Week 6.1

Thursday, May 2, 2024 10:13 PM

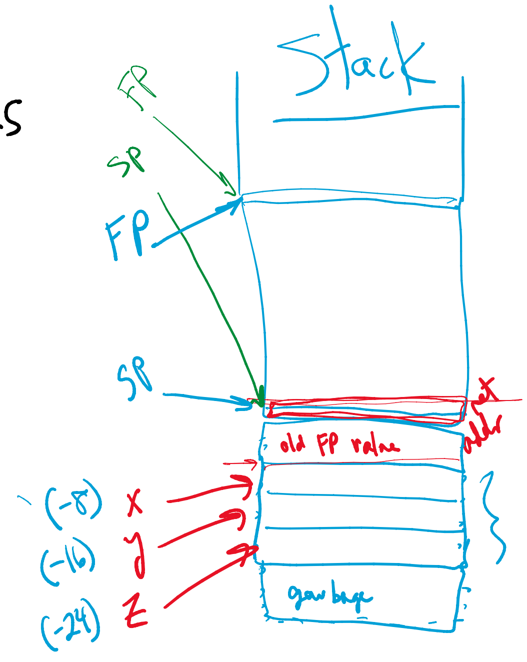
Stage 1: no functions or calls (other than main)
no structs or pointers
no globals

$arith, $cmp, $copy, $branch, $jump, $ret

Summary:

1. create a template (everything but main)
2. prologue
3. output ISA for basic blocks
4. epilogue

2 important registers
FP (frame pointer) %rbp
SP (stack pointer) %rsp



prologue

- emit 'main' label
- push FP onto stack
- set FP = SP
- allocate space on stack for main's locals (incl. double-word alignment)
- zero-initialize all locals
- jump to main_entry

store mapping from local → offset

epilogue

epilogue

- emit epilogue label
- SP = FP
- restore old FP
  by popping off stack
- pop return address &
  jump to it

translating LIR instructions

either variable
or constant

"memory location of x"

- X = $copy op
  store value of op in [X]

- X = $arith <aop> op$_1$ op$_2$
  apply <aop> to values of op$_1$, op$_2$
  & store result in [X]

  } division is weird

- X = $cmp <rop> op$_1$ op$_2$
  compare op$_1$ & op$_2$ (sets condition code)
  store ∅ or 1 to [X] depending on
  <rop> & condition code

- $jump lbl
  jump to main_lbl

- $branch op lbl$_1$ lbl$_2$
  compare op to ∅ (sets code)
  if ne jump to lbl$_1$
  [illegible]

if ne jump to lbl_1
else jump to lbl_2

- $ret op

  return value goes into
  a specific reg. (%rax)

  jump to main_epilogue