龙芯汇编自动生成

徐 伟

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2025年11月24日

栈式分配介绍

栈式分配：What

将IR翻译为汇编指令：

```
%op2 = add i32 %op0, %op1
```

```
add.w $rd, $rj, $rk
```

如何得到%op0与%op1的值?

栈式分配：What

栈式分配的定义

1. 程序的所有变量都保存在栈上
2. 只在参与计算时提取到寄存器中

栈式分配的步骤

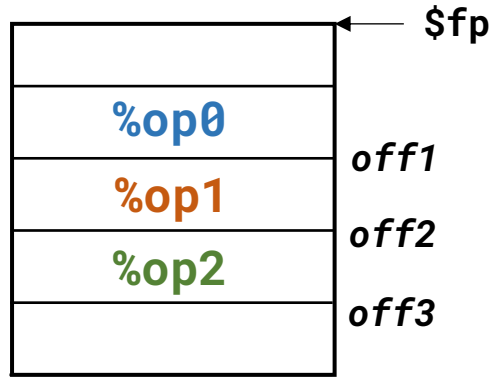
1. 变量分配：为每个变量分配栈帧位置
2. 指令选择
   a. load：提取源操作数至寄存器
   b. 指令选择：根据指令类型选择合适的汇编指令
   c. store：将结果从寄存器写回栈帧

栈式分配：What

将IR翻译为汇编指令：

```
%op2 = add i32 %op0, %op1
```

```
ld.w  $t0, $fp, off1
ld.w  $t1, $fp, off2
add.w $t0, $t0, $t1
st.w  $t0, $fp, off3
```



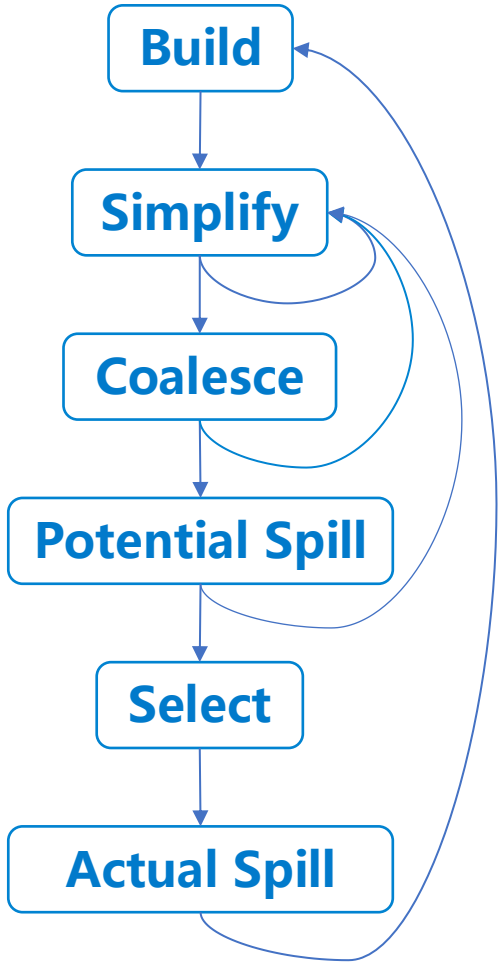
程序部分栈帧

栈式分配：Why

寄存器分配

- 目标：变量保存在寄存器中
- 难以用算法实现最优解
- 需要构建冲突图、循环迭代
- 需要处理复杂的寄存器溢出
- ……

实现复杂！



寄存器分配

汇编程序示例

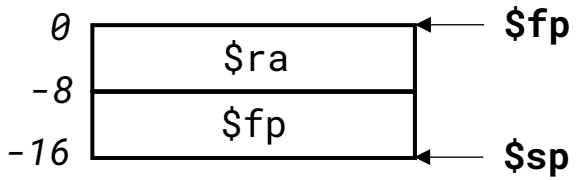
程序1：返回值

```
define i32 @main() {
label_entry:
  ret i32 0
}
```

IR程序

```
    .text
    .globl main
    .type main, @function
main:
    st.d   $ra, $sp, -8
    st.d   $fp, $sp, -16
    addi.d $fp, $sp, 0
    addi.d $sp, $sp, -16
.main_label_entry:
    addi.w $a0, $zero, 0
    b      main_exit
main_exit:
    addi.d $sp, $sp, 16
    ld.d   $ra, $sp, -8
    ld.d   $fp, $sp, -16
    jr $ra
```

汇编代码



程序栈帧

程序2：局部变量赋值

```
define void @main() {
label_entry:
  %op0 = alloca i32
  store i32 1234, i32* %op0
  ret void
}
```

IR程序

```
        .text
        .globl main
        .type main, @function
main:
        st.d    $ra, $sp, -8
        st.d    $fp, $sp, -16
        addi.d $fp, $sp, 0
        addi.d $sp, $sp, -32
.main_label_entry:
        addi.d $t0, $fp, -28
        st.d    $t0, $fp, -24
        ld.d    $t0, $fp, -24
        addi.w $t1, $zero, 1234
        st.w    $t1, $t0, 0
        addi.w $a0, $zero, 0
        b       main_exit
main_exit:
        addi.d $sp, $sp, 32
        ld.d    $ra, $sp, -8
        ld.d    $fp, $sp, -16
        jr      $ra
```

汇编代码

程序2：局部变量赋值

```
define void @main() {
label_entry:
  %op0 = alloca i32
  store i32 1234, i32* %op0
  ret void
}
```

IR程序

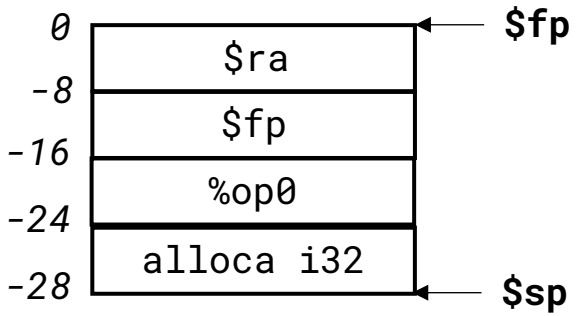
```
.....

addi.d $t0, $fp, -28
st.d   $t0, $fp, -24

ld.d   $t0, $fp, -24
addi.w $t1, $zero, 1234
st.w   $t1, $t0, 0

.....
```

汇编代码



程序栈帧

程序3：分支

```
define i32 @main() {
label_entry:
  %op0 = icmp slt i32 1, 2
  br i1 %op0, Label %L1, %L2
L1:
  ret i32 0
L2:
  ret i32 1
}
```

IR程序

程序3：分支

```
        .text
        .globl main
        .type main, @function
main:
        st.d    $ra, $sp, -8
        st.d    $fp, $sp, -16
        addi.d $fp, $sp, 0
        addi.d $sp, $sp, -32
.main_label_entry:
        addi.w $t0, $zero, 1
        addi.w $t1, $zero, 2
        slt     $t2, $t0, $t1
        st.b    $t2, $fp, -17
        ld.b    $t0, $fp, -17
        bstrpick.w $t0, $t0, 0, 0
        st.w    $t0, $fp, -21
        ld.w    $t0, $fp, -21
        addi.w $t1, $zero, 0
        xor     $t2, $t0, $t1
        sltu    $t2, $zero, $t2
        st.b    $t2, $fp, -22
        ld.b    $t0, $fp, -22
bstrpick.d    $t0, $t0, 0, 0
        bne     $t0, $zero, .main_label3
        b       .main_label5
.main_label3:
        addi.w $a0, $zero, 0
        b       main_exit
.main_label4:
        addi.w $a0, $zero, 0
        b       main_exit
.main_label5:
        addi.w $a0, $zero, 1
        b       main_exit
main_exit:
        addi.d $sp, $sp, 32
        ld.d    $ra, $sp, -8
        ld.d    $fp, $sp, -16
        jr      $ra
```

汇编代码

程序3：分支

```
define i32 @main() {
label_entry:
  %op0 = icmp slt i32 1, 2
  br i1 %op0, Label %L1, %L2
L1:
  ret i32 0
L2:
  ret i32 1
}
```

IR程序

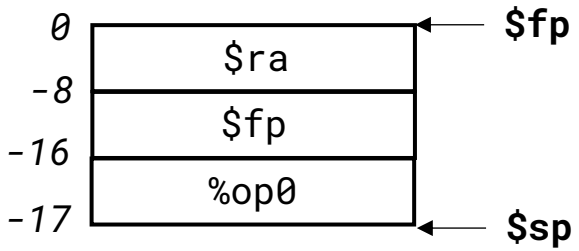
```
......

addi.w $t0, $zero, 1
addi.w $t1, $zero, 2
slt    $t2, $t0, $t1
st.b   $t2, $fp, -17

ld.b $t0, $fp, -17
bne  $t0, $zero, .L1
b    .L2

......
```

汇编代码



程序栈帧

程序4：全局变量

```
@a = global i32 zeroinitializer
define void @main() {
label_entry:
  store i32 10, i32* @a
  ret void
}
```

IR程序

```
        .text
        .section .bss, "aw", @nobits
        .globl   a
        .type    a, @object
        .size    a, 4
a:
        .space   4
        .text
        .globl   main
        .type    main, @function
main:
        st.d     $ra, $sp, -8
        st.d     $fp, $sp, -16
        addi.d   $fp, $sp, 0
        addi.d   $sp, $sp, -16
.main_label_entry:
        la.local $t0, a
        addi.w   $t1, $zero, 10
        st.w   $t1, $t0, 0
        addi.w $a0, $zero, 0
        b      main_exit
main_exit:
        addi.d $sp, $sp, 16
        ld.d   $ra, $sp, -8
        ld.d   $fp, $sp, -16
        jr     $ra
```

汇编代码

程序4：全局变量

```
@a = global i32 zeroinitializer
define void @main() {
label_entry:
  store i32 10, i32* @a
  ret void
}
```

IR程序

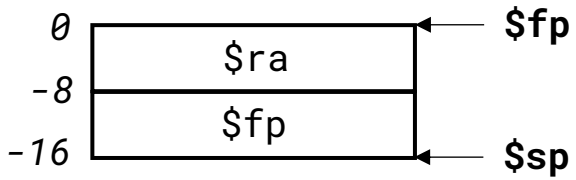
```
.text
.section .bss, "aw", @nobits
.globl a
.type a, @object
.size a, 4
a:
.space 4

......

.main_label_entry:
la.local $t0, a
addi.w   $t1, $zero, 10
st.w     $t1, $t0, 0

......
```

汇编代码



程序栈帧

程序5：函数调用

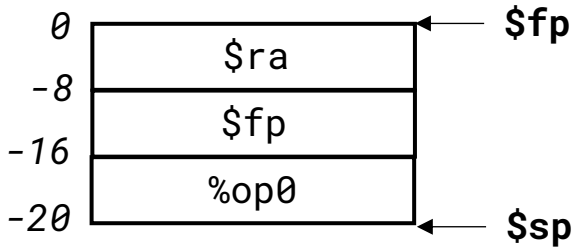
```
define i32 @func(i32 %arg0) {
label_entry:
  ret i32 %arg0
}
define void @main() {
label_entry:
  %op0 = call i32 @func(i32 1)
  ret void
}
```

IR程序

```
        .text
        .globl func
        .type func, @function
func:
        jr $ra

        .globl main
        .type main, @function
main:
        st.d    $ra, $sp, -8
        st.d    $fp, $sp, -16
        addi.d $fp, $sp, 0
        addi.d $sp, $sp, -32
.main_label_entry:
        addi.w $a0, $zero, 1
        bl      func
        st.w    $a0, $fp, -20
        addi.w $a0, $zero, 0
        b       main_exit
main_exit:
        addi.d $sp, $sp, 32
        ld.d    $ra, $sp, -8
        ld.d    $fp, $sp, -16
        jr      $ra
```

汇编代码



程序栈帧

一起努力
打造国产基础软硬件体系！

徐 伟

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2025年11月24日