APPENDIX D

Macro Instructions

Name	Actual Code	Space/Time
Absolute Value:		

abs Rd, Rs	addu Rd, $0, Rs
	bgez Rs, 1
	sub Rd, $0, Rs
		3/3

Branch if Equal to Zero:		
beqz Rs, Label	beq Rs, $0, Label	1/1
Branch if Greater than or Equal:		
bge Rs, Rt, Label	slt $at, Rs, Rt beq $at, $0, Label	2/2
If Reg.File[Rs] > = Reg.File[Rt] branch to Label Used to compare values represented in the two's complement number system.		
Branch if Greater than or Equal Unsigned		
bgeu Rs, Rt, Label	sltu $at, Rs, Rt beq $at, $0, Label	2/2
If Reg.File[Rs] > = Reg.File[Rt] branch to Label Used to compare addresses (unsigned values).		
Branch if Greater Than:		
bgt Rs, Rt, Label	slt $at, Rt, Rs bne $at, $0, Label	2/2
If Reg.File[Rs] > Reg.File[Rt] branch to Label Used to compare values represented in the two's complement number system.		
Branch if Greater Than Unsigned:		
bgtu Rs, Rt, Label	sltu $at, Rt, Rs bne $at, $0, Label	2/2
If Reg.File[Rs] > Reg.File[Rt] branch to Label Used to compare addresses (unsigned values).		
Branch if Less Than or Equal:		
ble Rs, Rt, Label	slt $at, Rt, Rs beq $at, $0, Label	2/2
If Reg.File[Rs] < = Reg.File[Rt] branch to Label Used to compare values represented in the two's complement number system.		

Branch if Less Than or Equal Unsigned:		
bleu Rs, Rt, Label	sltu $at, Rt, Rs beq $at, $0, Label	2/2
If Reg.File[Rs] < = Reg.File[Rt] branch to Label Used to compare addresses (unsigned values).		
Branch if Less Than:		
blt Rs, Rt, Label	slt $at, Rs, Rt bne $at, $0, Label	2/2
If Reg.File[Rs] < Reg.File[Rt] branch to Label Used to compare values represented in the two's complement number system.		
Branch if Less Than Unsigned:		
bltu Rs, Rt, Label	sltu $at, Rs, Rt bne $at, $0, Label	2/2
If Reg.File[Rs] < Reg.File[Rt] branch to Label Used to compare addresses (unsigned values).		
Branch if Not Equal to Zero:		
bnez Rs, Label	bne Rs, $0, Label	1/1
Branch Unconditional		
b Label	bgez $0, Label	1/1
Divide:		
div Rd, Rs, Rt	bne Rt, $0, ok break $0 ok: div Rs, Rt mflo Rd	4/41
Divide Unsigned:		
divu Rd, Rs, Rt	bne Rt, $0, ok break $0 ok: divu Rs, Rt mflo Rd	4/41
Load Address:		
la Rd, Label	lui $at, Upper 16-bits of Label ori Rd, $at, Lower 16-bits of Label	2/2
Used to initialize pointers.		
Load Immediate:		
li Rd, value	lui $at, Upper 16-bits of value ori Rd, $at, Lower 16-bits of value	2/2
Initialize registers with negative constants and values greater than 32,767.		
Load Immediate:		
li Rd, value	ori Rt, $0, value	1/1
Initialize registers with positive constants less than 32,768.		
Move:		
move Rd, Rs	addu Rd, $0, Rs	1/1
mul Rd, Rs, Rt	mult Rs, Rt mflo Rd	2/33
Multiply (with overflow exception):		
mulo Rd, Rs, Rt	mult Rs, Rt mfhi $at mflo Rd sra Rd, Rd, 31 beq $at, Rd, ok	7/37

Unaligned Load Halfword:		
ulhu Rd, 3(Rs)	lbu Rd, 4(Rs) lbu $at, 3(Rs) sll Rd, Rd, 8 or Rd, Rd, $at	*4/4*
Unaligned Load Word:		
ulw Rd, 3(Rs)	lwl Rd, 6(Rs) lwr Rd, 3(Rs)	*2/2*
Unaligned Store Halfword:		
ush Rd, 3(Rs)	sb Rd, 3(Rs) srl $at, Rd, 8 sb $at, 4(Rs)	*3/3*
Unaligned Store Word:		
usw Rd, 3(Rs)	swl Rd, 6(Rs) swr Rd, 3(Rs)	*2/2*

APPENDIX E

A Modified Trap Handler

```
#############################################################################
# SPIM S20 MIPS simulator.
# A modified trap handler that responds to keyboard interrupts.
#############################################################################
# Copyright (C) 1990-2000 James Larus, larus@cs.wisc.edu.
# ALL RIGHTS RESERVED.
#
# SPIM is distributed under the following conditions:
#
# You may make copies of SPIM for your own use and modify those copies.
#
# All copies of SPIM must retain my name and copyright notice.
#
# You may not sell SPIM or distributed SPIM in conjunction with a
# commercial product or service without the expressed written
# consent of James Larus.
#
# THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR
# IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED
# WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
# PURPOSE.
#############################################################################

        .kdata
__m1_:  .asciiz "  Exception "
__m2_:  .asciiz " occurred and ignored\n"
__e0_:  .asciiz "  [Interrupt] "
__e1_:  .asciiz ""
__e2_:  .asciiz ""
__e3_:  .asciiz ""
__e4_:  .asciiz "  [Unaligned address in inst/data fetch] "
__e5_:  .asciiz "  [Unaligned address in store] "
__e6_:  .asciiz "  [Bad address in text read] "
__e7_:  .asciiz "  [Bad address in data/stack read] "
```