编译原理 stage 5 报告

傅子轩 2020010742 计01

主要改动

1. `parser`中增加了对数组定义和访问的解析。增加了对数组传参的解析
2. 增加了`ast`结点`ArrayType`和`ArrayRef`，`ArrayRef`接受一个`Lvalue`和一个`Expr`，分别作为基址和偏移
3. `build_sym`中对数组声明生成对应的`ArrayType`，对`ArrayRef`分别`accept`其`arr_base`和`index`
4. 对于全局变量和数组，其作为左值的`val`为指向对应位置的指针。对于全局变量，和数组的最后一维下标运算`VarRef::ATTR(lv_kind)`设置为`MEM_VAR`。表示在对其`assign`和`LvalueExpr`时需要通过访存进行。例如：

```
int a;
int b[2][2];

int main(){
    a;    // lvalue, ATTR(val)=&a
    b[1]; // lvalue, ATTR(val)=&b[1][0]
}
```

5. 对于保存在寄存器上的局部变量和多维数组的非最后一层解引用（`int a[1][1]; a[1]`）设置为`SIMPLE_VAR`。在`LvalueExpr`的求值中直接访问对应的`Temp`。对于`ArrayRef`，其`arr_base`的类型必须为`ArrayType`，`ArrayRef`的类型为`arr_base`的元素类型.

6. `translation`中对于全局变量，使用`LOAD_SYMBOL`作为其地址，对于局部数组声明，使用`tac::ALLOC`在栈上分配空间，并使用`sw`和`fill_n`进行初始化（若存在初始化列表）

7. `riscv_md`中，先遍历所有指令，找到`ALLOC`，并在`RiscvStackManager`中预留空间，并记录偏移，随后再进行翻译。ALLOC指令的翻译就是简单的`addi dest, sp, offset`，对于全局数组初始化，使用`.word`和`.zero`或直接使用`.bss`段

思考题

step 11

先使用一块空间记录原`sp`，申请可变数组空间时，计算所需空间大小，减小`sp`以分配这块空间，返回时直接恢复原`sp`即可。(栈上的临时变量是用`fp`寻址的，所以`sp`的改变不会影响其他代码)