

The LLVM compiler framework

Writing a pass: Quick Start

Daniele Cattaneo

Politecnico di Milano

2020-04-17

These slides were originally written by Stefano Cherubin for the “Code Transformation and Optimization” course.

Contents

Introduction

LLVM framework quick start

Understanding LLVM

LLVM is **not** a compiler.

Understanding LLVM

LLVM is **not** a compiler.

LLVM is a
collection of components
which is **useful**
to build a compiler.

Getting LLVM

- ▶ “old” git mirrors
 - ▶ only llvm repo (subprojects in separated repos, can be added later)
 - ▶ `git clone -b release_90 --single-branch git@github.com:llvm-mirror/llvm.git`
- ▶ “new” git monorepo
 - ▶ all in one repo (llvm + major subprojects)
 - ▶ `git clone -b release/9.x --single-branch git@github.com:llvm/llvm-project.git`

What LLVM is made of

- ▶ C++ libraries
 - ▶ `src/include/llvm/...`
 - ▶ `src/lib/...`
- ▶ small application (tools)
 - ▶ `src/tools/...`
 - ▶ `src/utils/...`

You can find binaries of them in the installation directory under `root/bin/...`

clang

- ▶ clang is a compiler based on LLVM
- ▶ It compiles all major C-like languages
- ▶ It is part of the git monorepo
- ▶ It can be added as a tool in the LLVM framework but must be manually cloned in the tool directory
 1. `cd src/tools`
 2. `git clone http://llvm.org/git/clang` (git mirror version)
- ▶ You can easily see on a production quality compiler the impact of changes you made on your local copy of LLVM

Contents

Introduction

LLVM framework quick start

Commands

llvm-as LLVM assembler

llvm-dis LLVM disassembler

opt LLVM optimizer

llc LLVM static compiler

lli directly execute programs from LLVM bitcode

llvm-link LLVM bitcode linker

llvm-mca LLVM machine code analyzer

llvm-nm list LLVM bitcode and object file's symbol table

llvm-stress generate random .ll files

llvm-config prints out install configuration parameters

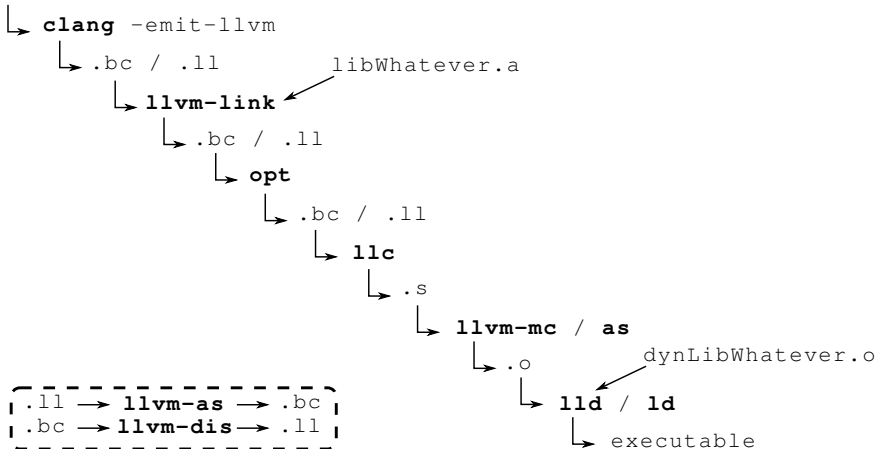
llvm-dwarfdump print contents of DWARF sections

For a complete reference, see the LLVM command guide*

*<http://llvm.org/docs/CommandGuide/index.html>

Simulating a LLVM driver manually

.c source



Writing a LLVM pass

There are a lot of tutorials available:

- ▶ Official developer guide
`llvm.org/docs/WritingAnLLVMPass`
- ▶ Out-of-source pass
`github.com/quarkslab/
llvm-dev-meeting-tutorial-2015`

We will follow the first one, with a few adjustments.

Testing

LLVM has an internal testing infrastructure*. Please use it.

llvm-lit LLVM Integrated Tester

1. Forge a proper LLVM-IR input file (.ll) for your test case
2. Instrument it with `lit` script comments
3. Run `lit` on your test
 - ▶ `llvm-lit /llvm/test/myTests/singleTest.ll`
run a single test
 - ▶ `llvm-lit /llvm/test/myTests`
run the test suite (folder)
4. Run `lit` on the LLVM test suite (regression testing)

To submit a bug report to LLVM developers you will be asked to write a `lit` test case that highlights the bug.

*<http://llvm.org/docs/TestingGuide.html>