Stage 6 Codegen Example 2

Tuesday, May 7, 2024 3:00 PM

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

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

  entry:
    x = $alloc 10
    y = $gep x 5
    z = $gfp y 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 $10, %r8              // allocate struct array
  cmpq $0, %r8
  jle .invalid_alloc_length
  movq $2, %rdi              // compute size: (10 * 2) + 1
  imulq %r8, %rdi
  incq %rdi
  call _cflat_alloc
  movq $10, %r8
  movq %r8, 0(%rax)          // store header info
  addq $8, %rax              // store pointer in x
  movq %rax, -8(%rbp)
  movq $5, %r8               // index x[5]
  cmpq $0, %r8
  jl .out_of_bounds
  movq -8(%rbp), %r9
  movq -8(%r9), %r10
  cmpq %r10, %r8
  jge .out_of_bounds
  imulq $16, %r8             // compute array offset (2 words * index)
  addq %r9, %r8              // add base pointer
  movq %r8, -16(%rbp)        // store in y
  movq -16(%rbp), %r8        // compute pointer to y.f1
  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
```