编译原理 stage 1 报告

傅子轩 2020010742 计01

主要改动

1. 在`scanner`中识别所有运算符和关键字
2. 在`parser`中添加了新运算符的产生式
3. 在`type_check`中为新增的`Expr`添加了类型信息
4. 在`translation`中为新增的`Expr`添加了到`tac`的翻译
5. 在`riscv_md`中新增了一些指令和伪指令，以将`tac`翻译成`riscv`代码
6. 为`if else`和`?:`规定了优先级和结合性以消除二义性。
7. 修复了一些bug，包括注释输出实际上是一个`use-after-free`，会使用被释放的内存，`scanner`里`<B> <EOF>`实际上应该为`<B><<EOF>>`，`scanner`中对换行符的处理在`\r\n`下位置信息会出错等等。

思考题

step 2

```
-(~(0x7fffffff))
```

使用取反获得`INT_MIN`，它是唯一一个取负会溢出的`int`值

step 3

```c
#include <stdio.h>

int main() {
    int a = 0x80000000;
    int b = -1;
    printf("%d\n", a / b);
    return 0;
}
```

`rv32`输出`-2147483648`

`x86`输出`Floating point exception`

step 4

使得指针解引用更加方便，例如：

```
if (ptr && ptr->member == 0) {...}
```

另外可以实现默认值：

```
return str || "";
```

如果没有短路求值代码将变得冗长。