# LLVM Pass Manager

Devang Patel

# Agenda

- Introduction
- Pass Manager Design
- How to use Pass Manager
- How to extend Pass Manager
- Q&A

# Introduction

- What does Pass Manager do ?

# Introduction

- What does Pass Manager do ?
- Keeps analysis information up to date

# Introduction

- What does Pass Manager do ?
- Keeps analysis information up to date
- Manages memory use

# Introduction

- What does Pass Manager do ?
- Keeps analysis information up to date
- Manages memory use
- Enforces discipline

# Introduction

- What does Pass Manager do ?
- Keeps analysis information up to date
- Manages memory use
- Enforces discipline
- Makes pass developers' life simple

# Introduction

- What does Pass Manager do ?
- Keeps analysis information up to date
- Manages memory use
- Enforces discipline
- Makes pass developer's life simple
- Does **not** find optimal optimization sequence

# Design

Pass Manager

Function Pass Manager

# Design

Pass Manager

Function Pass Manager

Module PM

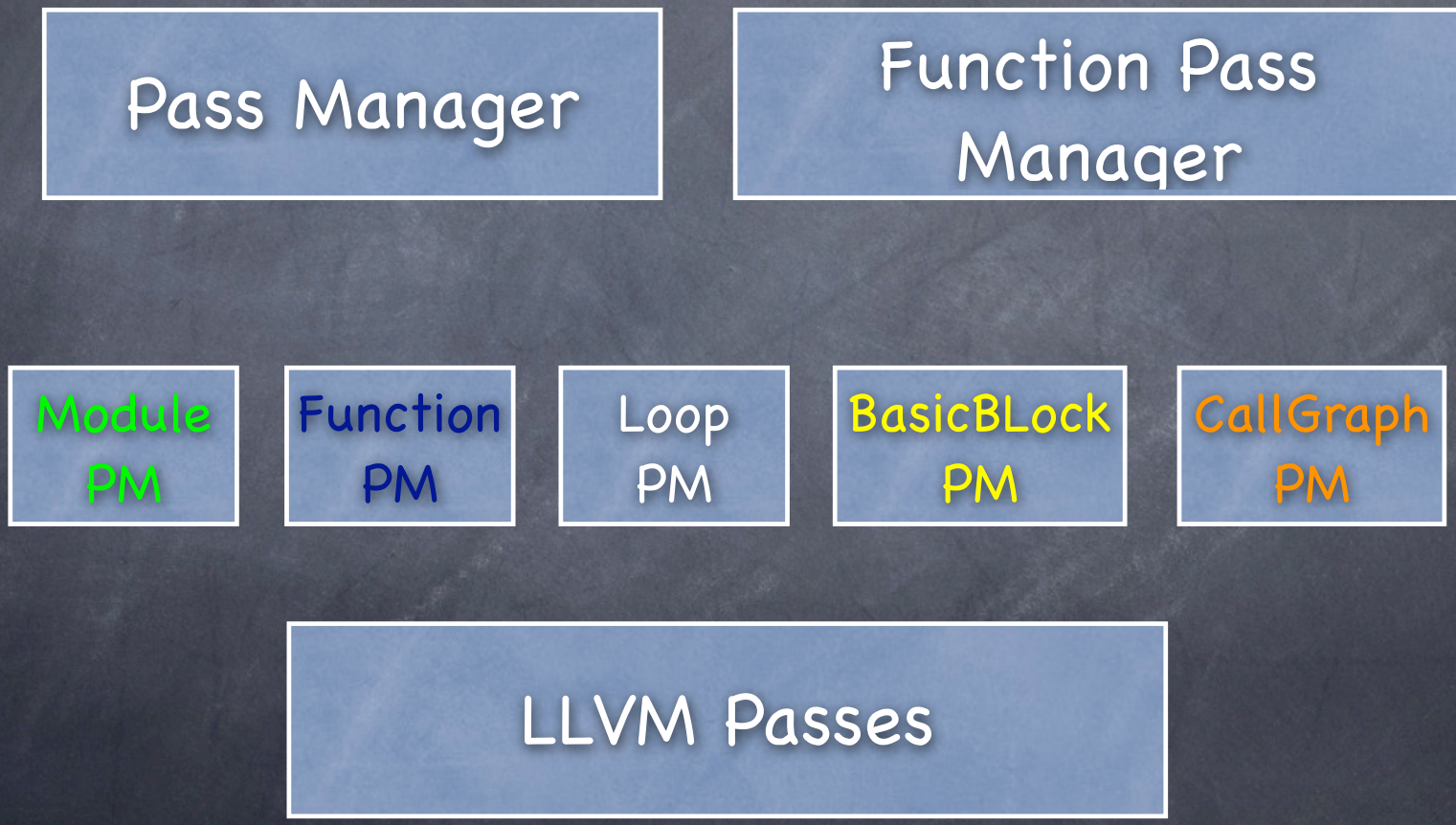
Function PM

Loop PM

BasicBLock PM

CallGraph PM

# Design


Pass Manager
Function Pass Manager
Module PM
Function PM
Loop PM
BasicBLock PM
CallGraph PM
LLVM Passes

# Design
## Information Maintained by Pass Manager

- List of passes
- List of Available analysis information at each stage of pass execution pipeline
- Last user of available analysis at each stage of pass execution pipeline

# Design

$opt in.bc -o out.bc -dce

For each function

Invoke Dead Code Elimination

# Design

$opt in.bc -o out.bc -dce -gcse

For each function

Invoke Dead Code Elimination

Invoke GCSE

# Design

$opt in.bc -o out.bc -dce -gcse

For each function

Invoke DCE

Construct Dominator Tree

Construct ET Forest

Invoke GCSE

Destroy Dominator Tree/ET Forest

# Design

$opt in.bc -o out.bc -dce  -constprop -inline

For each function

Invoke DCE

Invoke Constant Propagation

Construct call Graph

For each SCC

Invoke Function Inliner

# Design

$opt in.bc -o out.bc -inline -dce -constprop

Construct call Graph

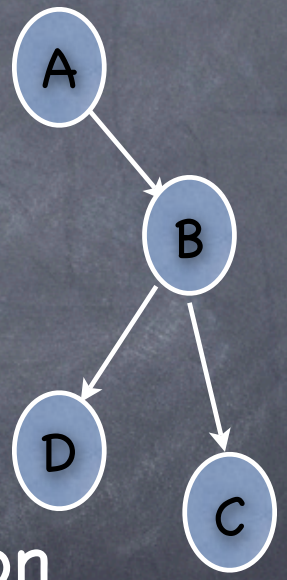
  For each SCC

    Invoke Function Inliner

    For each function

      Invoke DCE

      Invoke Constant Propagation



# How to use Pass Manager

- Select appropriate base class (e.g. LoopPass)
- Implement getAnalysisUsage()
- Implement appropriate run method (e.g. runOnLoop()

# How to use Pass Manager

```
class LoopRotate : public LoopPass {
  ...
  // LCSSA form makes instruction renaming easier.
  virtual void getAnalysisUsage(AnalysisUsage &AU) cons {
        AU.addRequiredID(LCSSAID);
        AU.addPreservedID(LCSSAID);
  }
  ...
  // Rotate Loop L as many times as possible. Return true if
  // loop is rotated at least once.
  bool runOnLoop(Loop *L, LPPassManager &LPM);
  ...
  };
```

# How to use Pass Manager

- opt is command line interface to use Pass Manager
- Use `-debug-pass=Structure` to see pass sequence executed by Pass Manager

# Extending Pass Manager

Introduce Loop Pass Manager !

# Extending Pass Manager

## (1) Define Loop Pass Manager

```
class LPPassManager : public PMDataManager,
                      public FunctionPass {
  ...
  bool runOnFunction(Function &F) {
   for each Loop L
     for each Loop Transformation Pass LP
      LP->runOnLoop(L, *this);
  }
  ...
};
```

# Extending Pass Manager

## (2) Define Loop Pass

```
class LoopPass : public Pass {
...
virtual bool runOnLoop(Loop &L);
void assignPassManager(PMStack &PM,
                       PassManagerType PT);
...
}
```

# Summary

- Pass Manager plays important role in making LLVM optimization framework flexible
- Pass Manager makes developers life easy
- More info Writing an LLVM Pass @ http://llvm.org/docs/WritingAnLLVMPass.html