 iff a == a (sic)

        jr      $ra               # return to caller
```

The generic format of a MIPS-assembly instruction:

Operation
add, sub, addi, seq.

The first operand register, ***source register*** one.

op rd, rs, rt

Register to hold the result. The ***destination register***.

The second operand source register two.

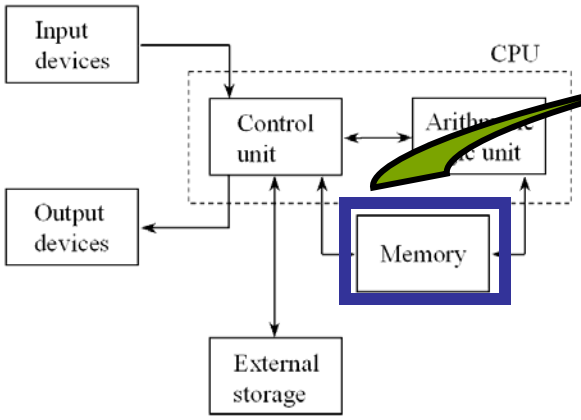
```
add     rd, rs, rt
addi    rd, rs, imm

sub     rd, rs, rt
subi    rd, rs, rt

seq     rd, rs, rt
```

There is no subi instruction... Instead, we can use addi and a negative immediate constant.

Instructions we know so far.



MEMORY	
Address	Content
0xFFFFFFFF	
0xFFFFFFFE	
0xFFFFFFFD	
0xFFFFFFFC	
. . .	
0x00000003	
0x00000002	
0x00000001	
0x00000000	

In MIPS the memory consists of 2^{32} memory cells.

Each cell has a unique *address.*

Four adjacent bytes forms a *word* of 32 bits.

Each cell can store 8 bits – a *byte*.

MIPS is a *Load-Store architecture* which means only load and store instructions are allowed to access memory.

```
lw      rt,  address
sw      rt,  address
```

Address format	Address computation
(register)	Content of register
imm	Immediate
imm(register)	Immediate + content of register
label	Address of label
label +/- imm	Address of label + /- immediate
label +/- imm(register)	Address of label +/- (Immediate + content of register)

to_and_from_memory.s

```
            .data
x:          .word 5
y:          .word 3
z:          .space 4

            .text
            .globl main
main:       la      $t0, x
            lw      $t1, 0($t0)
```

.data

A assembler ***directive*** instructing the assembler to treat what follows as data and place it in the ***data segment***.

We use ***labels*** so we can refer to these locations in memory later.

Each one of these labels denotes a address in the ***data segment***.

Text segment

Here we use the label x

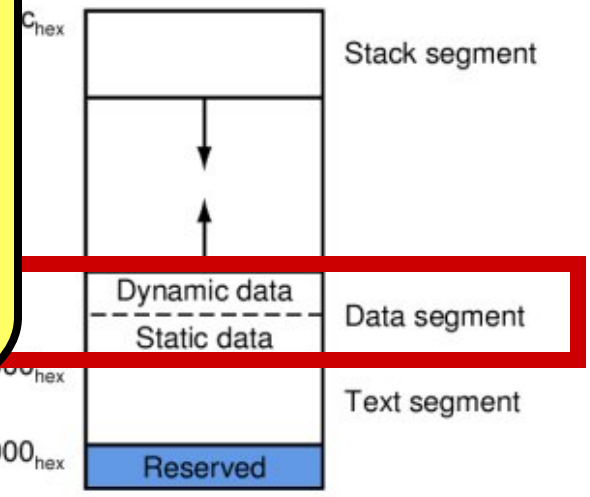
Load Address (la) translates the label x to the address and stores the result in $to.

Load Word (lw)

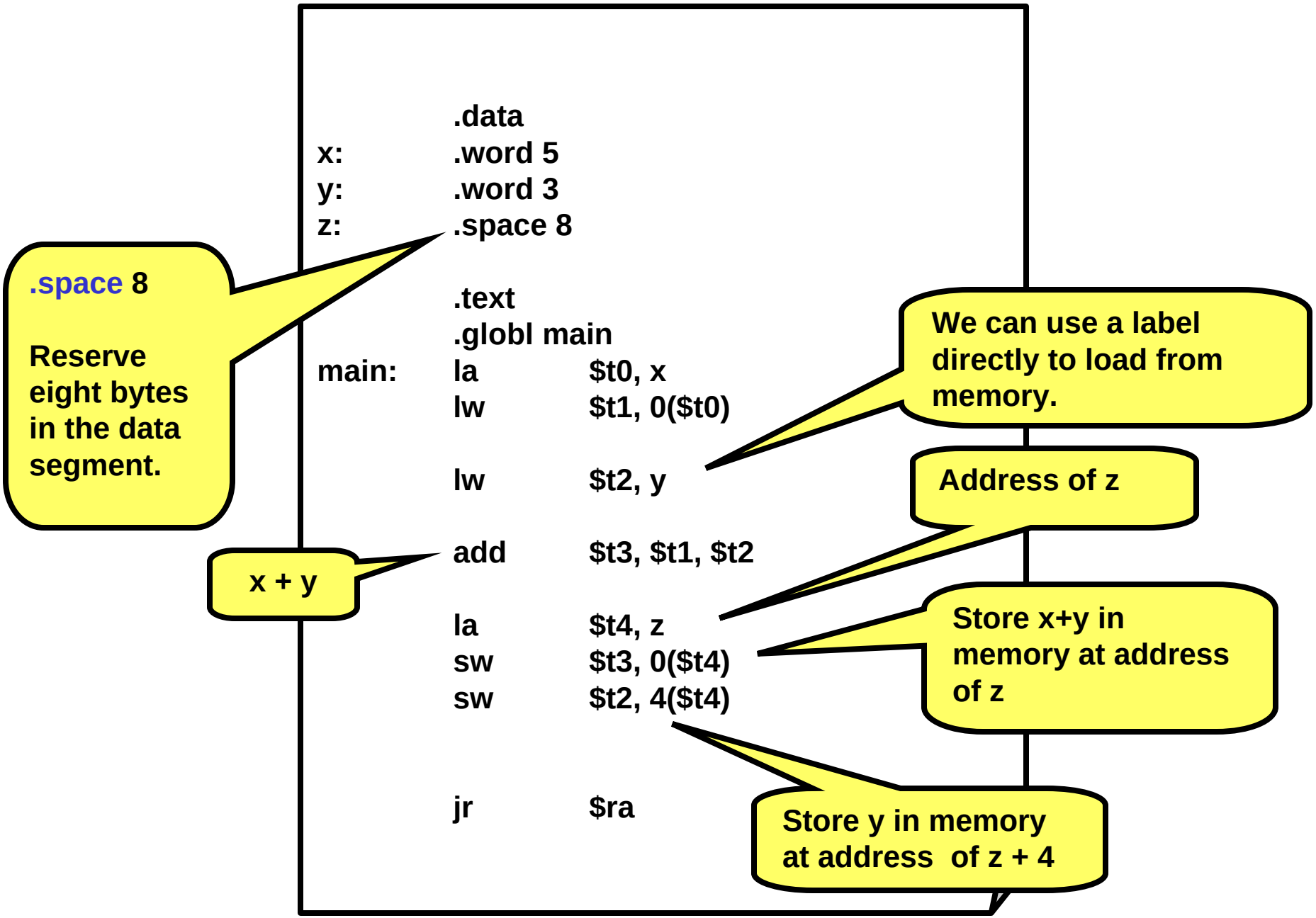
We have a memroy address in $t0.

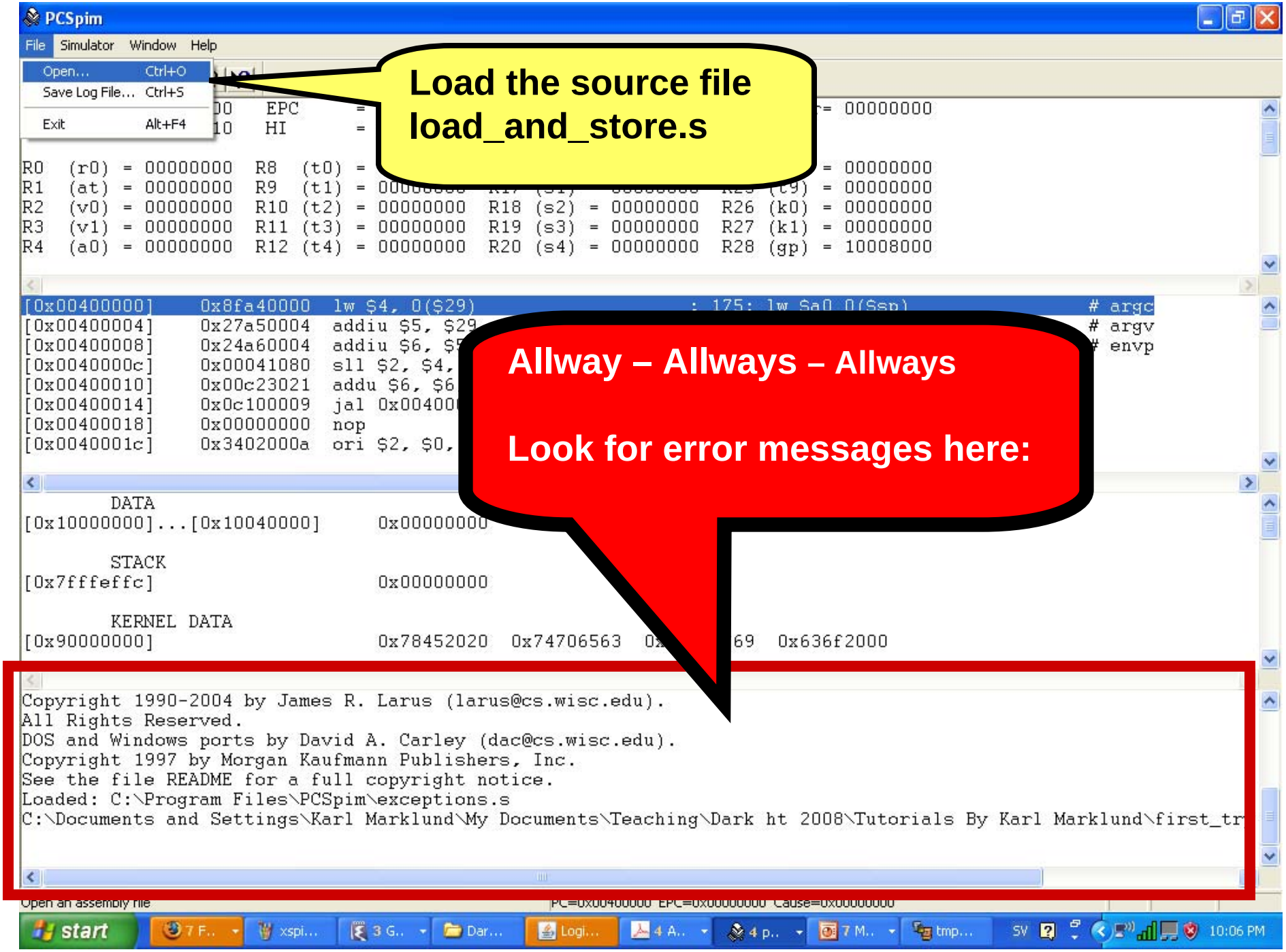
From the address $t0 in memory, load the content at this memory location to register $t1.

$t1 ←5 (x)



to_and_from_memory.s




PCSpim
File Simulator Window Help
Open... Ctrl+O
Save Log File... Ctrl+S
Exit Alt+F4
Load the source file
load_and_store.s
R0 (r0) = 00000000 R8 (t0) =
R1 (at) = 00000000 R9 (t1) =
R2 (v0) = 00000000 R10 (t2) = 00000000 R18 (s2) = 00000000 R26 (k0) = 00000000
R3 (v1) = 00000000 R11 (t3) = 00000000 R19 (s3) = 00000000 R27 (k1) = 00000000
R4 (a0) = 00000000 R12 (t4) = 00000000 R20 (s4) = 00000000 R28 (gp) = 10008000
[0x00400000] 0x8fa40000 lw $4, 0($29)
[0x00400004] 0x27a50004 addiu $5, $29
[0x00400008] 0x24a60004 addiu $6, $5
[0x0040000c] 0x00041080 sll $2, $4,
[0x00400010] 0x00c23021 addu $6, $6
[0x00400014] 0x0c100009 jal 0x00400
[0x00400018] 0x00000000 nop
[0x0040001c] 0x3402000a ori $2, $0,
argc
argv
envp
Allway – Allways – Allways
Look for error messages here:
DATA
[0x10000000]...[0x10040000] 0x00000000
STACK
[0x7fffeffc] 0x00000000
KERNEL DATA
[0x90000000] 0x78452020 0x74706563
Copyright 1990-2004 by James R. Larus (larus@cs.wisc.edu).
All Rights Reserved.
DOS and Windows ports by David A. Carley (dac@cs.wisc.edu).
Copyright 1997 by Morgan Kaufmann Publishers, Inc.
See the file README for a full copyright notice.
Loaded: C:\Program Files\PCSpim\exceptions.s
C:\Documents and Settings\Karl Marklund\My Documents\Teaching\Dark ht 2008\Tutorials By Karl Marklund\first_tr
start
10:06 PM

```
Simulator  Window  Help

       = 00400000   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
atus   = 3000ff10   HI       = 00000000   LO       = 00000000
                              General Registers
 (r0) = 00000000  R8  (t0) = 00000000  R16 (s0) = 00000000  R24 (t8) = 00000000
 (at) = 00000000  R9  (t1) = 00000000  R17 (s1) = 00000000  R25 (t9) = 00000000
 (v0) = 00000000  R10 (t2) = 00000000  R18 (s2) = 00000000  R26 (k0) = 00000000
 (v1) = 00000000  R11 (t3) = 00000000  R19 (s3) = 00000000  R27 (k1) = 00000000
 (a0) = 00000000  R12 (t4) = 00000000  R20 (s4) = 00000000  R28 (gp) = 10008000

00400000]    0x8fa40000  lw $4, 0($29)                ; 175: lw $a0 0($sp)          # argc
00400004]    0x27a50004  addiu $5, $29, 4             ; 176: addiu $a1 $sp 4        # argv
00400008]    0x24a60004  addiu $6, $5, 4              ; 177: addiu $a2 $a1 4        # envp
0040000c]    0x00041080  sll $2, $4, 2                ; 178: sll $v0 $a0 2
00400010]    0x00c23021  addu $6, $6, $2              ; 179: addu $a2 $a2 $v0
00400014]    0x0c100009  jal 0x00400024 [main]        ; 180: jal main
00400018]    0x00000000  nop                          ; 181: nop
0040001c]    0x3402000a  ori $2, $0, 10               ; 183: li $v0 10

     DATA
10000000]...[0x10010000]     0x00000000
10010000]                    0x00000005  0x00000003  0x00000000  0x00000000
10010010]...[0x10040000]     0x00000000

     STACK
7fffeffc]                    0x00000000

yright 1990-2004 by James R. Larus (larus@cs.
 Rights Reserved.
 and Windows ports by David A. Carley (dac@cs
yright 1997 by Morgan Kaufmann Publishers, In
 the file README for a full copyright notice.
ded: C:\Program Files\PCSpim\exceptions.s
Documents and Settings\Karl Marklund\My Docum                    008\Tutorials By Karl Marklund\load_and

lp, press F1                                                  =0x00000000
```

Single step until...

PCSpim

File Simulator Window Help

```
 PC       = 00400024   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000

                                 General Registers
R0   (r0) = 00000000   R8   (t0) = 00000000   R16  (s0) = 00000000   R24  (t8) = 00000000
R1   (at) = 00000000   R9   (t1) = 00000000   R17  (s1) = 00000000   R25  (t9) = 00000000
R2   (v0) = 00000000   R10  (t2) = 00000000   R18  (s2) = 00000000   R26  (k0) = 00000000
R3   (v1) = 00000000   R11  (t3) = 00000000   R19  (s3) = 00000000   R27  (k1) = 00000000
R4   (a0) = 00000000   R12  (t4) = 00000000   R20  (s4) = 00000000   R28  (gp) = 10008000
```

$t0 is 00000000

```
[0x00400008]    0x24a60004  addiu $6, $5, 4                ; 177: addiu $a2 $a1 4          # envp
[0x0040000c]    0x00041080  sll $2, $4, 2                  ; 178: sll $v0 $a0 2
[0x00400010]    0x00c23021  addu $6, $6, $2                ; 179: addu $a2 $a2 $v0
[0x00400014]    0x0c100009  jal 0x00400024 [main]          ; 180: jal main
[0x00400018]    0x00000000  nop                            ; 181: nop
[0x0040001c]    0x3402000a  ori $2, $0, 10                 ; 183: li $v0 10
[0x00400020]    0x0000000c  syscall                        ; 184: syscall                  # syscall
[0x00400024]    0x3c081001  lui $8, 4097 [x]               ; 8: la     $t0, x
```

... you reach the la $t0, x instruction.

```
        DATA
[0x10000000]..
[0x10010000]                                    0x00000003  0x00000000  0x00000000
[0x10010010]..

        STACK
[0x7fffeffc]                    0x00000000
```

Step again!

```
C:\Documents and Settings\Karl Marklund\My                  ht                  Karl Marklund\loa
[0x00400000]    0x8fa40000  lw $4, 0($29)                   lw                  0($sp)          # argc
[0x00400004]    0x27a50004  addiu $5, $29,                  iu $a1 $sp 4        # argv
[0x00400008]    0x24a60004  addiu $6, $5,                   addiu $a2 $a1 4     # envp
[0x0040000c]    0x00041080  sll $2, $4, 2                   sll $v0 $a0 2
[0x00400010]    0x00c23021  addu $6, $6, $2                 addu $a2 $a2 $v0
[0x00400014]    0x0c100009  jal 0x00400024                  jal main
```

PCSpim

File Simulator Window Help

```
 PC       = 00400028   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000

                                Registers
R0   (r0) = 00000000   R8   (t0) = 10010000   R16  (s0) = 00000000   R24  (t8) = 00000000
R1   (at) = 10010000   R9   (t1) = 00000000   R17  (s1) = 00000000   R25  (t9) = 00000000
R2   (v0) = 00000000   R10  (t2) = 00000000   R18  (s2) = 00000000   R26  (k0) = 00000000
R3   (v1) = 00000000   R11  (t3) = 00000000   R19  (s3) = 00000000   R27  (k1) = 00000000
R4   (a0) = 00000000   R12  (t4) = 00000000   R20  (s4) = 00000000   R28  (gp) = 10008000

[0x0040000c]    0x00041080  sll $2, $4, 2                    ; 178: sll $v0 $a0 2
[0x00400010]    0x00c23021  addu $6, $6, $2                  ; 179: addu $a2 $a2 $v0
[0x00400014]    0x0c100009  jal 0x00400024 [main]            ; 180: jal main
[0x00400018]    0x00000000  nop                              ; 181: nop
[0x0040001c]    0x3402000a  ori $2, $0, 10                   ; 183: li $v0 10
[0x00400020]    0x0000000c  syscall                          ; 184: syscall                  # sysc
[0x00400024]    0x3c081001  lui $8, 4097 [x]                 ; 8: la     $t0, x
[0x00400028]    0x8d090000  lw $9, 0($8)                     ; 9: lw     $t1, 0($t0)

        DATA
[0x10000000]..[0x10010000]
[0x10010000]                0x00000005
[0x10040000]..[0x10040000]

        STACK
[0x7fffeffc]                0x00000000

[0x00400000
[0x00400000
[0x00400000
[0x00400000
[0x00400010
[0x00400014
[0x00400020
```

$t0 is now 10010000

The content at address 0x1001000 is 0x00000005

The label x is refering to the address 0x1001000 in the data segment of the memory.

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 0040002c   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000

                              General Registers
R0   (r0) = 00000000                                  R16 (s0) = 00000000
R1   (at) = 00000000   R9   (t1) = 00000005           R24 (t8) = 00000000   R25 (t9) = 00000000
R2   (v0) = 00000000                                  R18 (s2) = 00000000   R26 (k0) = 00000000
R3   (v1) = 00000000   R11  (t3) = 00000000   R19 (s3) = 00000000   R27 (k1) = 00000000
R4   (a0) = 00000000   R12  (t4) = 00000000   R20 (s4) = 00000000   R28 (gp) = 10008000

[0x00400010]    0x00c23021  addu $6, $6, $2                ; 179: addu $a2 $a2 $v0
[0x00400014]    0x0c100009  jal 0x00400024 [main]          ; 180: jal main
[0x00400018]    0x00000000  nop                            ; 181: nop
[0x0040001c]    0x3402000a  ori $2, $0, 10                 ; 183: li $v0 10
[0x00400020]    0x0000000c  syscall                        ; 184: syscall                    # syscall
[0x00400024]    0x3c081001  lui $8, 4097 [x]               ; 8: la     $t0, x
[0x00400028]    0x8d090000  lw $9, 0($8)                   ; 9: lw     $t1, 0($t0)
[0x0040002c]    0x3c011001  lui $1, 4097                   ; 11: lw    $t2, y

        DATA
[0x10010000]    [0x10010000]    0x00000005
                [0x10040000]

        STACK
[0x7fffeffc]    0x00000000

[0x00400004
[0x00400008
[0x0040000c
[0x00400010
[0x00400014
[0x00400024
[0x00400028
```

$t1 is now 00000005

The content at address 0x1001000 i 0x00000005

The label x is refering to the address 0x1001000 in the data segment of the memory.

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 00400030   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000

                                  General Registers
R0  (r0) =                                (s0) = 00000000  R24 (t8) = 00000000
R1  (at) =                                (s1) = 00000000  R25 (t9) = 00000000
R2  (v0) =                                (s2) = 00000000  R26 (k0) = 00000000
R3  (v1) =                                (s3) = 00000000  R27 (k1) = 00000000
R4  (a0) =                                (s4) = 00000000  R28 (gp) = 10008000

[0x00400014]    0x0c100009  jal 0x004000    in]          ; 180: jal main
[0x00400018]    0x00000000  nop                          ; 181: nop
[0x0040001c]    0x3402000a  ori $2, $0, 10               ; 183: li $v0 10
[0x00400020]    0x0000000c  syscall                      ; 184: syscall                 # sysca
[0x00400024]    0x3c081001  lui $8, 4097 [x]             ; 8: la     $t0, x
[0x00400028]    0x8d090000  lw $9, 0($8)                 ; 9: lw     $t1, 0($t0)
[0x0040002c]    0x3c011001  lui $1, 4097                 ; 11: lw    $t2, y
[0x00400030]    0x8c2a0004  lw $10, 4($1)

[0x1000
[0x100]                                  0x00000003  0x00000000  0x00000000
[0x100]

[0x7fff

[0x0040                                  4                                   envp
[0x0040000c]                             2
[0x00400010]    0x00c23021  addu $6, $6, $2
[0x00400014]    0x0c100009  jal 0x00400024 [main]
[0x00400024]    0x3c081001  lui $8, 4097 [x]                                 0, x
[0x00400028]    0x8d090000  lw $9, 0($8)                                     1, 0($t0)
[0x0040002c]    0x3c011001  lui $1, 4097                                     2, y
```

Loading using a label directly

This cannot be translated directly to one machine instruction – its a pseduo instruction.

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 00400034   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000
                                 General Registers
R0   (r0) = 00000000  R8   (t0) = 10010000  R16  (s0) = 00000000
R1   (at) = 10010000  R9   (t1) = 00000005  R17  (s1) = 00000000  R25  (t9) = 00000000
R2   (v0) = 00000000  R10  (t2) = 00000003  R18  (s2) = 00000000  R26  (k0) = 00000000
R3   (v1) = 00000000  R11  (t3) = 00000000  R19  (s3) = 00000000  R27  (k1) = 00000000
R4   (a0) = 00000000  R12  (t4) = 00000000  R20  (s4) = 00000000  R28  (gp) = 10008000
```

$t0 is now 00000003

```
[0x00400018]    0x00000000  nop                                  ; 181: nop
[0x                          $0, 10                              ; 183: li $v0 10
[                                                                ; 184: syscall                  # syscall
[                            097 [x]                             ; 8: la     $t0, x
[                            $8)                                 ; 9: lw     $t1, 0($t0)
[                            097                                 ; 11: lw    $t2, y
[                            ($1)
[                            $9, $10                             ; 13: add   $t3, $t1, $t2
```

The label y is refering to the address 0x1001004 in the data segment of the memory.

```
[0x                          000000    0x00000003  0x00000000  0x00000000
[0x10010010]...[0x10040000]     0x0000000

        STACK
[0x7fffeffc]                    000
```

The content at address 0x1001004 is 0x00000003

```
[0x00400                                                        $a0
[0x00400                                                          $a
[0x00400                                  main]
[0x00400                                                        , x
[0x00400028]                                                    , 0($t0)
[0x0040002c]    0x3c011001  lui $1, 4097                        , y
[0x00400030]    0x8c2a0004  lw $10, 4($1)
```

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 00400038   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000
                                 General Registers
R0   (r0) = 00000000  R8   (t0) = 10010000  R16 (s0) = 00000000
R1   (at) = 10010000  R9   (t1) = 00000005  R17 (s1) = 00000000
R2   (v0) = 00000000  R10  (t2) = 00000000  R18 (s2) = 00000000  R26 (k0) = 00000000
R3   (v1) = 00000000  R11  (t3) = 00000008  R19 (s3) = 00000000  R27 (k1) = 00000000
R4   (a0) = 00000000  R12  (t4) = 00000000  R20 (s4) = 00000000  R28 (gp) = 10008000
```

$t0 is now 00000008

```
[0x0040001c]    0x3402000a  ori $2, $0, 10                ; 183: li $v0 10
[0x00400020]    0x0000000c  syscall                       ; 184: syscall                    # syscall
[0x00400000]    0x3c011001  lui $1, 4097 [x]              ; 8: la     $t0, x
[0x00400004]    0x8c280000  lw $8, 0($8)                  ; 9: lw     $t1, 0($t0)
[0x00400008]    0x3c011001  lui $1, 4097                  ; 11: lw    $t2, y
[0x0040000c]    0x8c2a0004  lw $10, 4($1)
[0x00400010]    0x00c23021  addu $6, $6, $2               ; 13: add   $t3, $t1, $t2
[0x00400014]    0x0c100009  jal 0x00400024 [main]         ; 15: la    $t4, z
```

The sum x + y (loaded from memory) is stored in $t3.

```
[0x10000000]...[0x10010000]     0x00000000
[0x10010000]                    0x00000005  0x00000003  0x00000000  0x00000000
[0x10010010]...[0x10040000]     0x00000000

        STACK
[0x7fffeffc]                    0x00000000
```

```
[0x00400010]    0x00c23021  addu $6, $6, $2
[0x00400014]    0x0c100009  jal 0x00400024 [main]
[0x00400024]    0x3c081001  lui $8, 4097 [x]
[0x00400028]    0x8d090000  lw $9, 0($8)
[0x0040002c]    0x3c011001  lui $1, 4097
[0x00400030]    0x8c2a0004  lw $10, 4($1)
[0x00400034]    0x012a5820  add $11, $9, $10
```

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 0040003c   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000
                                 General Registers
R0  (r0) = 00000000  R8  (t0) = 10010000  R16 (s0) = 00000000  R24 (t8) = 00000000
R1  (at) = 10010000  R9  (t1) = 00000005  R17 (s1) = 00000000  R25 (t9) = 00000000
R2  (v0) = 00                             (s2) = 00000000  R26 (k0) = 00000000
R3  (v1) = 0                              s3) = 00000000  R27 (k1) = 00000000
R4  (a0) = 0                              s4) = 00000000  R28 (gp) = 10008000
```

Load Address is also a pseudo instruction

```
[0x00400020]
[0x00400024]
[0x00400028]
[0x0040002c]    0x3c011001  lui $1,
[0x00400030]    0x8c2a0004  lw $10, 4($1)
[0x00400034]    0x012a5820  add $11, $9, $10
[0x00400038]    0x3c011001  lui $1, 4097 [z]
[0x0040003c]    0x342c0008  ori $12, $1, 8 [z]
```

```
    ; 184: syscall                                  # sysc
    ; 8: la     $t0, x
    ; 9: lw     $t1, 0($t0)
    ; 11: lw    $t2, y

    ; 13: add    $t3, $t1, $t2
    ; 15: la     $t4, z
```

```
        DATA
[0x10000000]...[0x10010000]     0x00000000
[0x10010000]                    0x00000005  0x00000003  0x00000000  0x00000000
[0x10010010]...[0x10040000]     0x00000000

        STACK
[0x7fffeffc]                    0x00000000
```

Step again!

```
[0x00400014]    0x0c100009  jal 0x00400024 [main]
[0x00400024]    0x3c081001  lui $8, 4097 [x]
[0x00400028]    0x8d090000  lw $9, 0($8)
[0x0040002c]    0x3c011001  lui $1, 4097
[0x00400030]    0x8c2a0004  lw $10, 4($1)
[0x00400034]    0x012a5820  add $11, $9, $10
[0x00400038]    0x3c011001  lui $1, 4097 [z]
```

PCSpim

File Simulator Window Help

```
 PC       = 00400040   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000

                              General Registers
R0   (r0) = 00000000  R8   (t0) = 10010000  R16  (s0) = 00000000
R1   (at) = 10010000  R9   (t1) = 00000005  R17  (s1) = 00000000
R2   (v0) = 00000000  R10  (t2) = 00000003  R18  (s2) = 00000000
R3   (v1) = 00000000  R11  (t3) = 00000000  R19  (s3) = 00000000  R27  (k1) = 00000000
R4   (a0) = 00000000  R12  (t4) = 10010008  R20  (s4) = 00000000  R28  (gp) = 10008000
```

$t4 is now 10010008

```
[0x00400024]
[0x00400028]
[0x0040002c]
[0x00400030]
[0x00400034]
[0x00400038]
[0x0040003c]
[0x00400040]
```

```
; 8: la     $t0, x
; 9: lw     $t1, 0($t0)
; 11: lw    $t2, y

; 13: add   $t3, $t1, $t2
; 15: la    $t4, z

; 16: sw    $t3, 0($t4)
```

The label z is refering to the address 0x1001008 in the data segment of the memory.

```
        DATA
[0x10000000]...[0x10010000]     0x00000000
[0x10010000]                    0x00000005  0x00000003  0x00000000  0x00000000
[0x10010010]...[0x10040000]     0x00000000

        STACK
[0x7fffeffc]                    0x00000000
```

Address 0x1001000

Address 0x1001004

```
[0x00400024]    0x3c081001  lui $8, 4097 [x]
[0x00400028]    0x8d090000  lw $9, 0($8)
[0x0040002c]    0x3c011001  lui $1, 4097
[0x00400030]    0x8c2a0004  lw $10, 4($1)
[0x00400034]    0x012a5820  add $11, $9, $10
[0x00400038]    0x3c011001  lui $1, 4097 [z]
[0x0040003c]    0x342c0008  ori $12, $1, 8 [z]
```

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 00400044   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000
                                General Registers
R0   (r0) = 00000000  R8   (t0) = 10010000  R16  (s0) = 00000000  R24  (t8) = 00000000
R1   (at) = 10010000  R9   (t1) = 00000005  R17  (s1) = 00000000  R25  (t9) = 00000000
R2   (v0) = 00000000  R10  (t2) = 00000003  R18  (s2) = 00000000  R26  (k0) = 00000000
R3   (v1) = 00000000  R11  (t3) = 00000008  R19  (s3) = 00000000  R27  (k1) = 00000000
R4   (a0) = 00000000  R12  (t4) = 10010008  R20  (s4) = 00000000  R28  (gp) = 10008000
```

```
[0x00400028]    0x8d090000  lw $9, 0($8)
[0x0040002c]    0x3c011001  lui $1, 4097
[0x00400030]    0x8c2a0004  lw $10, 4($1)
[0x00400034]    0x012a5820  add $11, $9, $10
[0x00400038]    0x3c011001  lui $1, 4097 [z]
[0x0040003c]    0x342c0008  ori $12, $1, 8 [z]
[0x00400040]    0xad8b0000  sw $11, 0($12)
```

```
; 9: lw      $t1, 0($t0)
; 11: lw     $t2, y

; 13: add    $t3, $t1, $t2
; 15: la     $t4, z

; 16: sw     $t3, 0($t4)
; 17: sw     $t2, 4($t3)
```

The value in register $t3 (x+z) is stored in the memory at location given by $t4 (z).

```
        DATA
[0x10000000]...[0x10010000]     0x00000000
[0x10010000]                    0x00000005  0x00000003  0x00000008  0x00000000
[0x10010010]...[0x10040000]     0x00000000

        STACK
[0x7fffeffc]                    0x00000000
```

Step again!

PCSpim

File Simulator Window Help

```
 PC       = 00400048   EPC      = 00000000   Cause    = 00000000   BadVAddr= 00000000
 Status   = 3000ff10   HI       = 00000000   LO       = 00000000
                                 General Registers
R0   (r0) = 00000000  R8   (t0) = 10010000  R16  (s0) = 00000000  R24  (t8) = 00000000
R1   (at) = 10010000  R9   (t1) = 80000005  R17  (s1) = 00000000  R25  (t9) = 00000000
R2   (v0) = 00000000  R10  (t2) = 00000003  R18  (s2) = 00000000  R26  (k0) = 00000000
R3   (v1) = 00000000  R11  (t3) = 00000000  R19  (s3) = 00000000  R27  (k1) = 00000000
R4   (a0) = 00000000  R12  (t4) = 10010008  R20  (s4) = 00000000  R28  (gp) = 10008000
```

```
[0x0040002c]  0x3c011001  lui $1, 4097
[0x00400030]  0x8c2a0004  lw $10, 4($1)
[0x00400034]  0x012a5820  add $11, $9, $10
[0x00400038]  0x3c011001  lui $1, 4097 [z]
[0x0040003c]  0x342c0008  ori $12, $1, 8 [z]
[0x00400040]  0xad8b0000  sw $11, 0($12)
[0x00400044]  0xad8a0004  sw $10, 4($12)
```

```
                ; 11: lw    $t2, y

                ; 13: add   $t3, $t1, $t2
                ; 15: la    $t4, z

                ; 16: sw    $t3, 0($t4)
                ; 17: sw    $t2, 4($t4)
                ; 20: jr    $ra
```

The value in register $t2 (x) is stored in the memory at location given by $t4 (z) + 4.

```
        DATA
[0x10000000]...[0x10010000]     0x00000000
[0x10010000]                    0x00000005  0x00000003  0x00000008  0x00000003
[0x10010010]...[0x10040000]     0x00000000

        STACK
[0x7fffeffc]                    0x00000000
```

Step again!


You are now done with part two of your MIPS assembly programming training.