Brief Introduction of RISC-V ISA

Spring 2020



Outline

- Homework Overview
- Review Concept of ISA
- RISC-V ISA for Homework3



Homework Overview

- Implementing subset of C grammar for RISC-V platform
- Grading: 30%
 - HW1 scanner (use lex tool)
 - 10%
 - HW2 parser (use yacc + lex tool)
 - 10%
 - HW3 code generator (yacc + lex + RISC-V)
 - 10% (Demo)



Review-Basic Concept of ISA

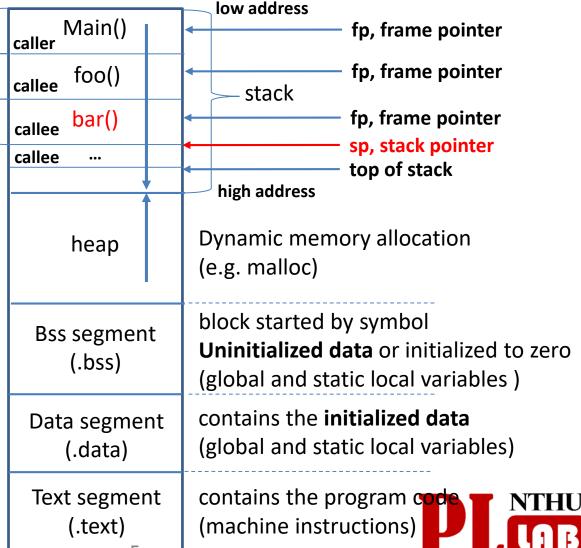
- Computational Instructions
 - op \$rd, \$rs, \$rt
 - op \$rd, \$rs, constant (Immediate mode)
- Data Transfer Instructions
 - lw \$rd, offset(\$rs)
 - sw \$rs, offset(\$rd)
- Program Control
 - Jal label
- op: opcode
- \$rd: destination register
- \$rs: 1st operand of source register
- \$rt: 2nd operand of target register



Review-File Format and Call Stack

stack frame of function 1
stack frame of function 2
stack frame of function 3

- ELF (Executable and Linking Format)
 - Define is a common standard file format for executable files



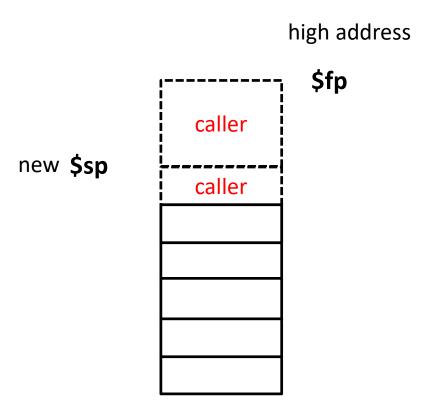
Review-Function Prologue

- Function starts working
 - create a stack frame for itself (change \$fp and \$sp)
 - save the return address in the stack frame
 - save any registers that it plans to change
- Changing \$fp and \$sp ...

```
new fp = old p - 4
```

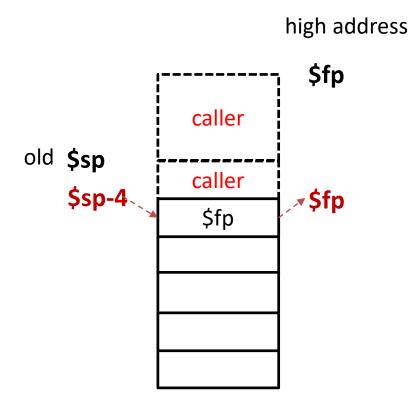
new \$sp = old \$sp - size of frame (in bytes)





low address

Foo (): ! Prologue sw fp, -4(sp)sw \$ra, -8(\$sp) sw \$s0, -12(\$sp)sw \$s1, -16(\$sp) sw \$s2, -20(\$sp)addi \$sp, \$sp, -20 ! End of Prologue

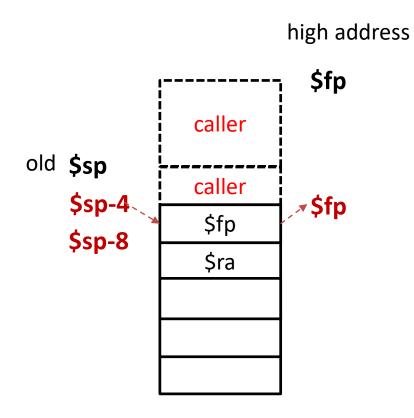


low address

Foo ():
! Prologue

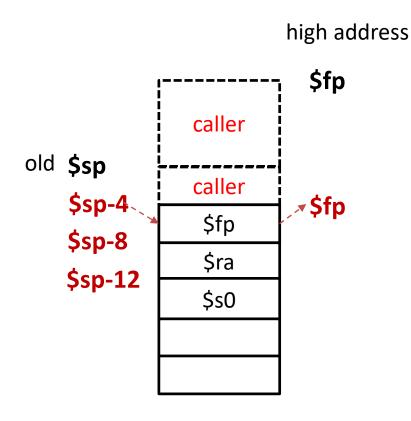
! End of Prologue





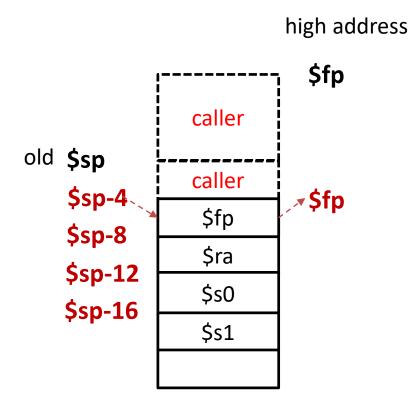
low address

Foo (): ! Prologue sw fp, -4(sp)sw \$ra, -8(\$sp) sw \$s0, -12(\$sp) sw \$s1, -16(\$sp)sw \$s2, -20(\$sp)addi \$sp, \$sp, -20 ! End of Prologue



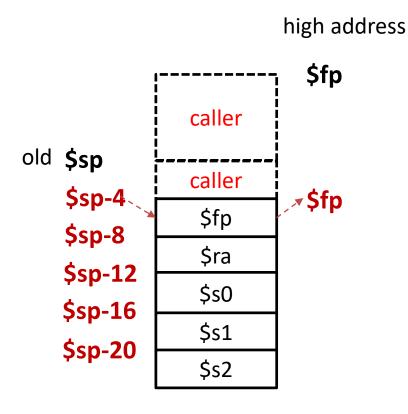
low address

Foo (): ! Prologue sw fp, -4(sp)sw \$ra, -8(\$sp) sw \$s0, -12(\$sp) sw \$s1, -16(\$sp) sw \$s2, -20(\$sp)addi \$sp, \$sp, -20 ! End of Prologue



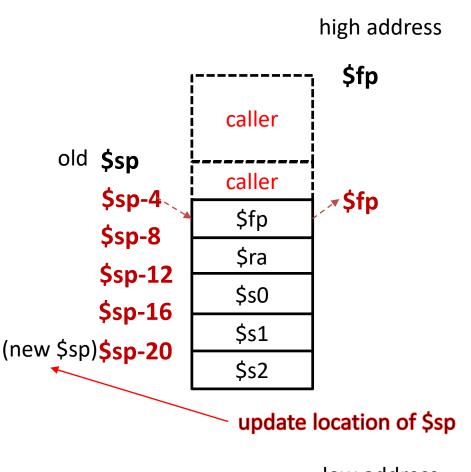
low address

Foo (): ! Prologue sw fp, -4(sp)sw \$ra, -8(\$sp) sw \$s0, -12(\$sp) sw \$s1, -16(\$sp) sw \$s2, -20(\$sp)addi \$sp, \$sp, -20 ! End of Prologue



low address

Foo (): ! Prologue sw fp, -4(sp)sw \$ra, -8(\$sp) sw \$s0, -12(\$sp)sw \$s1, -16(\$sp)sw \$s2, -20(\$sp)addi \$sp, \$sp, -20 ! End of Prologue



low address

$$sw $s2, -20(\$sp)$$

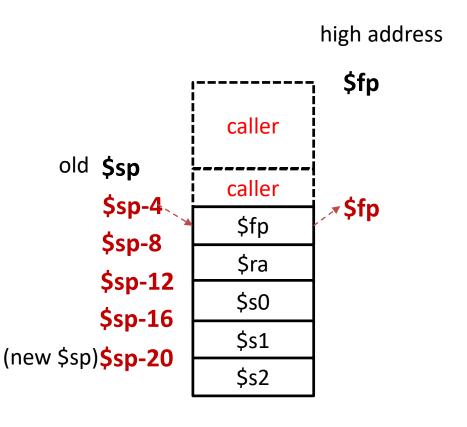
! End of Prologue



Review-Function Epilogue

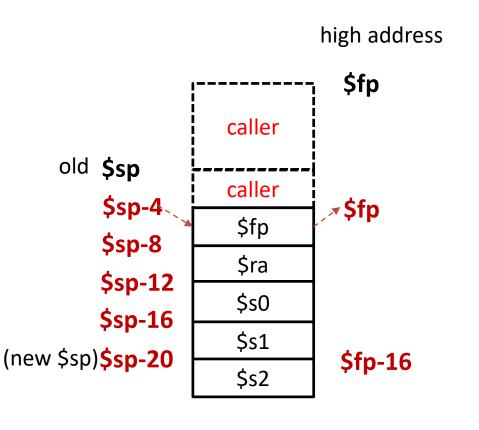
- Function end
 - pop any pushed arguments off the stack
 - restore the values of any saved registers
 - restore the saved value of \$ra (return address)
 - remove its stack frame (change \$fp and \$sp)
- Locations of saved values computed relative to \$fp
- Changing \$fp and \$sp ...
 new \$sp = old \$fp + 4
 new \$fp = memory[old \$fp]

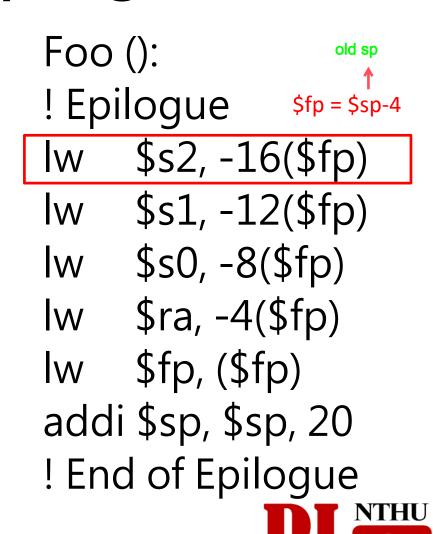


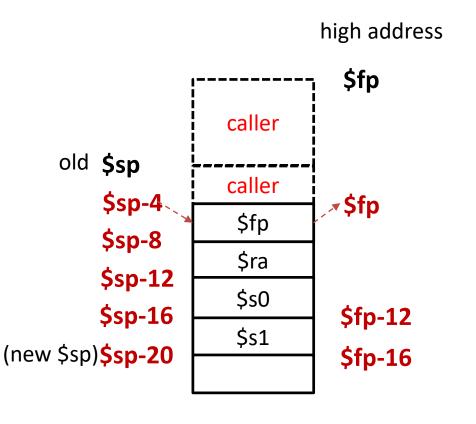


low address

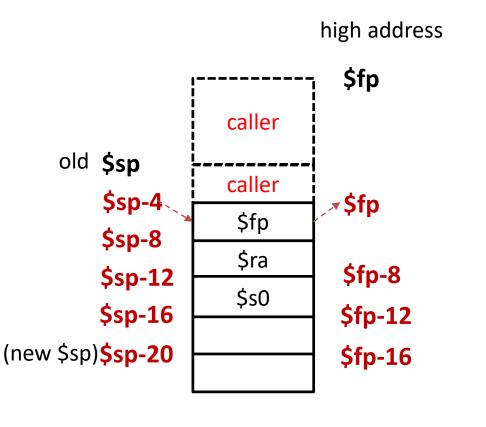
Foo (): ! Epilogue \$s2, -16(\$fp) lw \$s1, -12(\$fp) lw lw \$s0, -8(\$fp) lw \$ra, -4(\$fp) \$fp, (\$fp) lw addi \$sp, \$sp, 20 ! End of Epilogue



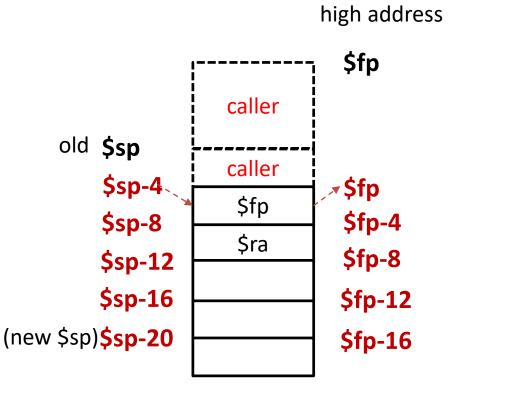




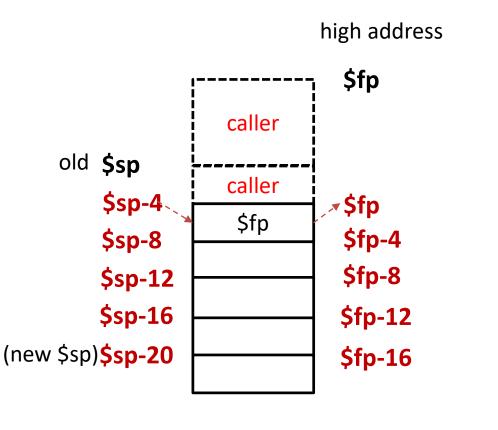
```
Foo ():
! Epilogue
              fp = sp-4
     $s2, -16($fp)
lw
     $s1, -12($fp)
lw
     $s0, -8($fp)
lw
     $ra, -4($fp)
lw
     $fp, ($fp)
lw
addi $sp, $sp, 20
! End of Epilogue
```



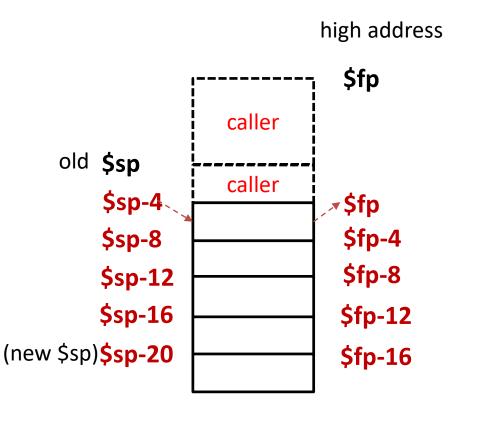
```
Foo ():
! Epilogue
             fp = sp-4
     $s2, -16($fp)
lw
     $s1, -12($fp)
lw
     $s0, -8($fp)
lw
     $ra, -4($fp)
lw
     $fp, ($fp)
lw
addi $sp, $sp, 20
! End of Epilogue
```



```
Foo ():
! Epilogue
             fp = sp-4
     $s2, -16($fp)
lw
     $s1, -12($fp)
lw
     $s0, -8($fp)
lw
     $ra, -4($fp)
lw
     $fp, ($fp)
lw
addi $sp, $sp, 20
! End of Epilogue
```



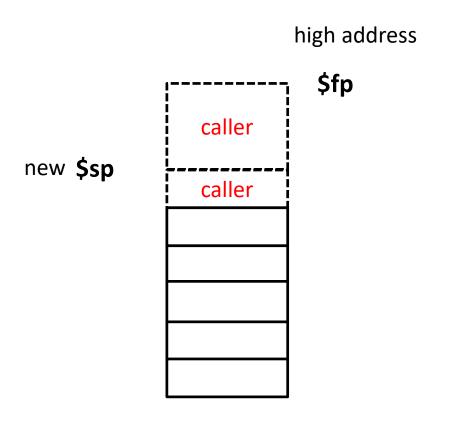
```
Foo ():
! Epilogue
              fp = sp-4
     $s2, -16($fp)
lw
     $s1, -12($fp)
lw
     $s0, -8($fp)
lw
     $ra, -4($fp)
lw
     $fp, ($fp)
lw
addi $sp, $sp, 20
! End of Epilogue
```



low address

Foo (): ! Epilogue fp = sp-4\$s2, -16(\$fp) lw \$s1, -12(\$fp) lw \$s0, -8(\$fp) lw \$ra, -4(\$fp) lw \$fp, (\$fp) lw addi \$sp, \$sp, 20

! End of Epilogue



Foo (): ! Epilogue \$fp = \$sp-4 \$s2, -16(\$fp) lw lw \$s1, -12(\$fp) lw \$s0, -8(\$fp) lw \$ra, -4(\$fp) \$fp, (\$fp) lw addi \$sp, \$sp, 20

! End of Epiloque

Introduction to RISC-V RV321 ISA Available for HW3



RISC-V

- RISC-V (pronounced "risk-five")
 - RISC-I, II, III, IV before
- Originated in 2010 by researchers at UC Berkeley
 - Krste Asanović, David Patterson and students
- Features
 - Full Free open-source
 - Allowing for user extensibility of the architecture
- Andes F1 board only support RV32I ISA of RISC-V

RISC-V General Purpose Registers

• 32 32-bit General Purpose Registers

Register	ABI Name	Description	Saver
x0	zero	Hard-wired zero	
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
х3	gp	Global pointer	
x4	tp	Thread pointer	
x5-7	t0-2	Temporaries	Caller
x8	s0/fp	Saved register/frame pointer	Callee
x9	s1	Saved register	Callee
x10-11	a0-1	Function arguments/return values	Caller
x12-17	a2-7	Function arguments	Caller
x18-27	s2-11	Saved registers	Callee
x28-31	t3-6	Temporaries	Caller
f0-7	ft0-7	FP temporaries	Caller
f8-9	fs0-1	FP saved registers	Callee
f10-11	fa0-1	FP arguments/return values	Caller
f12-17	fa2-7	FP arguments	Caller
f18-27	fs2-11	FP saved registers	Callee
f28-31	ft8-11	FP temporaries	Caller



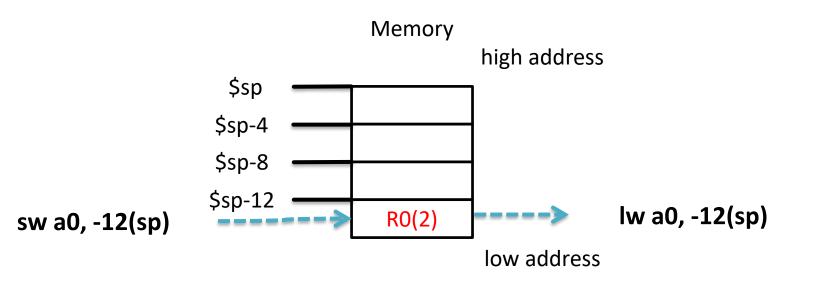
Arithmetic ISA

指令	寫法	說明
add	add a0, a1, a2	a0 = a1 + a2
addi	addi a0, a1, 3	a0 = a1 + 3
sub	sub a0, a1, a2	a0 = a1 - a2
mul	mul a0, a1, a2	a0 = a1 * a2
div	div a0, a1, a2	a0 = a1 / a2
rem	rem a0, a1, a2	a0 = a1 % a2



Load/Store ISA

指令	寫法	說明
Li	li a0, 2	r0 = 2
lw	lw a0, -12(sp)	Load data from address sp-12 and store it into a0.
SW	sw a0, -12(sp)	Store a0 into address sp-12.



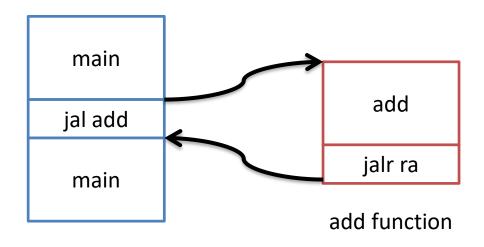
Call stack



JMUP ISA

指令	寫法	說明
jal	jal add	Jump to 'add' function and link
jalr	jalr ra	Return from function.

Program



main function



Conditional Branch ISA

指令	寫法		說明
beq	beq	a0, a1, offset	if a0 == a1 then jump
bne	bne	a0, a1, offset	if a0 != a1 then jump
blt	blt	a0, a1, offset	if a0 < a1 then jump
bge	bge	a0, a1, offset	if a0 >= a1 then jump



```
.global codegen
     codegen:
      // BEGIN PROLOGUE
       // codegen is the callee here, so we save callee-saved registers
       sw s0, -4(sp) // save frame pointer
       addi sp, sp, -4
       addi s0, sp, 0 // set new frame
       sw sp, -4(s0)
       sw s1, -8(s0)
       sw s2, -12(s0)
11
       sw s3, -16(s0)
       sw s4, -20(s0)
13
       sw s5, -24(s0)
       sw s6, -28(s0)
      sw s7, -32(s0)
                                                                                                  void codegen();
       sw s8, -36(s0)
       sw s9, -40(s0)
                                                                                                  void codegen()
       sw s10, -44(s0)
       sw s11, -48(s0)
       addi sp, s0, -48 // update stack pointer
       // END PROLOGUE
                                                                                                      int a = 1 + 2 * 1; // a = 3
                                li t0, 1
                                                                li t0, 0
23
       li t0, 1
       sw t0, -4(sp)
                        71
                                 sw t0, -4(sp)
                                                      108
                                                                sw t0, -4(sp)
                                                                                                      int b = (a + 3) / 2; // b = 3
25
       addi sp, sp, -4
                                 addi sp, sp, -4
                                                                addi sp, sp, -4
       li t0, 2
                        73
                                li t0, 26
                                                      110
                                                                li t0, 26
       sw t0, -4(sp)
                                                                                                      digitalWrite(26, 1);
                        74
                                sw t0, -4(sp)
                                                      111
                                                                sw t0, -4(sp)
28
       addi sp, sp, -4
                        75
                                addi sp, sp, -4
                                                      112
                                                                addi sp, sp, -4
29
       li t0, 1
                                                                                                      delay(a * 1000); // delay 3 seconds
       sw t0, -4(sp)
30
                        76
                                lw a0, 0(sp)
                                                      113
                                                                lw a0, 0(sp)
31
       addi sp, sp, -4
                                 addi sp, sp, 4
                                                      114
                                                                addi sp, sp, 4
                        77
       lw t0, 0(sp)
                                                                                                      digitalWrite(26, 0);
                        78
                                lw a1, 0(sp)
                                                      115
                                                                lw a1, 0(sp)
       addi sp, sp, 4
                        79
                                 addi sp, sp, 4
                                                      116
                                                                addi sp, sp, 4
       lw t1, 0(sp)
                                                                                                      delay(b * 1000); // delay 3 seconds
       addi sp, sp, 4
                                 sw ra, -4(sp)
                                                      117
                                                                sw ra, -4(sp)
       mul t0, t0, t1
                                                      118
                                                                addi sp, sp, -4
                        81
                                 addi sp, sp, -4
37
       sw t0, -4(sp)
                                jal ra, digitalWrite 119
                                                                jal ra, digitalWrite
                        82
                                                                                        10
       addi sp, sp, -4
                        83
                                lw ra, 0(sp)
                                                                lw ra, 0(sp)
39
       lw t0, 0(sp)
       addi sp, sp, 4
                                                                addi sp, sp, 4
                                 addi sp, sp, 4
                                                      121
       lw t1, 0(sp)
                                                      122
       addi sp, sp, 4
                                                      123
                                                                lw t0, -56(s0)
                        86
                                lw t0, -52(s0)
       add t0, t0, t1
                                                      124
                                                                sw t0, -4(sp)
                                 sw t0, -4(sp)
       sw t0, -4(sp)
       addi sp, sp, -4
                                 addi sp, sp, -4
                                                      125
                                                                addi sp, sp, -4
                                                                                             145
                                                                                                      // BEGIN EPILOGUE
       lw t0, -52(s0)
                                                                li t0, 1000
                                li t0, 1000
                                                      126
                                                                                             146
                                                                                                      // restore callee-saved registers
       sw t0, -4(sp)
                                                                                             147
                                                                                                      // s0 at this point should be the same as in prologue
                        90
                                 sw t0, -4(sp)
                                                      127
                                                                sw t0, -4(sp)
       addi sp, sp, -4
                                                                                             148
                                                                                                      lw s11, -48(s0)
                        91
                                 addi sp, sp, -4
                                                      128
                                                                addi sp, sp, -4
       li t0, 3
                                                                                              149
                                                                                                      lw s10, -44(s0)
                                                      129
                                                                lw t0, 0(sp)
       sw t0, -4(sp)
                                lw t0, 0(sp)
                                                                                             150
                                                                                                      lw s9, -40(s0)
51
       addi sp, sp, -4
                                 addi sp, sp, 4
                                                      130
                                                                addi sp, sp, 4
                        93
                                                                                                      lw s8, -36(s0)
                                                                                             151
52
       lw t0, 0(sp)
                                                                lw t1, 0(sp)
                                lw t1, 0(sp)
                                                      131
53
       addi sp, sp, 4
                                                                                              152
                                                                                                      lw s7, -32(s0)
                                 addi sp, sp, 4
                                                      132
                                                                addi sp, sp, 4
       lw t1, 0(sp)
                                                                                             153
                                                                                                      lw s6, -28(s0)
                                 mul t0, t0, t1
                                                      133
                                                                mul t0, t0, t1
55
       addi sp, sp, 4
                                                                                                      lw s5, -24(s0)
                                                                                             154
       add t0, t0, t1
                                                      134
                                                                sw t0, -4(sp)
                                 sw t0, -4(sp)
                                                                                             155
                                                                                                      lw s4, -20(s0)
       sw t0, -4(sp)
                                                      135
                                                                addi sp, sp, -4
                                 addi sp, sp, -4
                                                                                                      lw s3, -16(s0)
                                                                                             156
       addi sp, sp, -4
                                                                lw a0, 0(sp)
                                                      136
                                lw a0, 0(sp)
                                                                                             157
                                                                                                      lw s2, -12(s0)
       li t0, 2
                                 addi sp, sp, 4
                                                      137
                                                                addi sp, sp, 4
                                                                                             158
                                                                                                      lw s1, -8(s0)
                                                                                                                                                             NTHU
60
       sw t0, -4(sp)
                       100
       addi sp, sp, -4
                                                      138
                                                                sw ra, -4(sp)
                                                                                             159
                                                                                                      lw sp, -4(s0)
                                sw ra, -4(sp)
       lw t0, 0(sp)
                                                                                                      addi sp, sp, 4
                                                      139
                                                                addi sp, sp, -4
                       102
                                 addi sp, sp, -4
63
       addi sp, sp, 4
                                                                                             161
                                                                                                      lw s0, -4(sp)
                                                      140
                                                                jal ra, delay
                                jal ra, delay
       lw t1, 0(sp)
                                                                                             162
                                                                                                      // END EPILOGUE
                                                      141
                                                                lw ra, \theta(sp)
       addi sp, sp, 4
                                lw ra, 0(sp)
                                                                                      30
                                                                                             163
```

164

jalr zero, 0(ra) // return

addi sp, sp, 4

142

143

144

addi sp, sp, 4

div t0, t1, t0

addi sp, sp, -4

sw t0, -4(sp)

```
1
      .global codegen
     codegen:
       // BEGIN PROLOGUE
       // codegen is the callee here, so we save callee-saved registers
       sw s0, -4(sp) // save frame pointer
                                                       void codegen();
       addi sp, sp, -4
       addi s0, sp, 0 // set new frame
                                                       void codegen()
       sw sp, -4(s0)
                                                   3
       sw s1, -8(s0)
                                                          int a = 1 + 2 * 1; // a = 3
       sw s2, -12(s0)
10
                                                          int b = (a + 3) / 2; // b = 3
11
       sw s3, -16(s0)
12
       sw s4, -20(s0)
                                                          digitalWrite(26, 1);
       sw s5, -24(s0)
13
                                                          delay(a * 1000); // delay 3 seconds
       sw s6, -28(s0)
14
                                                          digitalWrite(26, 0);
       sw s7, -32(s0)
15
       sw s8, -36(s0)
16
                                                          delay(b * 1000); // delay 3 seconds
       sw s9, -40(s0)
17
                                                  10
       sw s10, -44(s0)
18
       sw s11, -48(s0)
19
20
        addi sp, s0, -48 // update stack pointer
21
       // END PROLOGUE
```



```
1
      .global codegen
     codegen:
       // BEGIN PROLOGUE
       // codegen is the callee here, so we save callee-saved registers
        sw s0, =10-4(sp) // save frame pointer
                                                        void codegen();
       addi sp, sp, -4
       addi s0, sp, 0 // set new frame
                                                        void codegen()
       sw sp, -4(s0)
       sw s1, -8(s0)
                                                          int a = 1 + 2 * 1; // a = 3
                                                   4
       sw s2, -12(s0)
10
                                                          int b = (a + 3) / 2; // b = 3
11
       sw s3, -16(s0)
12
       sw s4, -20(s0)
                                                          digitalWrite(26, 1);
       sw s5, -24(s0)
13
                                                          delay(a * 1000); // delay 3 seconds
14
       sw s6, -28(s0)
                                                          digitalWrite(26, 0);
       sw s7, -32(s0)
15
       sw s8, -36(s0)
16
                                                          delay(b * 1000); // delay 3 seconds
       sw s9, -40(s0)
17
                                                  10
       sw s10, -44(s0)
18
       sw s11, -48(s0)
19
        addi sp, s0, -48
                           update stack pointer
20
       // END PROLOGUE
21
```



```
23
24
        sw t0, -4(sp)
25
        addi sp, sp, -4
26
        li t0, 2
27
        sw t0, -4(sp)
28
        addi sp, sp, -4
        li t0, 1
29
        sw t0, -4(sp)
30
31
        addi sp, sp, -4
        lw t0, 0(sp)
32
        addi sp, sp, 4
33
        lw t1, 0(sp)
34
35
        addi sp, sp, 4
        mul t0, t0, t1
36
37
        sw t0, -4(sp)
38
        addi sp, sp, -4
        lw t0, 0(sp)
39
        addi sp, sp, 4
40
        lw t1, 0(sp)
41
42
        addi sp, sp, 4
43
        add t0, t0, t1
44
        sw t0, -4(sp)
        addi sp, sp, -4
45
        lw t0, -52(s0)
46
47
        sw t0, -4(sp)
        addi sp, sp, -4
49
        li t0, 3
        sw t0, -4(sp)
50
        addi sp, sp, -4
51
        lw t0, 0(sp)
52
53
        addi sp, sp, 4
        lw t1, 0(sp)
54
55
        addi sp, sp, 4
        add t0, t0, t1
        sw t0, -4(sp)
57
        addi sp, sp, -4
        li t0, 2
59
        sw t0, -4(sp)
61
        addi sp, sp, -4
        lw t0, 0(sp)
62
        addi sp, sp, 4
63
        lw t1, 0(sp)
64
65
        addi sp, sp, 4
        div t0, t1, t0
        sw t0, -4(sp)
67
        addi sp, sp, -4
```

```
void codegen();
void codegen()

{
    int a = 1 + 2 * 1; // a = 3
    int b = (a + 3) / 2; // b = 3
    digitalWrite(26, 1);
    delay(a * 1000); // delay 3 seconds
    digitalWrite(26, 0);
    delay(b * 1000); // delay 3 seconds
}
```



```
23
24
        sw t0, -4(sp)
25
        addi sp, sp, -4
        li t0, 2
27
        sw t0, -4(sp)
28
        addi sp, sp, -4
        li t0, 1
29
        sw t0, -4(sp)
30
31
        addi sp, sp, -4
        lw t0, 0(sp)
32
        addi sp, sp, 4
33
        lw t1, 0(sp)
        addi sp, sp, 4
35
        mul t0, t0, t1
36
        sw t0, -4(sp)
        addi sp, sp, -4
38
        lw t0, 0(sp)
39
        addi sp, sp, 4
40
        lw t1, 0(sp)
41
        addi sp, sp, 4
42
43
        add t0, t0, t1
        sw t0, -4(sp)
44
        addi sp, sp, -4
45
        lw t0, -52(s0)
46
47
        sw t0, -4(sp)
        addi sp, sp, -4
48
49
       li t0, 3
        sw t0, -4(sp)
50
        addi sp, sp, -4
51
        lw t0, 0(sp)
52
53
        addi sp, sp, 4
54
        lw t1, 0(sp)
        addi sp, sp, 4
55
56
        add t0, t0, t1
        sw t0, -4(sp)
57
        addi sp, sp, -4
58
        li t0, 2
59
        sw t0, -4(sp)
60
       addi sp, sp, -4
61
        lw t0, 0(sp)
62
        addi sp, sp, 4
63
       lw t1, 0(sp)
64
        addi sp, sp, 4
65
66
        div t0, t1, t0
67
        sw t0, -4(sp)
68
        addi sp, sp, -4
```

```
void codegen();
void codegen()

{
   int a = 1 + 2 * 1; // a = 3
   int b = (a + 3) / 2; // b = 3
   digitalWrite(26, 1);
   delay(a * 1000); // delay 3 seconds
   digitalWrite(26, 0);
   delay(b * 1000); // delay 3 seconds
}
```



```
li t0, 1
sw t0, -4(sp)
addi sp, sp, -4
li t0, 26
sw t0, -4(sp)
addi sp, sp, -4
lw a0, 0(sp)
addi sp, sp, 4
lw a1, 0(sp)
addi sp, sp, 4
sw ra, -4(sp)
addi sp, sp, -4
jal ra, digitalWrite
lw ra, 0(sp)
addi sp, sp, 4
lw t0, -52(s0)
sw t0, -4(sp)
addi sp, sp, -4
li t0, 1000
sw t0, -4(sp)
addi sp, sp, -4
lw t0, 0(sp)
addi sp, sp, 4
lw t1, 0(sp)
addi sp, sp, 4
mul t0, t0, t1
sw t0, -4(sp)
addi sp, sp, -4
lw a0, 0(sp)
addi sp, sp, 4
sw ra, -4(sp)
addi sp, sp, -4
jal ra, delay
lw ra, 0(sp)
addi sp, sp, 4
```

69

70

71 72

73

74

75

76

77

78

79

80

81

82

83

84 85

86

87

88

89

90

91 92

93

94 95

96

97

98 99

100

101 102

103

104

```
void codegen();
void codegen()

{
   int a = 1 + 2 * 1; // a = 3
   int b = (a + 3) / 2; // b = 3
   digitalWrite(26, 1);

delay(a * 1000); // delay 3 seconds
   digitalWrite(26, 0);
   delay(b * 1000); // delay 3 seconds
}
```



```
li t0, 1
sw t0, -4(sp)
addi sp, sp, -4
li t0, 26
sw t0, -4(sp)
addi sp, sp, -4
lw a0, 0(sp)
addi sp, sp, 4
lw a1, 0(sp)
addi sp, sp, 4
sw ra, -4(sp)
addi sp, sp, -4
jal ra, digitalWrite
lw ra, 0(sp)
addi sp, sp, 4
lw t0, -52(s0)
sw t0, -4(sp)
addi sp, sp, -4
li t0, 1000
sw t0, -4(sp)
addi sp, sp, -4
lw t0, 0(sp)
addi sp, sp, 4
lw t1, 0(sp)
addi sp, sp, 4
mul t0, t0, t1
sw t0, -4(sp)
addi sp, sp, -4
lw a0, 0(sp)
addi sp, sp, 4
sw ra, -4(sp)
addi sp, sp, -4
jal ra, delay
lw ra, 0(sp)
addi sp, sp, 4
```

69 70

71 72

73 74

75 76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91 92

93

94 95

96

97

98 99

100

101 102

103

104

```
void codegen();
void codegen()

{
   int a = 1 + 2 * 1; // a = 3
   int b = (a + 3) / 2; // b = 3
   digitalWrite(26, 1);

delay(a * 1000); // delay 3 seconds

digitalWrite(26, 0);
delay(b * 1000); // delay 3 seconds

delay(b * 1000); // delay 3 seconds
}
```



```
li t0, 0
107
         sw t0, -4(sp)
108
         addi sp, sp, -4
109
         li t0, 26
110
         sw t0, -4(sp)
111
112
         addi sp, sp, -4
         lw a0, 0(sp)
113
         addi sp, sp, 4
114
         lw a1, 0(sp)
115
         addi sp, sp, 4
116
117
         sw ra, -4(sp)
         addi sp, sp, -4
118
         jal ra, digitalWrite
119
         lw ra, 0(sp)
120
121
         addi sp, sp, 4
122
         lw t0, -56(s0)
123
124
         sw t0, -4(sp)
125
         addi sp, sp, -4
         li t0, 1000
126
127
         sw t0, -4(sp)
         addi sp, sp, -4
128
         lw t0, 0(sp)
129
         addi sp, sp, 4
130
131
         lw t1, 0(sp)
132
         addi sp, sp, 4
         mul t0, t0, t1
133
134
         sw t0, -4(sp)
135
         addi sp, sp, -4
136
         lw a0, 0(sp)
         addi sp, sp, 4
137
138
         sw ra, -4(sp)
139
         addi sp, sp, -4
140
         jal ra, delay
         lw ra, 0(sp)
141
142
         addi sp, sp, 4
```

```
void codegen();
void codegen()

int a = 1 + 2 * 1; // a = 3
int b = (a + 3) / 2; // b = 3
digitalWrite(26, 1);
delay(a * 1000); // delay 3 seconds
digitalWrite(26, 0);

delay(b * 1000); // delay 3 seconds

delay(b * 1000); // delay 3 seconds
}
```



```
li t0, 0
107
         sw t0, -4(sp)
108
         addi sp, sp, -4
109
         li t0, 26
110
111
         sw t0, -4(sp)
112
         addi sp, sp, -4
         lw a0, 0(sp)
113
         addi sp, sp, 4
114
         lw a1, 0(sp)
115
         addi sp, sp, 4
116
117
         sw ra, -4(sp)
         addi sp, sp, -4
118
         jal ra, digitalWrite
119
         lw ra, 0(sp)
120
121
         addi sp, sp, 4
122
         lw t0, -56(s0)
123
124
         sw t0, -4(sp)
125
         addi sp, sp, -4
         li t0, 1000
126
         sw t0, -4(sp)
127
         addi sp, sp, -4
128
         lw t0, 0(sp)
129
         addi sp, sp, 4
130
131
         lw t1, 0(sp)
132
         addi sp, sp, 4
         mul t0, t0, t1
133
134
         sw t0, -4(sp)
135
         addi sp, sp, -4
136
         lw a0, 0(sp)
         addi sp, sp, 4
137
138
         sw ra, -4(sp)
139
         addi sp, sp, -4
140
         jal ra, delay
         lw ra, 0(sp)
141
142
         addi sp, sp, 4
```

```
void codegen();
void codegen()

{
   int a = 1 + 2 * 1; // a = 3
   int b = (a + 3) / 2; // b = 3
   digitalWrite(26, 1);
   delay(a * 1000); // delay 3 seconds
   digitalWrite(26, 0);

delay(b * 1000); // delay 3 seconds

delay(b * 1000); // delay 3 seconds
}
```



```
145
        // BEGIN EPILOGUE
146
        // restore callee-saved registers
        // s0 at this point should be the same as in prologue
147
        lw s11, -48(s0)
148
        lw s10, -44(s0)
149
                                                   void codegen();
        lw s9, -40(s0)
150
                                                   void codegen()
        lw s8, -36(s0)
151
152
        lw s7, -32(s0)
        lw s6, -28(s0)
153
                                                     int a = 1 + 2 * 1; // a = 3
        lw s5, -24(s0)
154
                                                     int b = (a + 3) / 2; // b = 3
        lw s4, -20(s0)
155
                                                     digitalWrite(26, 1);
        lw s3, -16(s0)
156
                                                     delay(a * 1000); // delay 3 seconds
        lw s2, -12(s0)
157
                                                     digitalWrite(26, 0);
        lw s1, -8(s0)
158
        lw sp, -4(s0)
159
                                                     delay(b * 1000); // delay 3 seconds
160
        addi sp, sp, 4
                                             10
        lw = 0, -4(sp)
161
        // END EPILOGUE
162
163
        jalr zero, 0(ra) // return
164
```



End

