

## 数字系统II实验

MIPS汇编程序设计



## 实验目的

- 1. 掌握QTSPIM的调试技术
- 2. 了解MIPS汇编语言与机器语言之间的 对应关系
- 3. 掌握MIPS汇编程序设计
- 4. 了解C语言语句与汇编指令之间的关系
- 5. 熟悉常见的MIPS汇编指令

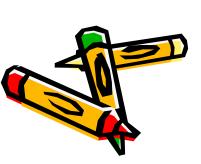
掌握程序的内存映像

## 实验环境简介——QTSPIM

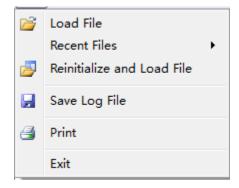
#### · QTSPIM:

> File Simulator Registers Text Segment Data Segment Window Help n = ∃ 0 Int Regs [16] Int Regs [16] User Text Segment [00400000]..[00440000] EPC ; 183: lw \$a0 0(\$sp) # argc 004000001 8fa40000 lw \$4, 0(\$29) Cause 00400004] 27a50004 addiu \$5, \$29, 4 ; 184: addiu \$a1 \$sp 4 # argv BadVAddr = 0 00400008] 24a60004 addiu \$6, \$5, 4 ; 185: addiu \$a2 \$a1 4 # envp Status = 3000ff10 0040000c] 00041080 sll \$2, \$4, 2 ; 186: sll \$v0 \$a0 2 004000101 00c23021 addu \$6. \$6. \$2 : 187: addu \$a2 \$a2 \$v0 ; 188: jal main 00400014] 0c000000 jal 0x00000000 [main] LO 00400018] 00000000 nop ; 189: nop 0040001cl 3402000a ori \$2, \$0, 10 : 191: li \$v0 10 [00400020] 0000000c syscall ; 192: syscall # syscall 10 (exit) [at] = 0[v0] = 0Kernel Text Segment [80000000]..[80010000] [v1] = 080000180] 0001d821 addu \$27, \$0, \$1 ; 90: move \$k1 \$at # Save \$at [a0] = 1 80000184] 3c019000 lui \$1, -28672 : 92: sw Sv0 s1 # Not re-entrant and we can't trust Ssp [a1] = 7ffff860 [a2] = 7ffff868 800001881 ac220200 sw \$2. 512(\$1) 8000018c] 3c019000 lui \$1, -28672 ; 93: sw \$a0 s2 # But we need to use these registers [a3] = 0R8 [t0] = 0 [80000190] ac240204 sw \$4, 516(\$1) 800001941 401a6800 mfc0 \$26, \$13 ; 95: mfc0 \$k0 \$13 # Cause register R9 [t1] = 0 R10 [t2] = 0 80000198] 001a2082 srl \$4, \$26, 2 ; 96: srl \$a0 \$k0 2 # Extract ExcCode Field 8000019c] 3084001f andi \$4, \$4, 31 ; 97: andi \$a0 \$a0 0x1f R11 [t3] = 0R12 [t4] = 0 800001a0] 34020004 ori \$2, \$0, 4 ; 101: li \$v0 4 # syscall 4 (print str) R13 [t5] = 0; 102: la \$a0 \_\_m1\_ 800001a4] 3c049000 lui \$4, -28672 [ m1 ] R14 [t6] = 0 800001a81 0000000c syscall : 103: syscall R15 [t7] = 0 800001ac1 34020001 ori \$2, \$0, 1 ; 105: li \$v0 1 # syscall 1 (print int) R16 [s0] = 0; 106: srl \$a0 \$k0 2 # Extract ExcCode Field R17 [s1] = 0 800001b4] 3084001f andi \$4, \$4, 31 ; 107: andi \$a0 \$a0 0x1f R18 [s2] = 0 ; 108: syscall 800001b8] 0000000c syscall R19 [s3] = 0 800001bc] 34020004 ori \$2, \$0, 4 ; 110: li \$v0 4 # syscall 4 (print\_str) R20 [s4] = 0; 111: andi \$a0 \$k0 0x3c 800001c01 3344003c andi \$4, \$26, 60 R21 [s5] = 0; 112: lw \$a0 excp(\$a0) 800001c41 3c019000 lui \$1. -28672 R22 [s6] = 0800001c81 00240821 addu \$1. \$1. \$4 R23 [s7] = 0800001cc1 8c240180 lw \$4, 384(\$1) R24 [t8] = 0 800001d01 00000000 nop R25 [t9] = 0 800001d4] 0000000c syscall ; 114: syscall R26 [k0] = 0 800001d81 34010018 ori \$1. \$0. 24 ; 116: bne \$k0 0x18 ok\_pc # Bad PC exception requires special \$PIM Version 9.1.4 of September 4, 2011 Copyright 1990-2010, James R. Larus. \$PIM is distributed under a BSD license. see the file README for a full copyright notice

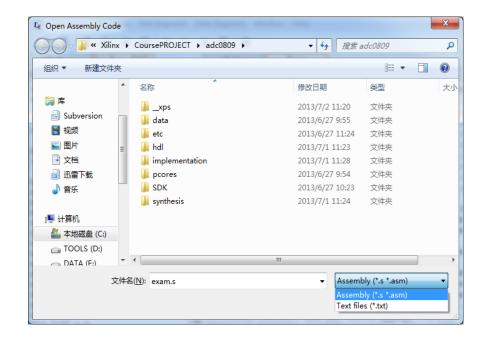
> > QtSpim消息

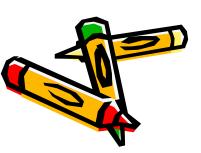


## File菜单

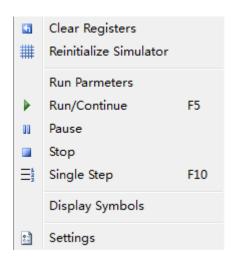


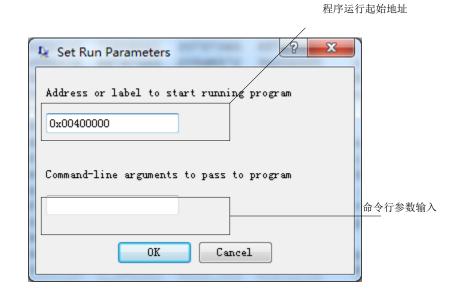




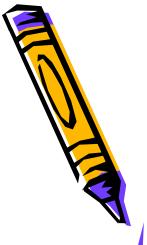


### Simulator菜单









## 汇编程序结构

```
数据段
  .data
   prompt: .asciiz "\n Please Input a value for N = "
   result: .asciiz " The sum of the integers from 1 to N is"
   bye: .asciiz "\n **** Adios Amigo - Have a good day ****"
  .globl main
  .text
main:
  li $v0, 4 # 打印字符串spim系统功能掉用号存入$v0
                                                        代码段
  la $a0, prompt # 将消息prompt地址存入 $a0
  syscall # 打印消息
  li $v0, 5 # 读取整数spim系统功能掉用号存入$v0
  syscall # 从console读入整数存入 $v0
  blez $v0, end # 如果 $v0 < = 0跳转到end标号处
  li $t0, 0 # 清除 $t0 为0
  add $t0, $t0, $v0 # 将和存入 $t0
  addi $v0, $v0, -1 # 輸入数据减1
  bnez $v0, loop # $v0!= 0, 继续循环
  li $v0, 4 # 打印字符串spim系统功能掉用号存入$v0
  la $a0, result # 将消息result地址存入 $a0
  syscall # 打印消息
  li $v0, 1 #打印整数spim系统功能掉用号存入$v0
  move $a0, $t0 #整数值存入 $a0
  syscall # 打印整数
  b main # 跳转到main标号
end: li $v0, 4 # 打印字符串spim系统功能掉用号存入$v0
  la $a0, bye # 将消息bye地址存入 $a0
  syscall #打印消息
  li $v0.10 # 退出程序系统功能调用
  syscall #返回到spim内核
```





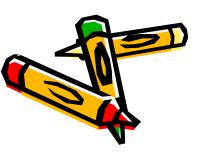
## QtSpim系统功能调用

•			
功能描述	功能	输入参数	输出参数
	号		
	(\$v0		
	)		
显示整数	1	\$a0:整数值	
显示字符串	4	\$a0:字符串首地址	
直到字符串			
结束符0			
读入整数	5		\$v0:输入的整数值
读入字符串	8	\$a0:内存空间首地	
		址	
		\$a1:内存空间长度	
退出	10		

## QtSpim汇编、调试程序示例

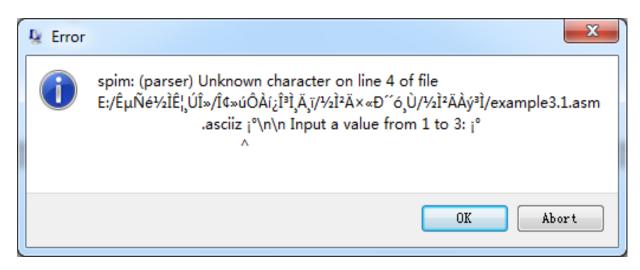
• 用户程序入口

```
.text
.globl __start
__start:
lw $a0, 0($sp) # argc
addiu $a1, $sp, 4 # argv
addiu $a2, $a1, 4 # envp
sll $v0, $a0, 2
addu $a2, $a2, $v0
jal main
li $v0,10
syscall # 退出
```

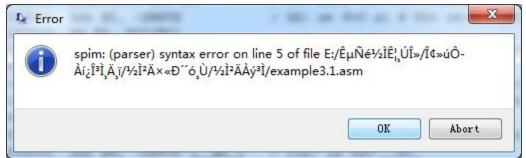


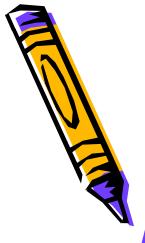
### 汇编提示信息

• 装载汇编源程序时完成









#### 查看内存映像

- · (MIPS)大字节序、
- · (PC)小字节序

变量名	地址	数据	定义值
jumptabl	0x10010000	0x24	main
e		0x00	
		0x40	
		0x00	
		0x60	case1
prompt	0x10010010	0x0a —	\n
		0x0a	\n
		0x20	space
		0x49	I



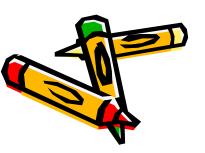
.align 2

jumptable: .word main, case1, case2, case3 prompt : .asciiz "\n\n Input a value from 1 to 3:

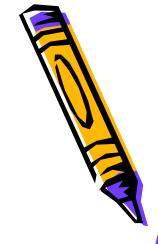




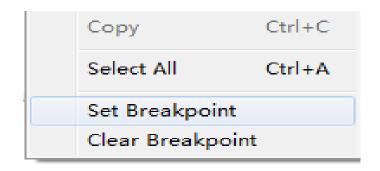
```
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
[10010000] 00400024 00400060 00400068 00400070 $ . @ . ` . @ . h . @ . p .
[10010010] 49259a0a 7475706e 76206120 65756c61 . . Input a val
[10010020] 6f726620 2031206d 33206f74 0000203a from 1 to 3:
[10010030]..[1003ffff] 00000000
```



# 设置断点、单步执行,修改寄存器值



• 光标处



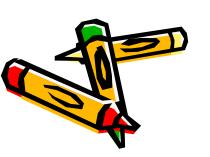
```
  [00400068] 00108080 sll $16, $16, 2 ; 22: sll $s0, $s0, 2 #xóÒÆ2λ
  [0040006c] 04010002 bgez $0 8 [output-0x0040006c]; 23: b output
```



## 实验任务

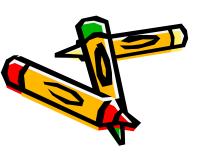
用汇编程序实现以下伪代码:要求采用移位指令实现 除法运算。

```
int main()
{ int K, Y;
int Z[50];
Y = 56;
for(K=0;K<50;K++)
Z[K] = Y - 16 * ( K / 4 + 210);
}</pre>
```



## 实验要求

- 完成汇编语言程序设计、调试、测试全过程
- 指出用户程序的内存映像,包括代码段和数据段
- 完成软件实验报告



#### 实验报告要求

- 1. 实验要求
- 2. 汇编源程序设计思路(算法)、源代码及注释
- 3. 详细的调试、测试过程(可以截图并附加说明)
- 4. 程序内存映像(最后结果Z[k](k=0~49)的内存映像需验收)
- 5. 心得体会

#### . DATA

Z: .SPACE 200

. TEXT

MAIN:

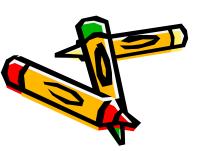
LA \$S3,Z

LI \$T0,0

LI \$T1,56

LI \$V0,10

SYSCALL



<b>戸</b> Ζ	0x0012fe60
— [0x0]	0xfffff318
— [0x1]	0xfffff318
— [0x2]	0xfffff318
— [0x3]	0xfffff318
— [0x4]	0xfffff308
— [0x5]	0xffffff308
— [0x6]	0xffffff308
- [0x7]	0xfffff308
- [0x8]	0xffffff2f8
— [0x9]	0xffffff2f8
— [0xa]	0xfffff2f8
— [0xb]	0xffffff2f8
— [0xc]	0xfffff2e8
— [0xd]	0xffffff2e8
— [0xe]	0xffffff2e8
— [0xf]	0xffffff2e8
- [0x10]	0xfffff2d8
— [0x11]	0xfffff2d8
− [0x12]	0xffffff2d8
— [0x13]	0xffffff2d8
— [0x14]	0xffffff2c8
− [0x15]	0xfffff2c8
— [0x16]	0xffffff2c8
_ [0x17]	0xffffff2c8
— [0x18]	0xfffff2b8
— [0x19]	0xfffff2b8
— [0x1a]	0xfffff2b8
— [0x1b]	0xfffff2b8
_ [0x1c]	0xffffff2a8
_ [0x1d]	0xffffff2a8
_ [0x1e]	0xffffff2a8
_ [0x1f]	0xffffff2a8
- [0x20]	0xfffff298
_ [0x21]	0xfffff298
_ [0×22]	0xfffff298
_ [0x23]	0xfffff298