Stage 6 Codegen Example 1

Tuesday, May 7, 2024 2:59 PM

```
struct foo {
  f1: int
  f2: int
}

fn main() -> int {
  let x:&foo, y:&int, z:&int

  entry:
    x = $alloc 1
    y = $gfp x f1
    z = $gfp x f2
    $ret 0
}
```

```
.data

out_of_bounds_msg: .string "out-of-bounds array access"
invalid_alloc_msg: .string "invalid allocation amount"

.text

.globl main
main:
  pushq %rbp
  movq %rsp, %rbp
  subq $32, %rsp
  movq $0, -8(%rbp)
  movq $0, -16(%rbp)
  movq $0, -24(%rbp)
  jmp main_entry

main_entry:
  movq $1, %r8                  // allocate single struct
  cmpq $0, %r8
  jle .invalid_alloc_length
  movq $2, %rdi                 // compute size: 2 fields + header
  imulq %r8, %rdi
  incq %rdi
  call _cflat_alloc
  movq $1, %r8
  movq %r8, 0(%rax)             // store header
  addq $8, %rax
  movq %rax, -8(%rbp)           // store pointer to x
  movq -8(%rbp), %r8            // compute pointer to x.f1
  leaq 0(%r8), %r9
  movq %r9, -16(%rbp)           // store in y
  movq -8(%rbp), %r8            // compute pointer to x.f2
  leaq 8(%r8), %r9
  movq %r9, -24(%rbp)           // store in z
  movq $0, %rax
  jmp main_epilogue

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

.out_of_bounds:
  lea out_of_bounds_msg(%rip), %rdi
  call _cflat_panic

.invalid_alloc_length:
  lea invalid_alloc_msg(%rip), %rdi
  call _cflat_panic
```