Learning LLVM Part-1

作者：Sky3
原文链接：https://ld246.com/article/1524808552388
来源网站：链滴

（**这个还没写完TAT**）

LLVM不仅仅是一个编译器，而是一个有很多特性的编译框架，比如JIT、支持了很多非类C语言，如Rust等等，此外，还是一种App Store上的发布方式。

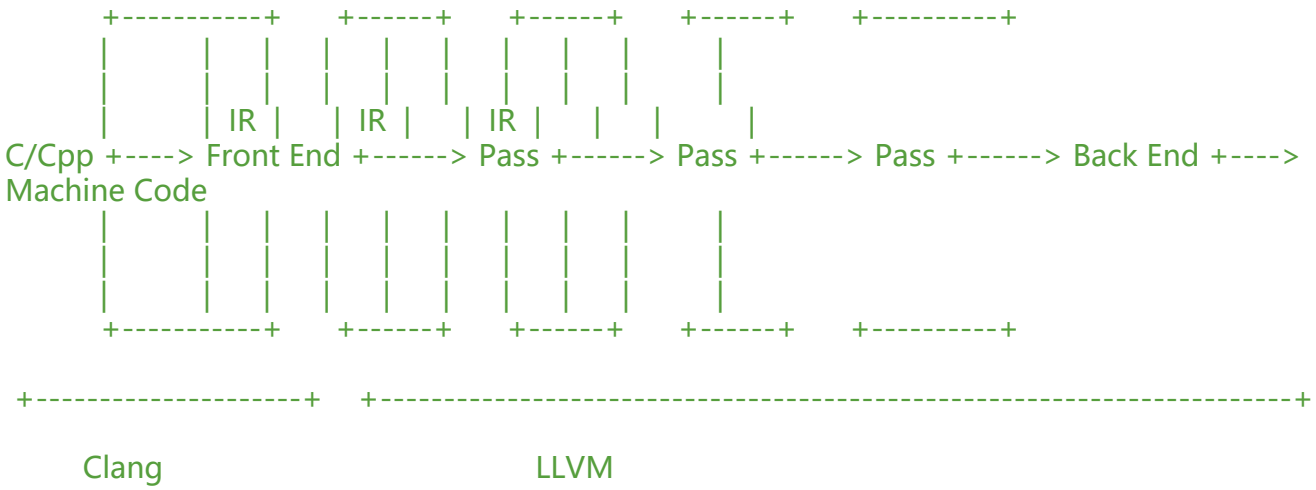
LLVM和其他编译器不同的地方是：

- LLVM的中间表示，即LLVM IR是一个创新，有了比汇编语言更好的可读性
- LLVM比其他的编译器更加模块化
- LLVM不仅仅可以实现编译优化，还可以做到：架构模拟、动态二进制分析工具、源代码变换（抽语法树等）、干预系统调用，还可以利用LLVM Pass给程序注入错误信息，来模拟一些硬件

LLVM的组成

LLVM架构和所有现代编译器的架构相同：

前端(Front end)、流程(Pass)、后端(Back end)



- 前端（Front End）获取你的源代码然后将它转变为某种中间表示。这种翻译简化了编译器其他部的工作，这样它们就不需要面对比如C++源码的所有复杂性。
- 流程（Pass）将程序在中间表示之间互相变换。一般情况下，流程也用来优化代码：流程输出的（间表示,IR）程序和它输入的（中间表示,IR）程序相比在功能上完全相同，只是在性能上得到改进,这分通常是给你发挥的地方，你的研究工具可以通过观察和修改编译过程流中的IR来完成任务。
- 后端（Back end）可以生成实际运行的机器码，一般不会动这部分代码，除非要编译针对某特殊架的机器码。

虽然当今大多数编译器都使用了这种架构，但是LLVM有一点值得注意而与众不同：整个过程中，程都使用了同一种中间表示。在其他编译器中，可能每一个流程产出的代码都有一种独特的格式。LLV在这一点上对hackers大为有利。我们不需要担心我们的改动该插在哪个位置，只要放在前后端之间个地方就足够了。

文件目录

然后我们分析一下LLVM的目录。

llvm/examples

这个目录是LLVM IR和JIT的使用用例。

llvm/include

LLVM库函数中导出的头文件，有三个子目录：

- llvm/include/llvm
- llvm/include/llvm/Support
- llvm/include/llvm/Config

llvm/lib

LLVM源码的主体，如子目录下的IR、AsmParser等Pass的源码也在这里。

llvm/projects

llvm/test

llvm/tools

llvm/utils

使用LLVM

使用已经编译好的可执行文件

http://releases.llvm.org

选择并下载、解压合适的版本即可。

编译安装

以LLVM 3.5.2为例，机器版本为Ubuntu Server 16.04 x64。

为了方便，自己写了一个脚本：

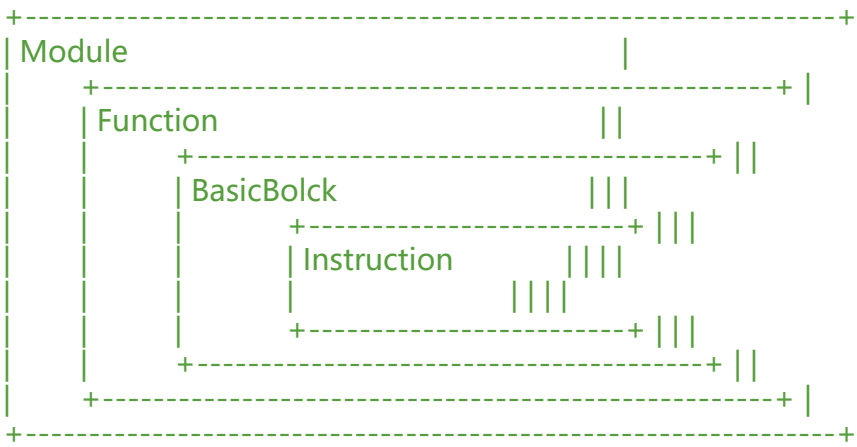
```
# download llvm src
cd ~ \
mkdir llvm3.5 && cd llvm3.5 \
wget http://llvm.org/releases/3.5.2/llvm-3.5.2.src.tar.xz \
tar -xf llvm-3.5.2.src.tar.xz \
mv llvm-3.5.2.src llvm \
rm -rf *.xz

# download clang src
cd llvm/tools \
wget http://llvm.org/releases/3.5.2/cfe-3.5.2.src.tar.xz \
```

```
tar -xf cfe-3.5.2.src.tar.xz \
mv cfe-3.5.2.src clang \
rm -rf *.xz

# download clang-tools-extra src
cd clang/tools \
wget http://llvm.org/releases/3.5.2/clang-tools-extra-3.5.2.src.tar.xz \
tar -xf clang-tools-extra-3.5.2.src.tar.xz \
mv clang-tools-extra-3.5.2.src.tar.xz  extra\
rm -rf *.xz

# download compiler-rt src
cd ~/llvm3.5/llvm/projects \
wget http://llvm.org/releases/3.5.2/compiler-rt-3.5.2.src.tar.xz \
tar -xf compiler-rt-3.5.2.src.tar.xz \
mv compiler-rt-3.5.2.src compiler-rt \
rm -rf *.xz

# begin compile llvm
cd ~/llvm3.5 \
mkdir build \
cmake ../llvm/ \
sudo make -j4 \
make install
```

编译需要很长很长很长很长很长时间...所以需要慢慢慢慢慢慢慢等。

编译好的二进制文件在~/llvm3.5/build/bin，查看版本：./~/llvm3.5/build/bin/clang -v

理解LLVM IR



Modules contain Functions, which contain BasicBlocks, which contain Instructions. Everything but Module descends from Value.

模块（Module），函数（Function），代码块（BasicBlock），指令（Instruction）

模块包含了函数，函数又包含了代码块，后者又是由指令组成。除了模块以外，所有结构都是从值产而来的。

编写第一个Pass