Stage 4 Codegen Example 1

Monday, May 6, 2024 6:38 PM

```
fn main() -> int {
  let x:int
  entry:
    x = $call_dir foo(40, 2) then exit

  exit:
    $ret x
}

fn foo(p: int, q: int) -> int {
  let x:int
  entry:
    x = $arith add p q
    $ret x
}
```

```
.data
.text

.globl foo                  // start foo function
foo:
  pushq %rbp                // prologue
  movq %rsp, %rbp
  subq $16, %rsp            // note alignment
  movq $0, -8(%rbp)
  jmp foo_entry

foo_entry:
  movq 16(%rbp), %r8        // positive offset for parameter
  addq 24(%rbp), %r8        // here too
  movq %r8, -8(%rbp)
  movq -8(%rbp), %rax
  jmp foo_epilogue

foo_epilogue:
  movq %rbp, %rsp
  popq %rbp
  ret

.globl main                 // start main function
main:
  pushq %rbp                // prologue
  movq %rsp, %rbp
  subq $16, %rsp            // note alignment
  movq $0, -8(%rbp)
  jmp main_entry

main_entry:
  pushq $2                  // push second arg
  pushq $40                 // push first arg
  call foo                  // push return address and jump
  movq %rax, -8(%rbp)       // save return value
  addq $16, %rsp            // restore stack
  jmp main_exit

main_exit:
  movq -8(%rbp), %rax
  jmp main_epilogue

main_epilogue:
  movq %rbp, %rsp
  popq %rbp
  ret
```