Zack Fravel

010646947

Lab 1: M 4:10 - 5:55 PM

Lab 5 - SPIM Simulator

3/28/16

The main purpose of this lab was our introduction to writing assembly code from given C++ examples. We were given a pre lab example that was a simple function call with two variables that called upon another function. The code for the pre lab is included at the end of the report. Once we got the pre lab working, we were to change the main program to accommodate for arrays and use the same functions to perform the same operation multiple times using a for loop.

## Approach

The following is the program we wish to implement in MIPS using the standard 32 register architecture.

```
void swap (int a, int b)
   int temp=a;
   a=b;
  b=temp;
int distance (int a, int b)
   if (b > a)
     swap (a,b);
 return (a-b)
void main(void)
 int var1 =50;
 int var2= 200;
 int result=0;
 int sum =0;
 result = distance (var1, var2);
  return;
}
```

```
int main(void)
{
  int var1[4] = {4, 7, 12, 5};
  int var2[4] = {15, 3, 6, 14};
  int result[4] = {0};

for (int i=0; i < 4; i++)
  result[i] = distance (var1[i], var2[i]);
}</pre>
```

The code on the left is the pre lab code we were assigned, and the segment on the right is the new main program we were asked to implement in the lab.

## **Experimentation**

In my pre lab example, I have around 32 instructions that MIPS reads to complete the entire program. Of these instructions, about 31% are lw and sw functions, 32% arithmetic, and the rest being control (branch) instructions. One of the main things I had to work on was the use of the \$sp when calling different functions. We push the \$sp by 12 to account for 3 different variables when calling the distance function, then store the 3 addresses we had in the registers to memory so we can call upon them after we call the distance function.

```
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
             0000000050 0000000200
                                    000000000 000000000
[10010000]
[10010010]..[1003fffff] 00000000
User Stack [7ffffe20]..[80000000]
[7ffffe20]
             0000000001 2147483229
                                    0000000000 2147483591
                                                             . . . . ] . . . . . . .
                                                             7ffffe301
             2147483553
                         2147483537
                                    2147483519
                                                2147483443
             2147483377 2147483342 2147483328 2147483287
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 0000000
[10010000]
            000000050 000000200 000000000 000000000
[10010010]..[1003fffff] 00000000
User Stack [7ffffe14]..[80000000]
[7ffffe14]
             000000000
                         000000000
                                    0268500992
7ffffe201
             0000000001
                         2147483229
                                    0000000000
                                                2147483591
                                                             . . . . ] . . . . . . . .
[7ffffe30]
             2147483553
                         2147483537
                                    2147483519
                                                2147483443
[7ffffe40]
             2147483377
                         2147483342
                                    2147483328
                                                2147483287
User data segment [10000000]..[10040000]
[10000000]..[1000ffff]
                       00000000
             000000050 0000000200
[10010000]
                                     000000000 000000000
                                                              2 . . . . . . . . . . . . . . .
[10010010]..[1003fffff] 00000000
User Stack [7ffffe14]..[80000000]
[7ffffe14] 0268501000 0268500
                         0268500996
                                     0268500992
[7ffffe14]
                                                             2147483591
7ffffe201
             0000000001
                         2147483229
                                     0000000000
[7ffffe30]
             2147483553
                         2147483537
                                     2147483519
                                                2147483443
             2147483377
                         2147483342
                                     2147483328
                                                2147483287
7ffffe401
[7ffffe50]
                                     0000000000 1934962483
             2147483276 2147483262
```

When we do this, the stack increases and allows us to store 3 more 32 bit addresses on the top of the stack.

When we use the jump-and-link (jal) and jump-register (jr), this effects the program counter (PC) as well as our \$ra. Whenever you call a function, the \$ra is saved so you can call upon it at the end of your function to return to wherever you were in the main program, therefore whenever you use a (jal) or (jr), the \$ra becomes a new address. This is why it is necessary to

save the \$ra when calling another function. The PC changes as well when you use these

instruction types, because a jump indicates jumping to another instruction.

```
PC = 4194384
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
                                                                                                                                                                                                                                                                                                                                                                                                                                                                   00400000]..[00440000]
                                                                                                                                                                                                                                                                                                                                                                                      er Text Segment [00400000]..[00440
183: lw $a0 0($sp) # argc
184: addiu $a1 $sp 4 # argv
185: addiu $a2 $a1 4 # envp
186: sl1 $v0 $a0 2
187: addu $a2 $a2 $v0
188: ja1 main
189: nop
191: li $v0 10
192: syscall # syscall 10 (exit)
9: la $t0, var1
10: la $t1, var2
                                                                                                                                                               [00400000]
[00400004]
[00400008]
                                                                                                                                                                                                               8fa40000
27a50004
24a60004
                                                                                                                                                                                                                                                             lw $4, 0($29)
addiu $5, $29, 4
addiu $6, $5, 4
sll $2, $4, 2
addu $6, $6, $2
                                                                                                                                                                 0040000c
                                                                                                                                                                                                                00041080
                                                                                                                                                                                                                                                              addu $6, $6, $2
jal 0x00400024 [main]
                                                                                                                                                                 004000101
                                                                                                                                                                                                                00c23021
                                                                                                                                                               [00400010]
[00400014]
[00400018]
[0040001c]
[00400020]
                                                                                                                                                                                                               0c100009
00000000
3402000a
0000000c
                                                                                                                                                                                                                                                           jal 0x00400024 [main]
nop
ori $2, $0, 10
syscall
lui $8, 4097 [var]
lui $1, 4097 [var2]
ori $9, $1, 4 [var2]
lui $1, 4097 [result]
ori $10, $1, 8 [result]
lw $4, 0($8)
lw $5, 0($9)
addi $29, $29, -12
sw $8. 8($29)
 R0 [r0] = 0

R1 [at] = 268500992

R2 [v0] = 4

R3 [v1] = 0

R4 [a0] = 50

R5 [a1] = 200

R6 [a2] = 2147483180

R7 [a3] = 0

R8 [t0] = 268500992

R9 [t1] = 268500996

R10 [t2] = 268501000

R11 [t3] = 0
                                                                                                                                                               [00400024]
[00400028]
                                                                                                                                                                                                                3c081001
3c011001
                                                                                                                                                              [00400028]
[0040002c]
[00400030]
[00400034]
[00400038]
                                                                                                                                                                                                                34290004
                                                                                                                                                                                                               3c011001
342a0008
8d040000
8d250000
                                                                                                                                                                                                                                                                                                                                                                                 ; 11: la $t2, result
                                                                                                                                                                                                                                                            [00400040]
[00400044]
                                                                                                                                                                                                               23bdfff4
                                                                                                                                                                                                                                                                                                                                                                                                                                                                             -12 #Pushing $sp into stack
                                                                                                                                                                                                               afa80008
                                                                                                                                                             [00400044] afa90004
[0040004c] afaa0000
[00400050] 0c100020
                                                                                                                                                                                                                                                               Use:

lw $4, 0($29)
addiu $5, $29, 4
;
addiu $5, $29, 4
;
ddiu $6, $5, 4
;
s1l $2, $4, 2
;
addu $6, $6, $2
;
jal 0x00400024 [main]
;
nop
ori $2, $0, 10
;
syscall
lui $8, 4097 [var1]
jui $1, 4097 [var2]
ori $9, $1, 4 [var2]
jui $1, 4097 [result]
;
ori $10, $1, 8 [result]
lui $5, 0($9)
addi $29, $29, -12
;
sw $8, 8($29)
;
sw $9, 4($29)
;
sw $9, 4($29)
jal 0x00400080 [distance];
lw $10, 0($29)
jal 0x0040080
lw $10, 0($29)
jal 0x044
s29, $29, 12
;
sw $2, 0($10)
ori $2, $0, 10
;
syscall
add $8, $4, $0
;
add $5, $8, $0
;
sal add $8, $4, $0
;
add $8, $4, $0
                                                                                                                                                                                                                                                                                                                                                                                       User Text Segment [00400000]..[00440000]; 183: lw $a0 0($sp) # argc; 184: addiu $a1 $sp 4 # argv; 185: addiu $a2 $a1 4 # envp; 186: s11 $v0 $a0 2; 187: addu $a2 $a2 $v0; 188: jal main; 189: nop; 191: li $v0 10; 192: syscall # syscall 10 (exit); 9: la $t0, var1; 10: la $t1, var2
PC
EPC
                                                                                                                                                                   [00400000] 8fa40000
[00400004] 27a50004
                                                   0
 Cause
 BadVAddr = 0
Status = 805371664
                                                                                                                                                                    [00400004]
[00400008]
[0040000c]
[00400014]
                                                                                                                                                                                                                     24a60004
00041080
                                                                                                                                                                                                                      00c23021
0c100009
                                                                                                                                                                      00400018
                                                                                                                                                                                                                      00000000
                                                                                                                                                                    [00400018]
[0040001c]
[00400020]
[00400024]
                                                                                                                                                                                                                     3402000a
0000000c
                                                                                                                                                                                                                      3c081001
                                                   0
268500992
                    [at]
[v0]
[v1]
[a0]
[a1]
[a2]
                                                                                                                                                                      00400028
                                                                                                                                                                                                                      3c011001
                                                                                                                                                                    [00400028]
[0040002c]
[00400030]
[00400034]
                                                                                                                                                                                                                     34290004
3c011001
                                        = 0
= 50
= 200
= 2147483180
                                                                                                                                                                                                                                                                                                                                                                                                ; 11: la $t2, result
                                                                                                                                                                                                                                                                                                                                                                                             ; 11: 1a $t2, result
; 14: lw $a0, 0($t0)
; 15: lw $a1, 0($t1)
; 19: add $sp, $sp, -12
; 20: sw $t0, 8($sp)
; 21: sw $t1, 4($sp)
; 22: sw $t2, 0($sp)
]; 23: jal distance
; 24: lw $t2, 0($sp)
; 25: lw $t1, 4($sp)
; 25: lw $t1, 4($sp)
; 27: add $sp, $sp, 12
; 30: sw $v0, 0($t2)
; 31: li $v0, 10
; 32: syscall
; 36: add $t0, $a0, $0
; 37: add $a1, $t0, $0
; 37: add $a1, $t0, $0
; 37: add $a1, $t0, $0
; 38: add $a1, $t0, $0
; 39: jr $ra
                                                                                                                                                                                                                     342a0008
                                                                                                                                                                   [00400034]
[00400038]
[0040003c]
[00400044]
                                                                                                                                                                                                                       8d040000
                                                                                                                                                                                                                     8d250000
23bdfff4
                 [a2] = 214748318(

[a3] = 0

[t0] = 268500992

[t1] = 268500996

[t2] = 268501000

[t4] = 0

[t5] = 0

[t6] = 0

[t7] = 0

[s0] = 0

[s1] = 0

[s2] = 0

[s3] = 0
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  -12 #Pushing $sp into stack
                                                                                                                                                                                                                      afa80008
                                                                                                                                                                                                                     afa90004
afaa0000
0c100020
                                                                                                                                                                      00400048
                                                                                                                                                                    [00400048]
[0040004c]
[00400050]
[00400054]
 R10
                                                                                                                                                                                                                     8faa0000
                                                                                                                                                                      00400058
                                                                                                                                                                                                                      8fa90004
                                                                                                                                                                                                                    8fa90004
8fa80008
23bd000c
ad420000
3402000a
0000000c
00804020
00a02020
                                                                                                                                                                    [00400056]
[0040005c]
[00400060]
 R16
R17
R18
                                                                                                                                                                     [00400064]
[00400068]
[0040006c]
[00400074]
                                                                                                                                                                       00400078
                                                                                                                                                                                                                     01002820
                                                                                                                                                                   [00400076] 03e00008
[00400080] 00804020
```

The PC changes by whatever amount is indicated in the offset in the instruction, making the next instruction to be executed whatever was specified.

There are 3 segments in the data section, our data is stored per byte of information.

## **Results and Conclusion**

I was not able to get the fully functioning main program from the lab working, however the pre lab works as is. Both the lab code and the pre lab code are included on the back.

```
.data
  var1: .word 50
   var2: .word 200
   result: .word 0
.text
main:
  la $t0, var1
la $t1, var2
                                                       .data
  la $t2, result
                                                          var1: .word 4, 7, 12, 5
                                                          var2: .word 15, 3, 6, 14
                                                          result: .space 4
  lw $a0, 0($t0)
lw $a1, 0($t1)
                                                       .text
 add $sp, $sp, -12 #Pushing $sp into stack sw $t0, 8($sp) sw $t1, 4($sp) sw $t2, 0($sp)
                                                       main:
                                                         la $t0, var1
la $t1, var2
                                                         la $t2, result
or $t3, $0, $0
  jal distance
                                                                                     # initialize i
  lw $t2, 0($sp)
 lw $t1, 4($sp)
lw $t0, 8($sp)
                                                        for_loop:
 add $sp, $sp, 12
                                                         slti $s0, $s3, 4
                                                         beq $s0, $0, for_done
 sw $v0, 0($t2)
  li $v0,10
                                                         sll $t4, $t3, 2
                                                                                     # t2 goes from i to i*4
 syscall
                                                         add $s0, $t4, $t0
                                                                                     # s0 is var1[i] address
                                                         add $s1, $t4, $t1
                                                                                     # s1 is var2[i] address
                                                         add $s2, $t4, $t2
                                                                                     # s2 is result[i] address
swap:
 add
        $t0, $a0, $0
        $a0, $a1, $0
 add
                                                         lw $a0, 0($s0)
 add
       $a1, $t0, $0
                                                         lw $a1, 0($s1)
 jr
        $ra
                                                         lw $a2, 0($s2)
distance:
 add
       $t0, $a0, $0
                                                         jal distance
 add
       $t1, $a1, $0
                                                         addi $s3, 1
        $t2, $t0, $t1
       $t2, $0, if_done
 beq
                                                         j for_loop
                                                        for done:
        $sp, $sp, -4 #Pushing $sp into stack
 add
                                                         li $v0,10
        $ra, 0($sp)
  SW
                                                         syscall
  add
        $a0, $t0, $0 #call the swap function
 add
        $a1, $t1, $0
```

jal

swap