9th Central and Eastern European
Software Engineering Conference
in Russia - CEE-SECR 2013

October 23 - 25, Moscow

2013 CEE-SEC(R)
Software Engineering
Conference in Russia

# LLVM and Clang
# Advancing Compilers and Tools

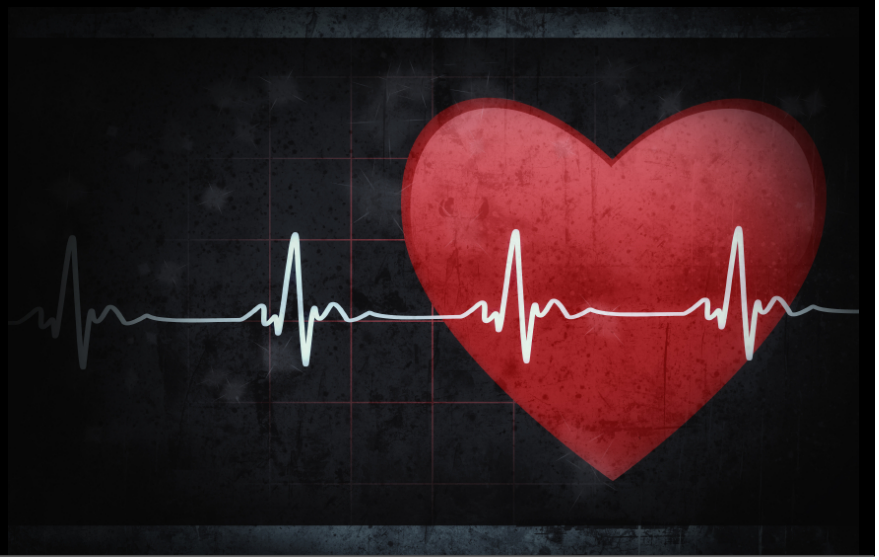

Chris Lattner

http://llvm.org


October 25, 2013

# LLVM is everywhere

- Industry
- Open Source
- Academia



# … for many different things

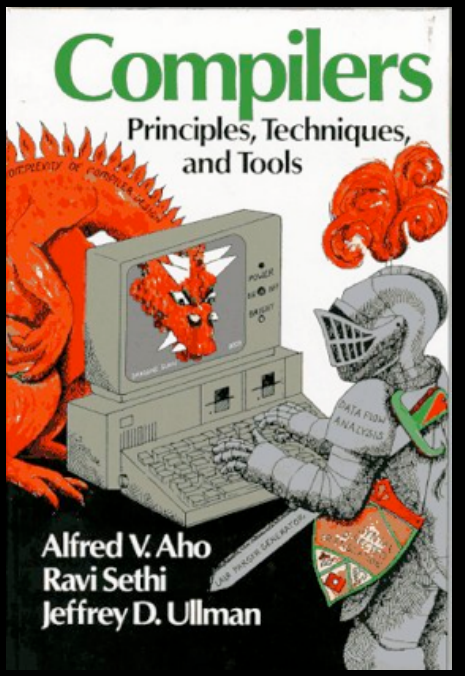
- System compiler for Apple and FreeBSD platforms
- Used by most GPGPU implementations
- Many new language implementations
- Finding bugs in source code
- Special effects in movies
- Games, Playstation 4

So… , what is it?

# What is a compiler?

**com·pil·er**
*noun*

1. a person who compiles information (as for reference purposes): *a compiler of anthologies.*
2. a computer program that transforms human readable source code of another computer program into the machine readable code that a CPU can execute.




- Clang and GCC are compilers
- What is LLVM?

# What is LLVM?

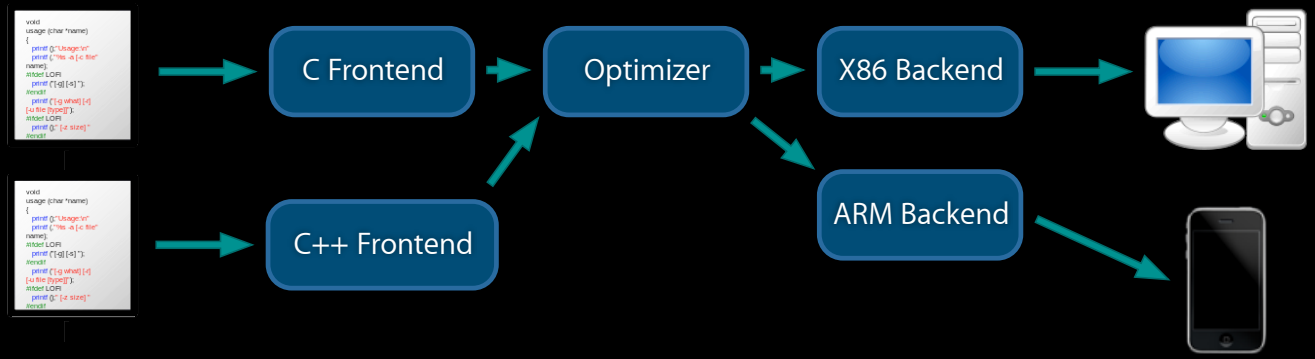
## llvm.org is an open source umbrella project

- Provides useful tools:
  - Assembler, linker, compiler, debugger, and more
- Strong community, with shared values:
  - Common processes, patch review, etc
  - Common design approaches
  - Preference for MIT/BSD License
- LLVM is a compiler **infrastructure**!


Compiler Infrastructure 101

# How does a compiler work?


- Frontend: Parse and validate source code
- Optimizer: Improve intermediate form
- Backend: Generate target specific code

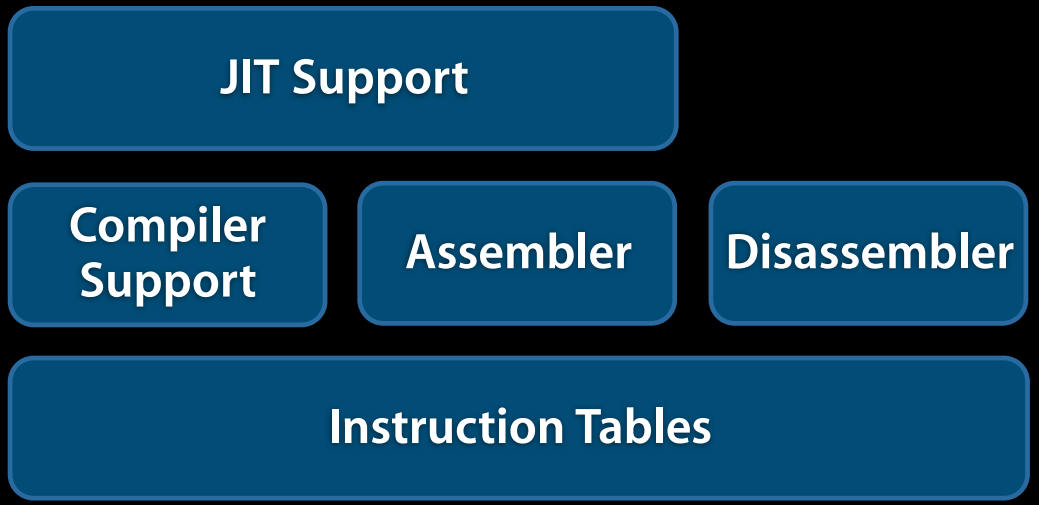



**Standard approach for at least 35 years!**

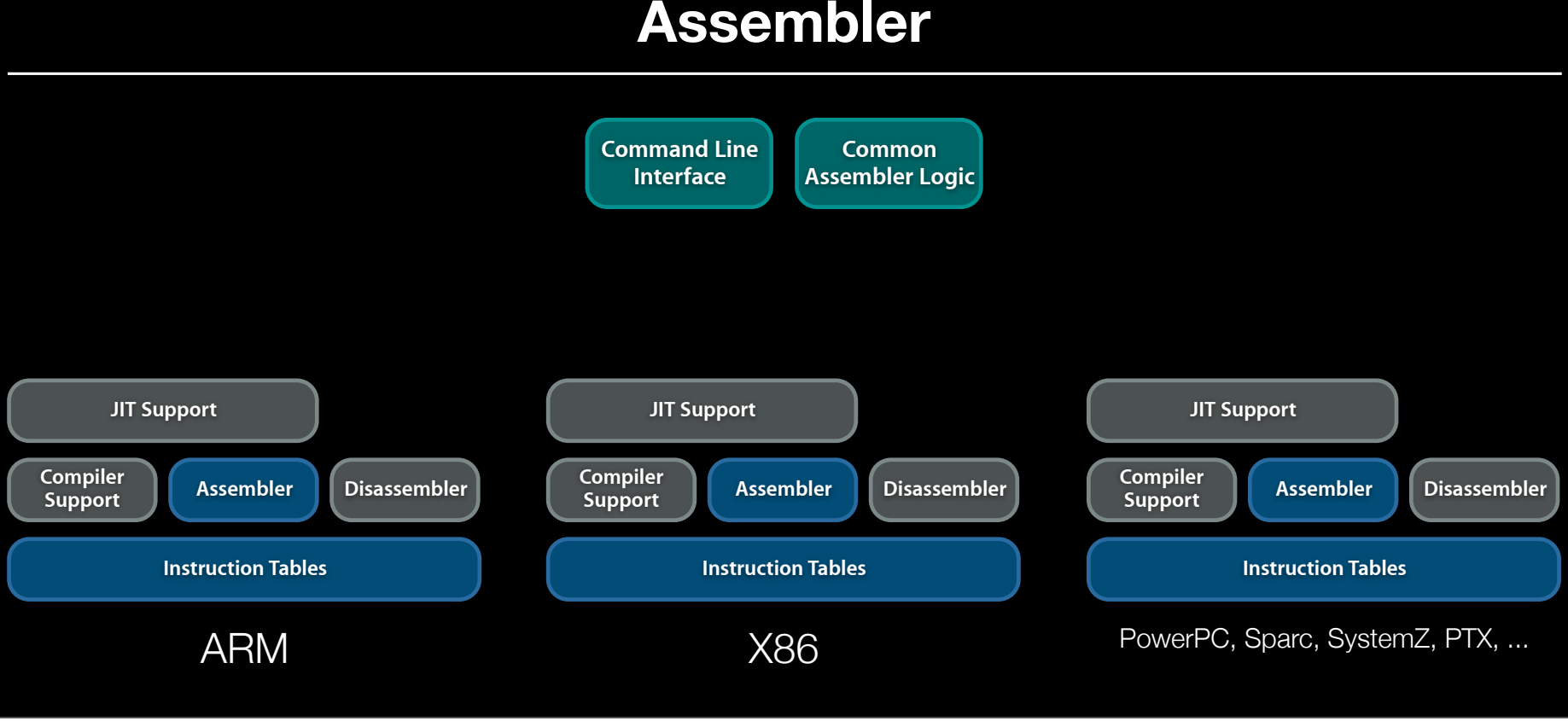
# In 2013, this is not good enough!

- Great compilers are a huge investment:
  - Source code analysis framework
  - Machine specific code generation
  - Performance optimization
- Other tools want these capabilities too!
  - Compiler “plugins” are not enough

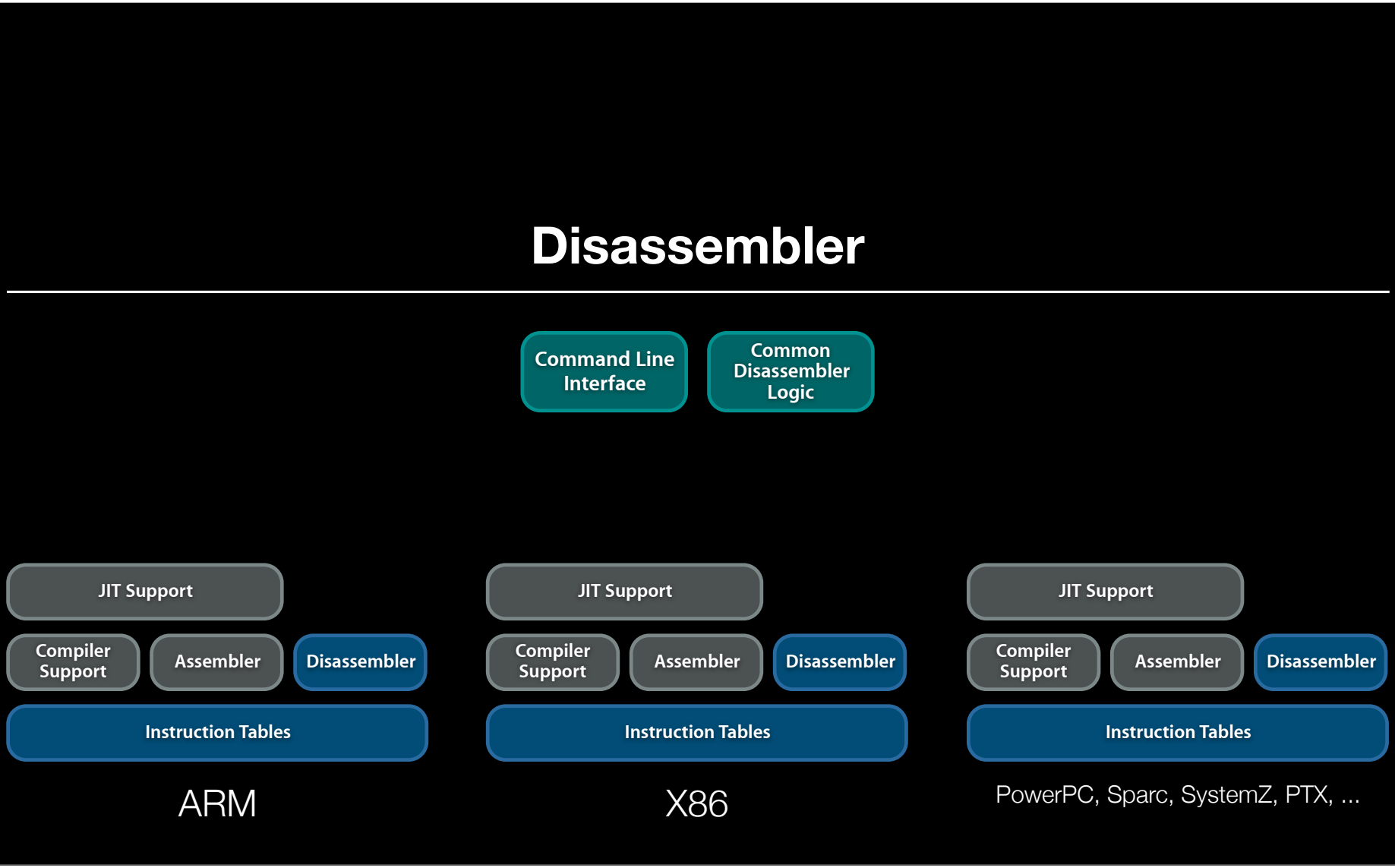
# Decomposing a processor target in LLVM



# Building an Assembler

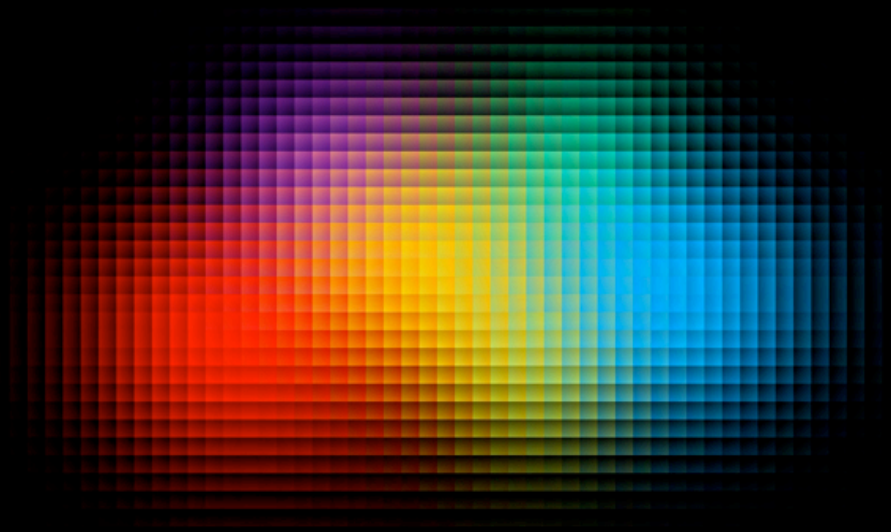


# Disassembler



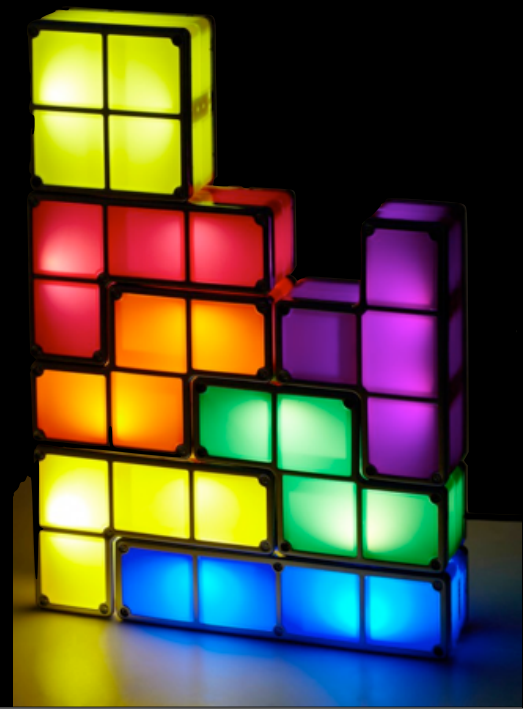
# Advantages of this Design

- One truth for instructions:
  - New features (e.g. AVX-512) added in one place
  - Assembler, disassembler, and compiler support all agree
- Compiler gets integrated assembler
- JIT encodings tested by static compiler
- Clients decide what features they need



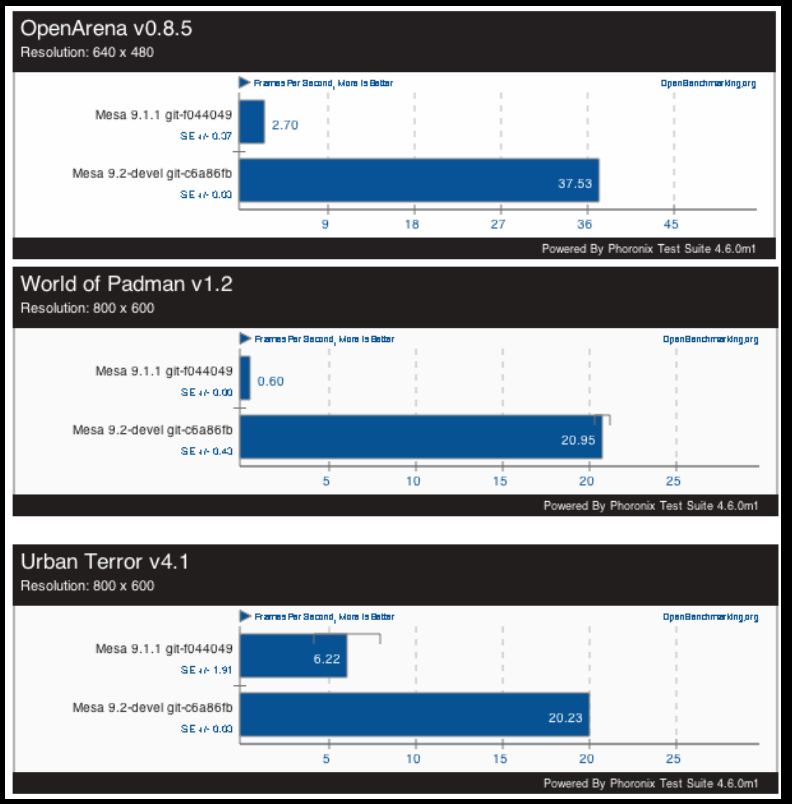
# Compiler Infrastructure?

- Library-based design
  - Modularity
  - Proper layering
  - Testability
- Follows “textbook” compiler design
  - Frontend, optimizer, backend
  - … with enforced layers
- Enables building things we never anticipated!




Applications of LLVM

# mesa 3d - LLVMpipe Software Rasterizer


Benchmarks from phoronix.com

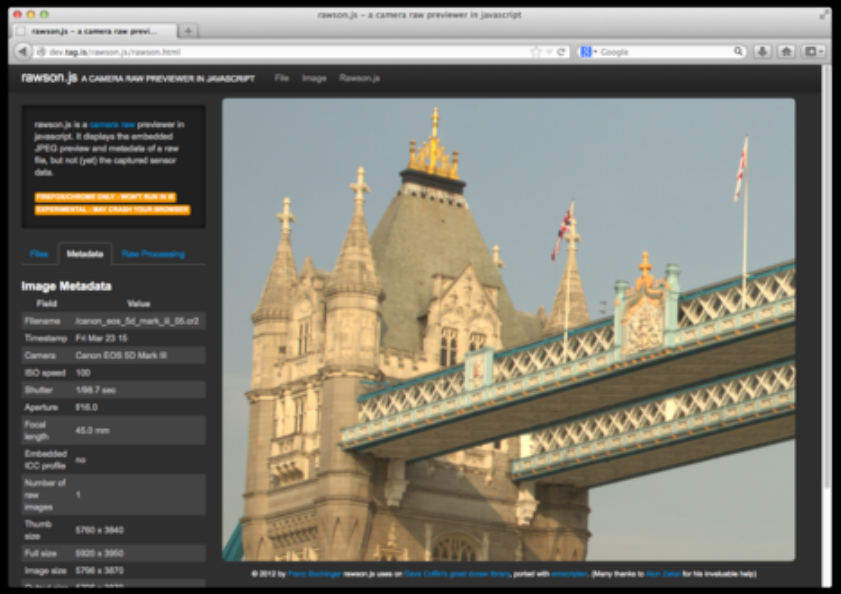
# Open Shading Language

- Special effects rendering engine:
  - Quality is everything
  - Huge: > 200GB per scene
  - 4-10 hours/frame
  - Many thousands of cores
- Driven by Sony Pictures Imageworks
  - Used in several well-known pictures

http://llvm.org/devmtg/2010-11/


Compile just about anything to Javascript!

Epic Citadel




rawson.js

https://github.com/kripken/emscripten/wiki

# Commercial Language Implementation

Xcode
Apple

C, C++, Objective-C

embarcadero
C++ Builder

CRAY
THE SUPERCOMPUTER COMPANY
FORTRAN


Apple, Intel, AMD,
NVidia, Rapidmind,
Gallium3d, ...


Adobe Pixel
Bender


C#, Cross Platform


# Research and Independent Languages


Rust

julia



LDC

LLVM D compiler


LLVM Pascal Compiler

Intel SPMD Program Compiler

# Clang Compiler

http://clang.llvm.org

# Clang - “C Lang”uage Family

- Compiles C, C++, and Objective-C
  - Drop-in compatible with GCC & Visual Studio (wip)
- Only compiler with:
  - Full C++’11 language and library
  - Modern Objective-C
- Follows the LLVM library-based “infrastructure” design
  - Builds on powerful LLVM backend
  - Reusable in other tools

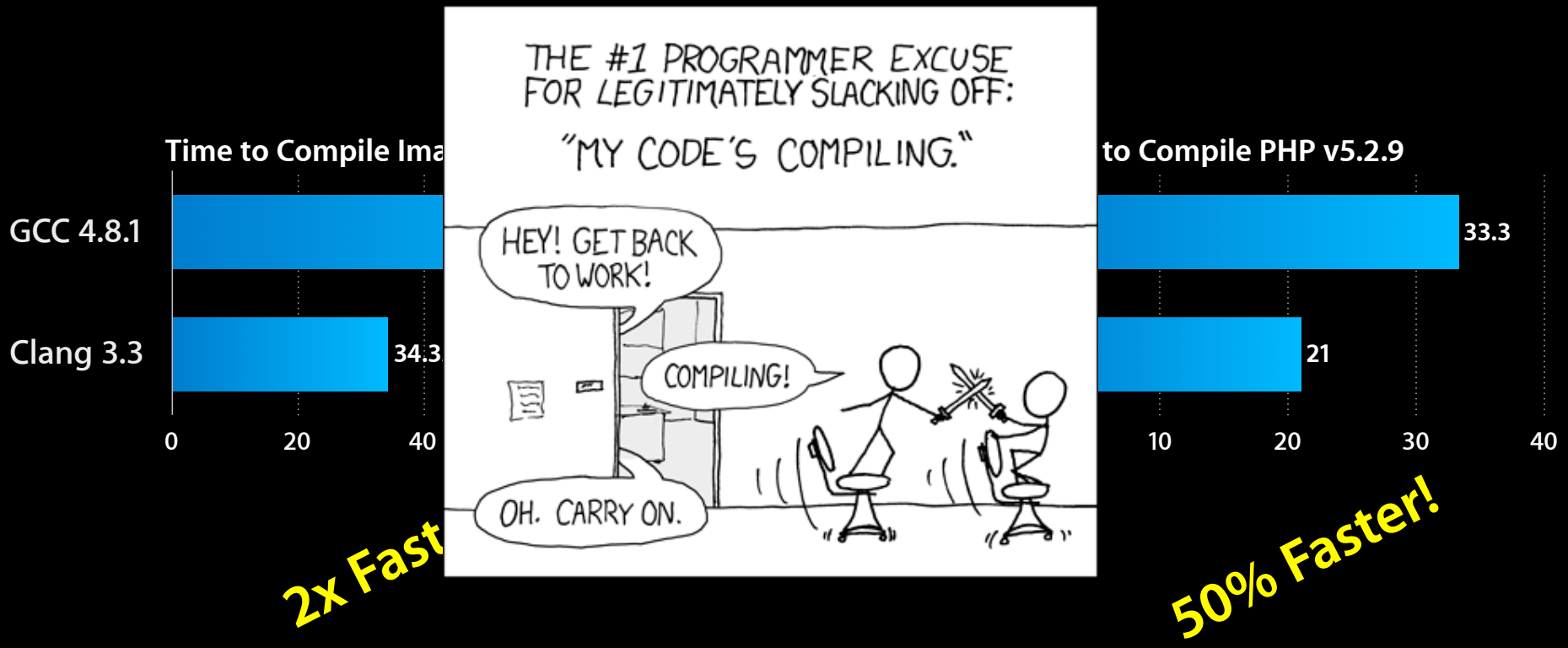
# Clang has great diagnostics

```
kosh ~ 105 % clang pointmain.c -o pointmain -g
In file included from pointmain.c:2:
./point.h:6:2: error: expected ';' after struct
}
 ^
 ;
pointmain.c:6:37: error: no member named 'horisontal' in 'struct Point'; did you mean 'horizontal'?
        printf("%d, %d\n", p1.vertical, p1.horisontal);
                                        ^~~~~~~~~~
                                        horizontal
In file included from pointmain.c:2:
./point.h:5:19: note: 'horizontal' declared here
        double vertical, horizontal;
                         ^
pointmain.c:6:11: warning: conversion specifies type 'int' but the argument has type 'double' [-Wformat]
        printf("%d, %d\n", p1.vertical, p1.horisontal);
               ~^          ~~~~~~~~~~~
pointmain.c:6:15: warning: conversion specifies type 'int' but the argument has type 'double' [-Wformat]
        printf("%d, %d\n", p1.vertical, p1.horisontal);
                    ~^                  ~~~~~~~~~~~~~
5 diagnostics generated.
kosh ~ 106 %
```

Note: Clang defaults to using C99 mode with warnings enabled

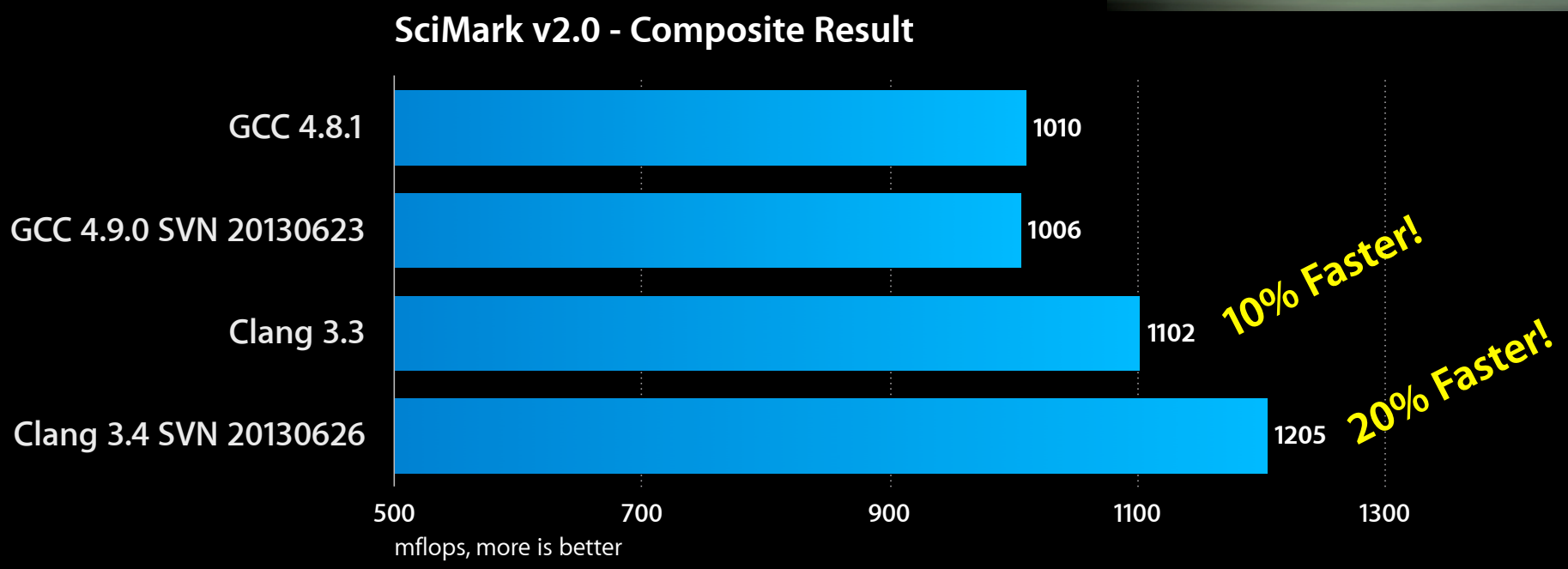
Clang is not confused by the earlier errors:
it still knows that p1.horizontal was intended

# Clang compiles fast




http://www.phoronix.com/scan.php?page=article&item=intel_haswell_llvm33

# Generates fast code


http://www.phoronix.com/scan.php?page=article&item=llvm_clang34_first

# Clang Applications

- Clang static analyzer http://clang-analyzer.llvm.org

```
if (v1 > 0)
{
  if (v2 == 0)
  {
    myName = [[NSString alloc] initWithString:name1];
  }
  else if (v1 > v2)
  {
    myName = [[NSString alloc] initWithString:name2];
  }
}
else
{
  myName = [[NSString alloc] initWithString:name3];
}
myKey = [myName mutableCopy];   Receiver in message expression is an uninitialized value
```

- Address Sanitizer http://clang.llvm.org/docs/AddressSanitizer.html
- Clang Format http://clang.llvm.org/docs/ClangFormat.html
- Many more...

# and so much more...

| | |
|---|---|
| http://**lldb**.llvm.org/ | LLDB Debugger |
| http://**lld**.llvm.org/ | LLD Linker |
| http://**libcxx**.llvm.org/ | C++ Standard Library |
| http://**compiler-rt**.llvm.org/ | Compiler Runtime |
| http://**dragonegg**.llvm.org/ | GCC Plugin |
| http://**openmp**.llvm.org/ | OpenMP Runtime |

**http://llvm.org/**

# LLVM Compiler Infrastructure

## High technology in service of great applications and tools


**http://llvm.org/**