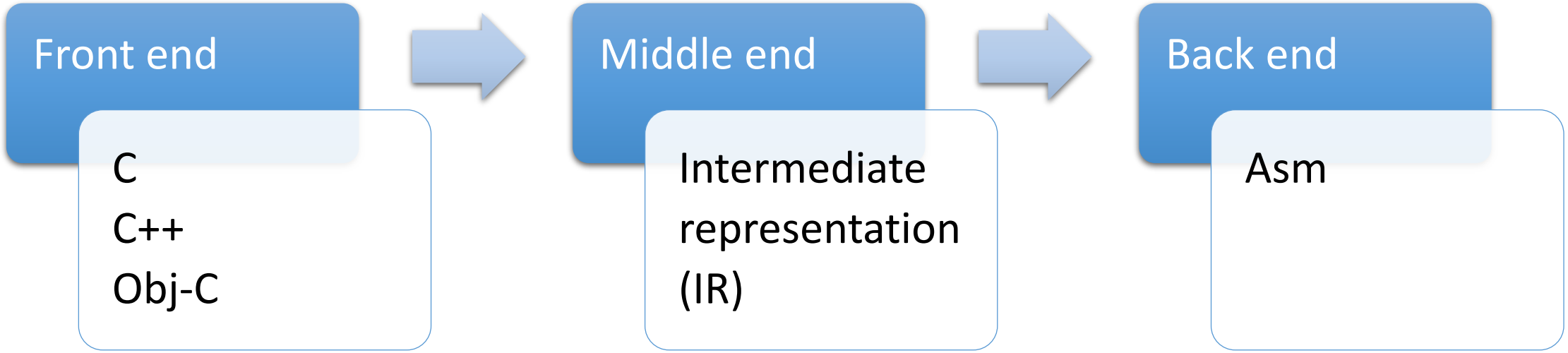
LLVM

МФТИ
Весна 2024

LLVM

- Дипломный проект Криса Латнера
- Впервые появился в 2003 году (LLVM, а не Крис!)
- Изначально планировалось как виртуальная машина (Low Level Virtual Machine)
- Сейчас разрабатывает RISC-V backend в SiFive

Архитектура компиляторов



Обсуждение: пишем компилятор

Как бы вы реализовывали подобный проект?

Из каких библиотек он бы состоял?

Обсуждение: пишем компилятор

- Front end
  - Lexer
  - Parser
  - AST
  - Semantic analyzer
  - AST to IR
- Middle end
  - IR
  - Optimizations over IR
- Back end
  - Codegen

Подобным образом реализованы и **GCC**, и **Clang/LLVM**

Но *разработчики* компиляторов больше любят **Clang** и **LLVM**

Почему?

Почему LLVM – one love
Первая причина – это IR

Gimple (GCC IR)

```
int square (int num)
{
  int D.2744;
  int _2;

  <bb 2> :
  _2 = num_1(D) * num_1(D);

  <bb 3> :
<L0>:
  return _2;
}
```

LLVM IR

```
define i32 @square(i32 %num) {
entry:
  %num.addr = alloca i32
  store i32 %num, ptr %num.addr
  %0 = load i32, ptr %num.addr
  %1 = load i32, ptr %num.addr
  %mul = mul nsw i32 %0, %1
  ret i32 %mul
}
```

LLVM IR – самоописывающийся, строго типизированный, единый* на весь middle end

Почему LLVM – one love
Вторая причина – это лицензия

GCC распространяется под лицензией GNU GPLv3

GNU GPL требует распространения с бинарными файлами (в том числе неизменными) исходного кода или письменного обязательства его предоставить

LLVM распространяется под лицензией Apache 2.0

Apache 2.0 позволяет комбинировать открытый и закрытый код без обязательства его предоставления

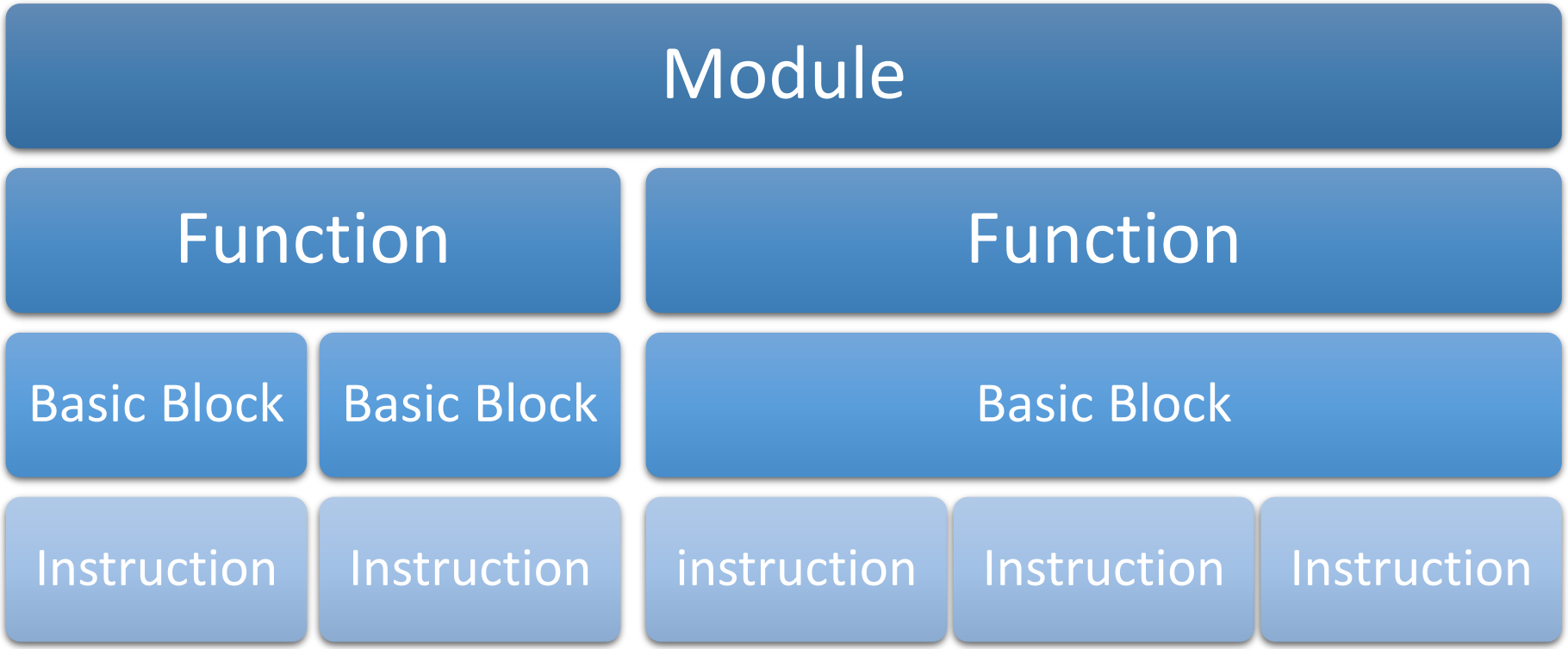
Почему LLVM – one love
Третья причина – это модульность

- Языковой front end clang переиспользуется в отладчике lldb
- Описание target’a из backend’a переиспользуется в линкере lld, отладчике lldb и других проектах
- Множество clang-based тулов – clang-format, clang-tidy, clangd и т.д.

Самоописываемость IR позволяет создавать отдельные средства для работы над IR отдельно от компилятора

Почему LLVM IR так удобен

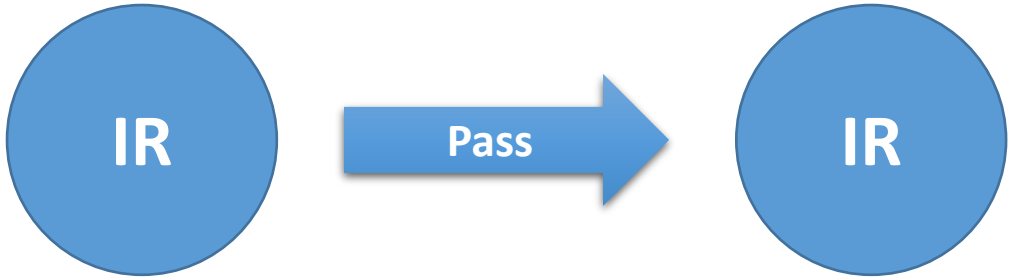
1. Четкая иерархия объектов



Почему LLVM IR так удобен

2. Четкий механизм работы с IR – Passes

- Принимает на вход IR
- *Возможно* модифицирует IR
- Имеет состояние
- Может зависеть от других пассов (анализов)



Запуск всех пассов контролируется с помощью Pass Manager

Низкоуровневый IR в LLVM

Низкоуровневый IR в LLVM называется **Machine IR** (**MIR**)

➢Очень похож на ассемблер

Machine Instruction состоит из

- Опкода (target specific)
- Набора операндов
  - Регистры (как виртуальные так и физические)
  - Флаги
  - Immediate
  - Метки

Низкоуровневый IR в LLVM

Для работы с MIR есть специальная иерархия пассов:

- Module Pass
- Machine Function Pass
- Machine Basic Block Pass

Почему LLVM удобен для snippy

Задача snippy – генерировать ассемблерные сниппеты с заданным распределением по опкодам

- Как это ложится на LLVM?

```
sections:
    - no:        1
      VMA:       0x80000000
      SIZE:      0x400000
      LMA:       0x80000000
      ACCESS:    rx
    - no:        2
      VMA:       0x80600000
      SIZE:      0x400000
      LMA:       0x80600000
      ACCESS:    rw

histogram:
    - [ADD, 1.0]
    - [ADDI, 1.0]
```

Почему LLVM удобен для snippy

Задача snippy – генерировать ассемблерные сниппеты с заданным распределением по опкодам

- Использует LLVM MIR (физические регистры)
- Простейший pass manager:
  - Создание функции в модуле
  - Генерация инструкций в функции
  - Конвертация MIR в obj
- Для поддержки других архитектур необходимо реализовать только target-specific функциональность **snippy**
- Добавление новой функциональности через расширение старых пассов или добавление новых

```
sections:
    - no:        1
      VMA:       0x80000000
      SIZE:      0x400000
      LMA:       0x80000000
      ACCESS:    rx
    - no:        2
      VMA:       0x80600000
      SIZE:      0x400000
      LMA:       0x80600000
      ACCESS:    rw

histogram:
    - [ADD, 1.0]
    - [ADDI, 1.0]
```

To be continued …

На следующем занятии

- Узнаем подробнее как пользоваться snippy
- Рассмотрим сценарии применения snippy
- Узнаем как можно применить модель в snippy