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

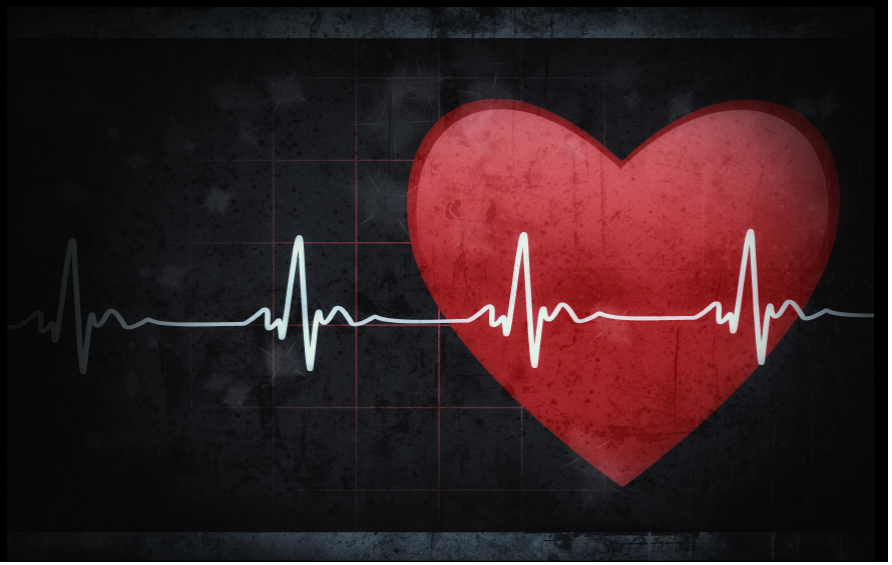
LLVM and Clang
Advancing Compilers and Tools

Chris Lattner
http://llvm.org

October 25, 2013

LLVM is everywhere

- Industry
- Open Source
- Academia



… for many different things

… for many different things

- System compiler for Apple and FreeBSD platforms

… for many different things

- System compiler for Apple and FreeBSD platforms
- Used by most GPGPU implementations

… for many different things

- System compiler for Apple and FreeBSD platforms
- Used by most GPGPU implementations
- Many new language implementations

… 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

… for many different things

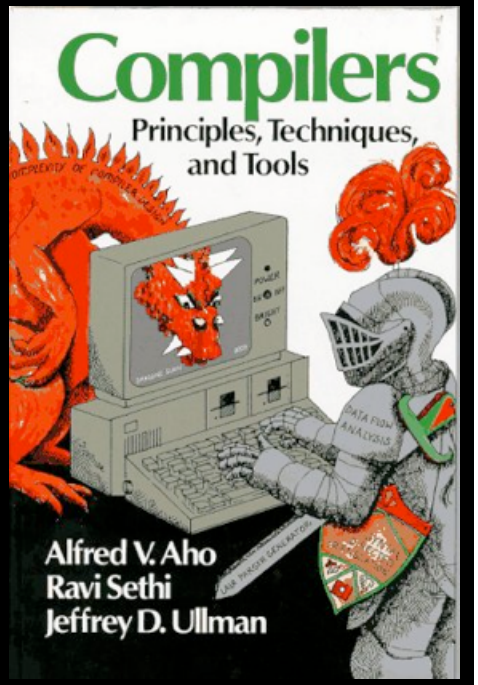
- System compiler for Apple and FreeBSD platforms
- Used by most GPGPU implementations
- Many new language implementations
- Finding bugs in source code
- Special effects in movies

… 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?

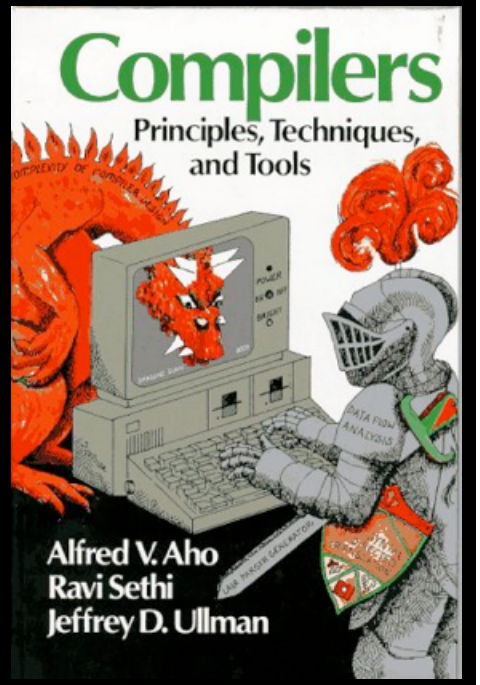
What is a compiler?



What is a compiler?

com·pil·er
noun

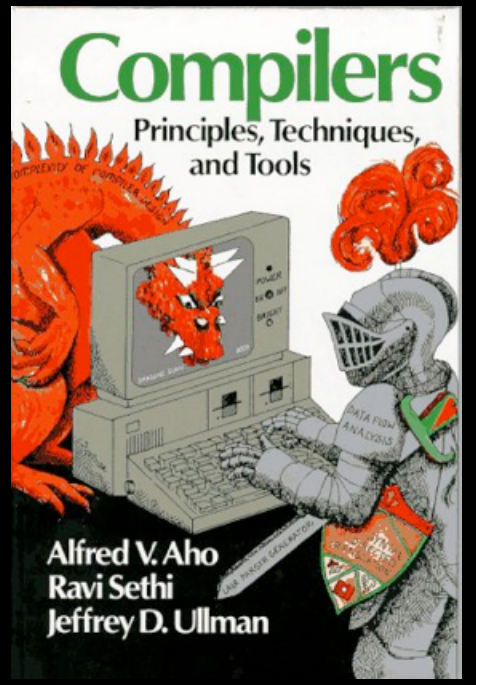
1. a person who compiles information (as for reference purposes): *a compiler of anthologies.*



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.

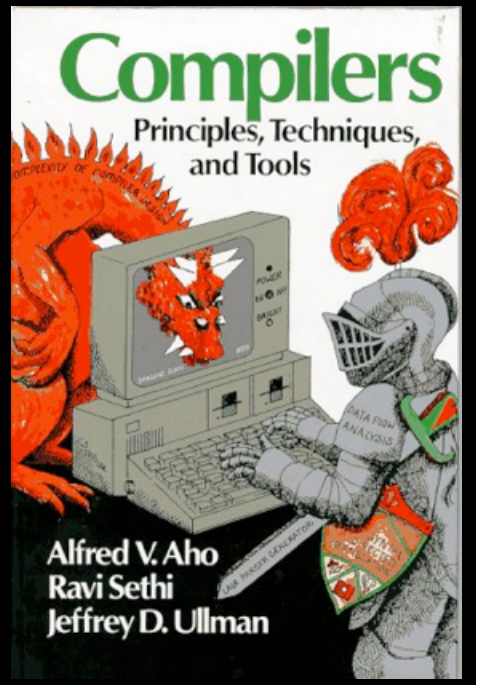


What is a compiler?

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.

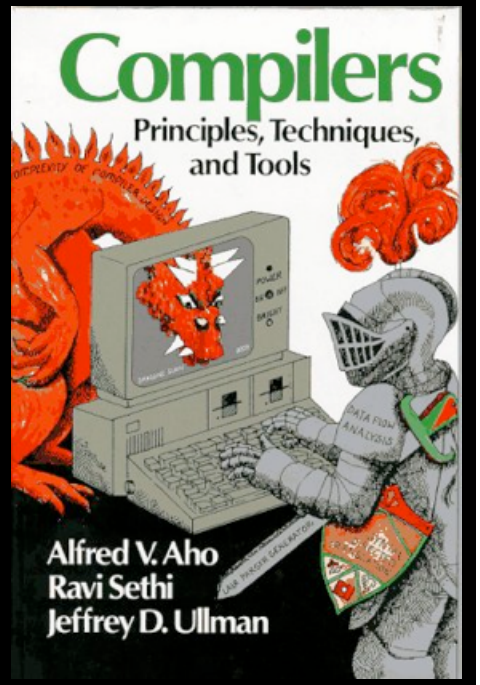
- Clang, GCC, ICC, MSVC++ are compilers



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, GCC, ICC, MSVC++ are compilers
- LLVM is not. What is LLVM?

What is LLVM?

llvm.org is an open source umbrella project

What is LLVM?

llvm.org is an open source umbrella project

What is LLVM?

llvm.org is an open source umbrella project

- Strong community, with shared values:
  - Common processes, patch review, etc
  - Common design approaches
  - Preference for MIT/BSD License

What is LLVM?

llvm.org is an open source umbrella project

- Strong community, with shared values:
  - Common processes, patch review, etc
  - Common design approaches
  - Preference for MIT/BSD License

- Provides useful tools:
  - Assembler, linker, compiler, debugger, and more

What is LLVM?

llvm.org is an open source umbrella project

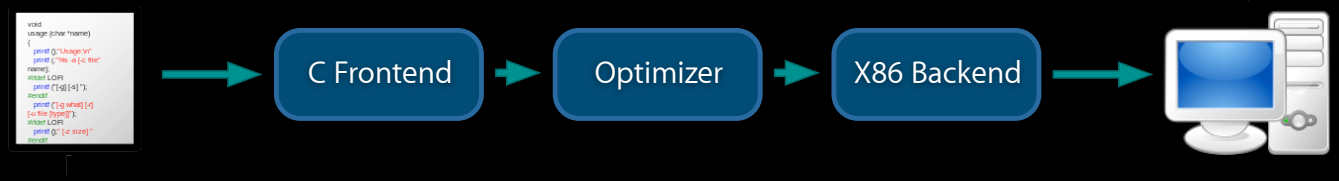
- Strong community, with shared values:
  - Common processes, patch review, etc
  - Common design approaches
  - Preference for MIT/BSD License
- Provides useful tools:
  - Assembler, linker, compiler, debugger, and more
- LLVM is a compiler **infrastructure**!


Compiler Infrastructure 101

How does a compiler work?

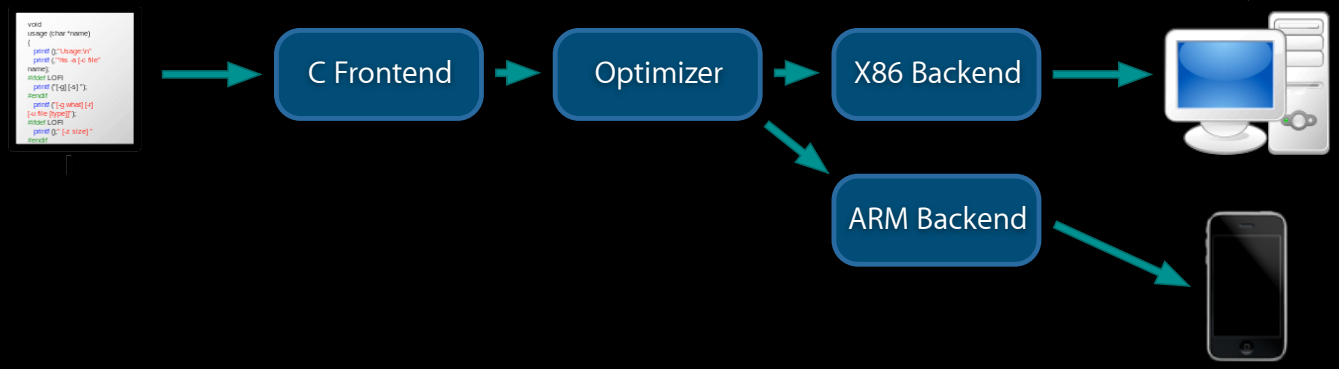
How does a compiler work?

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



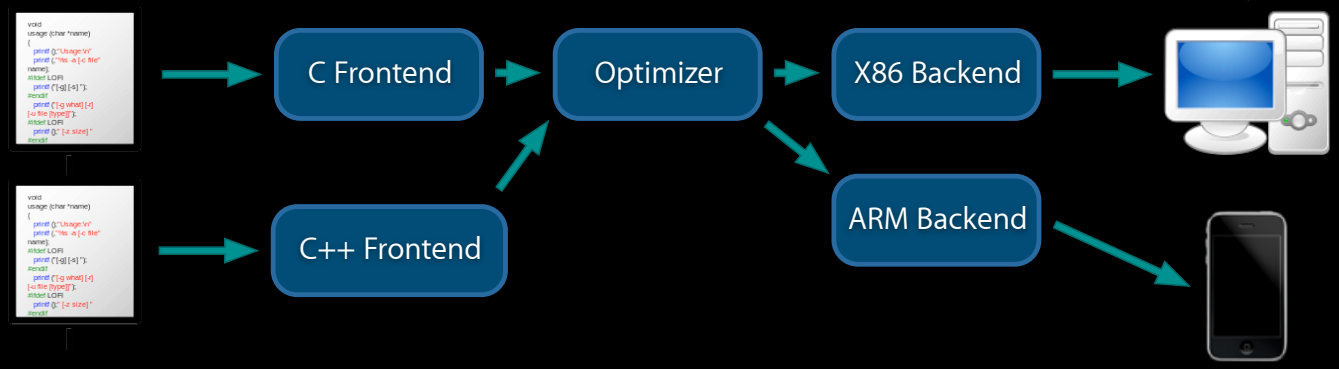
How does a compiler work?

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



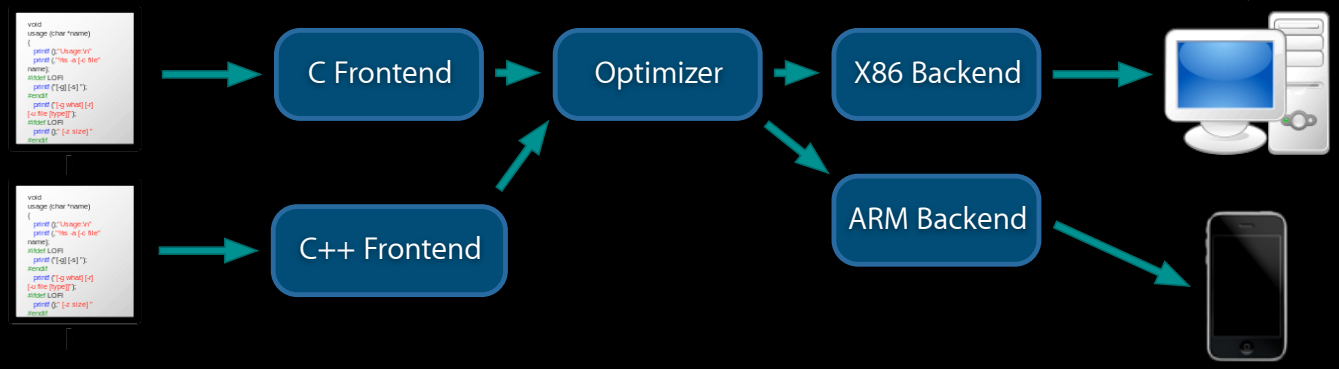
How does a compiler work?

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



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!

In 2013, this is not good enough!

- Great compilers are a huge investment:
  - Source code analysis framework
  - Machine specific code generation
  - Performance optimization

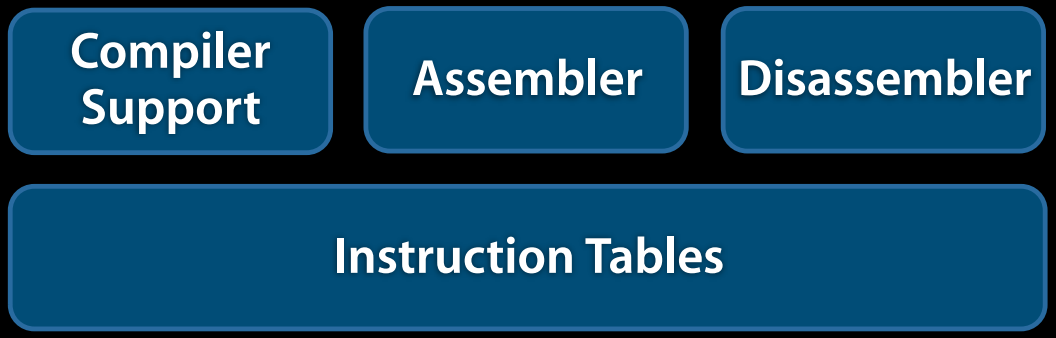
In 2013, this is not good enough!

- 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

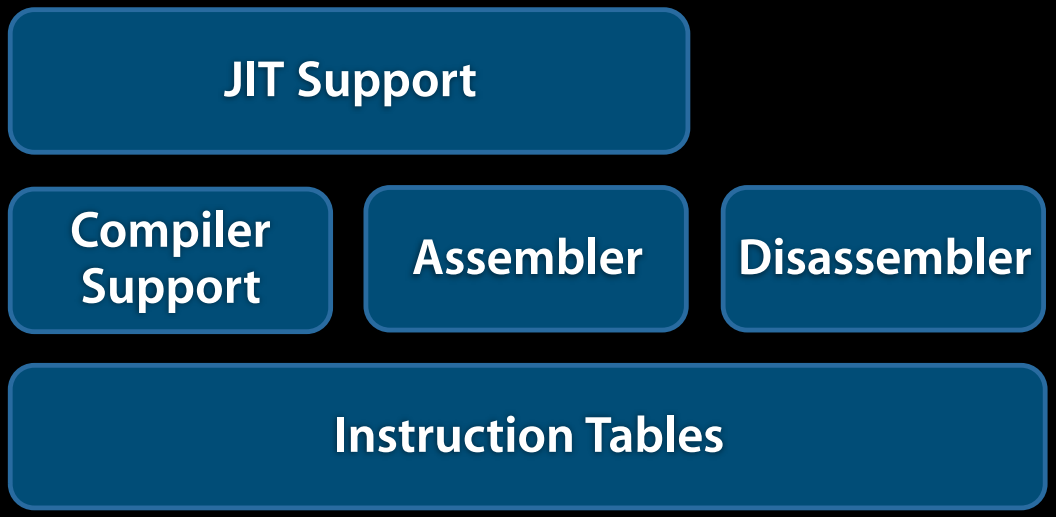
Decomposing a processor target in LLVM

Decomposing a processor target in LLVM

Decomposing a processor target in LLVM

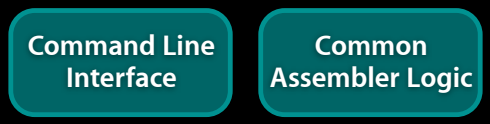


Decomposing a processor target in LLVM

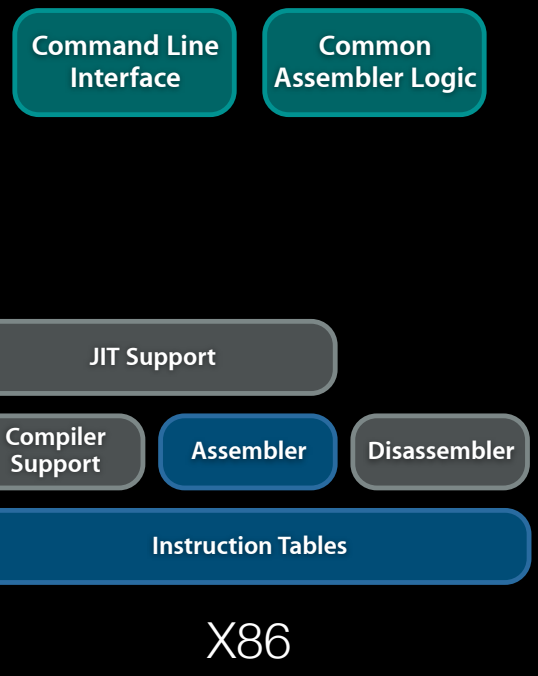


Building an Assembler

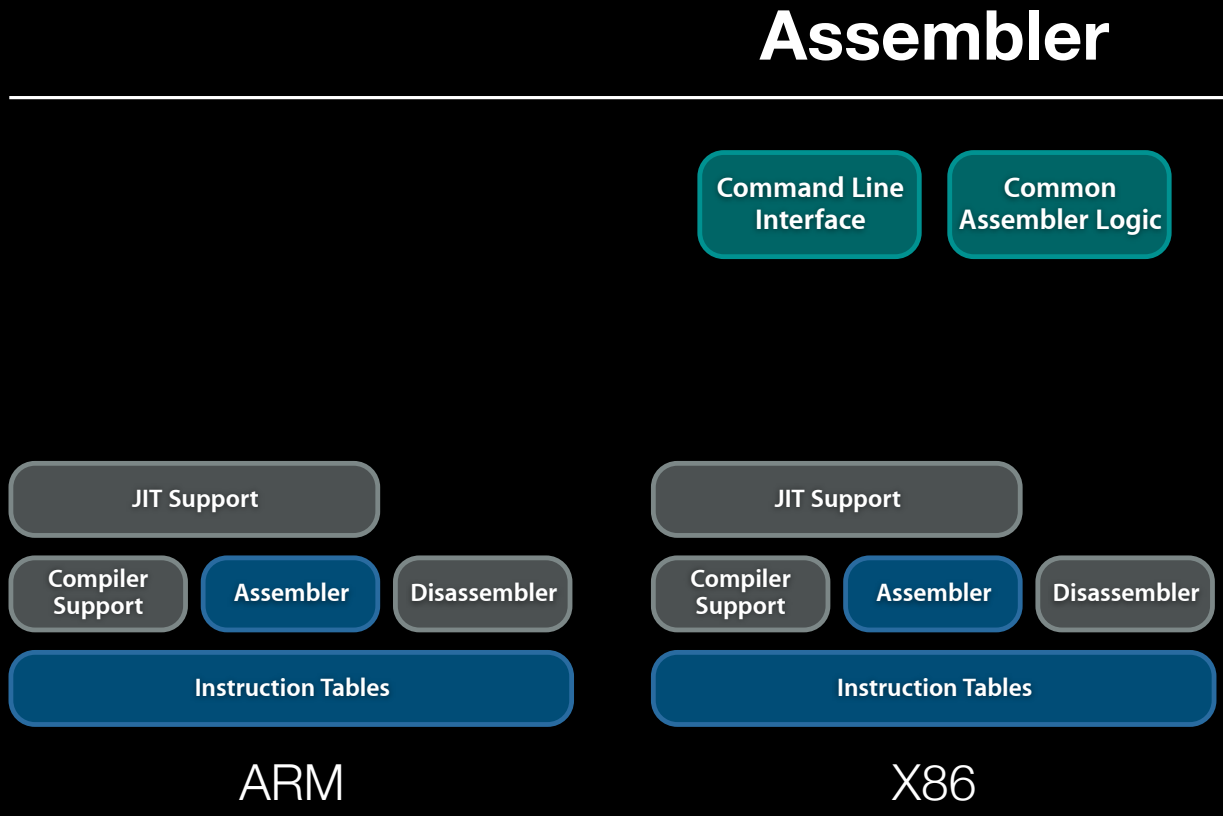
Building an Assembler



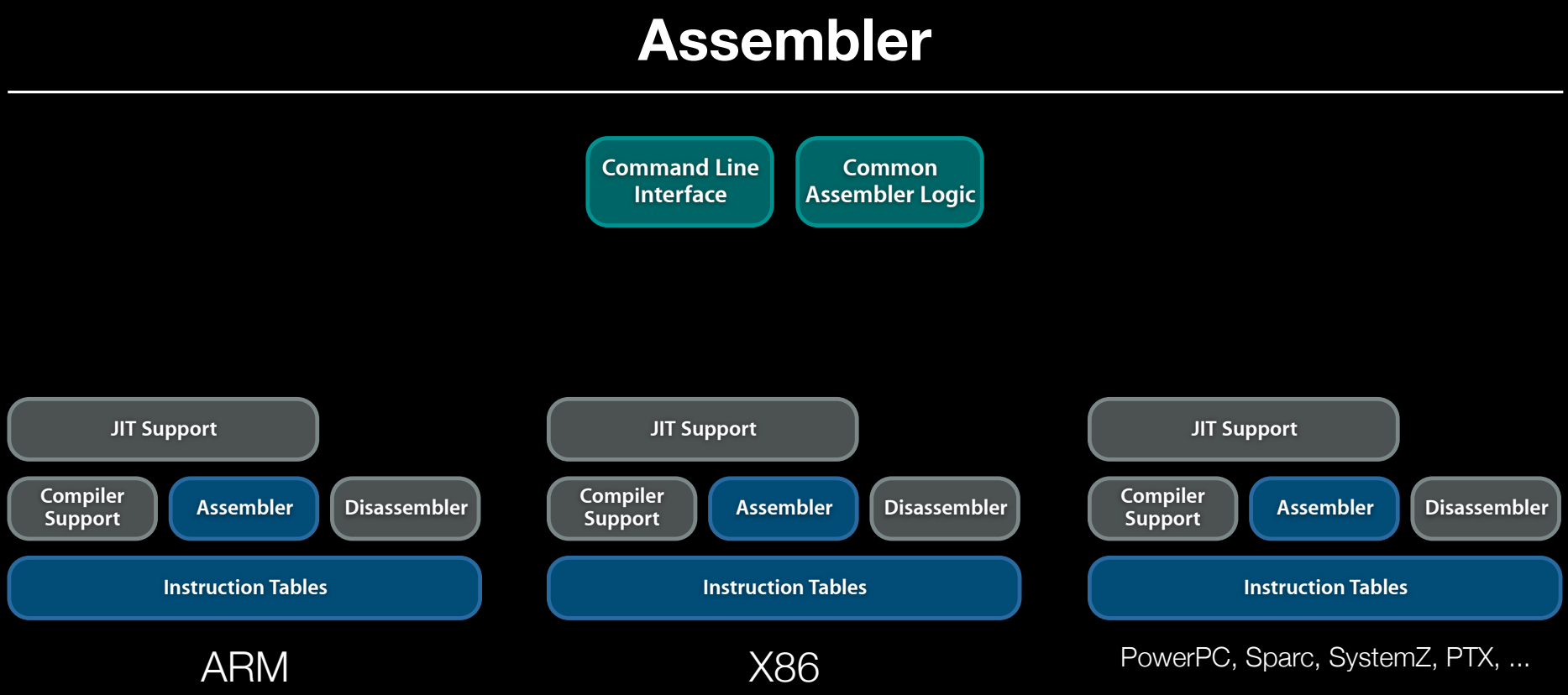
Building an Assembler



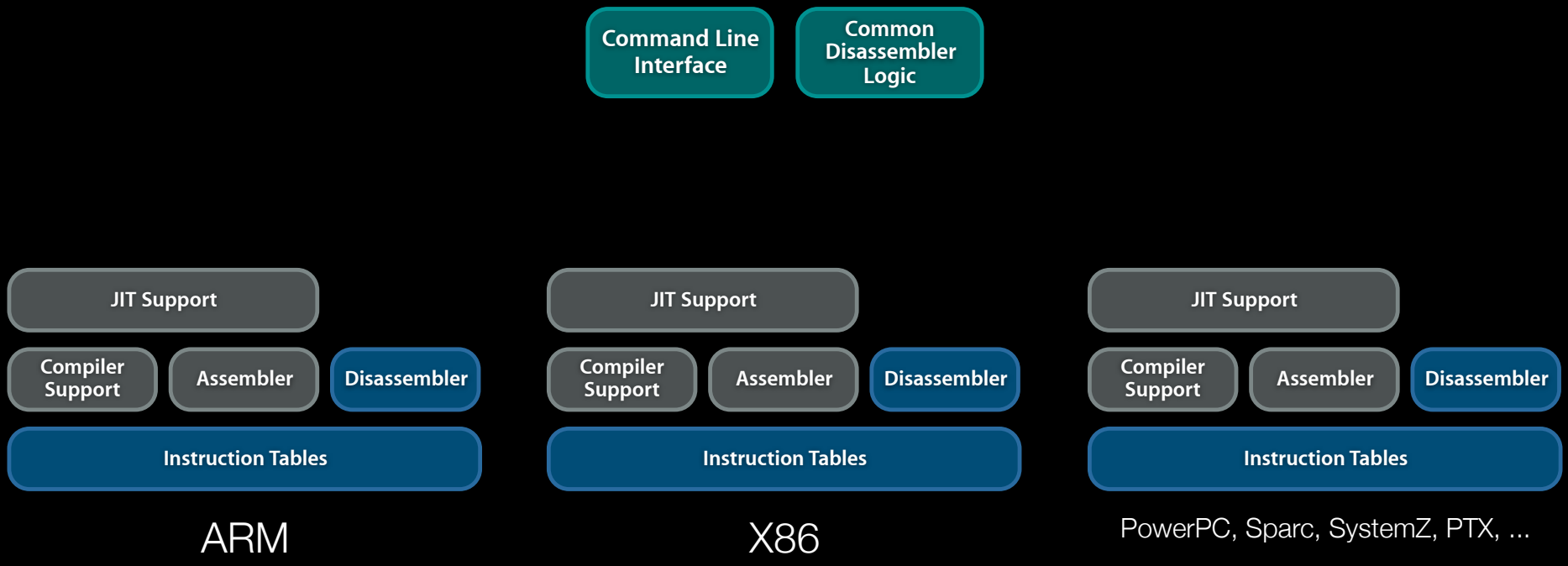
Building an Assembler



Building an Assembler

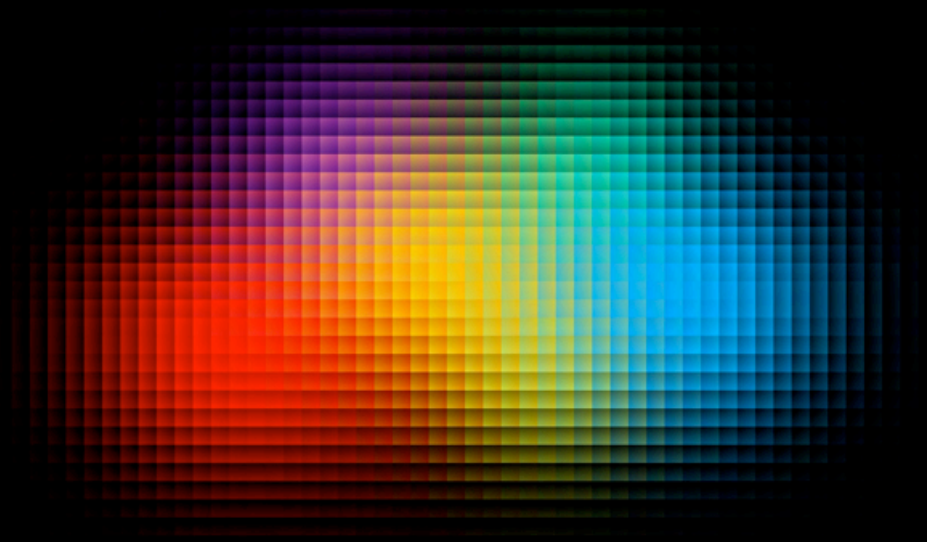


Disassembler



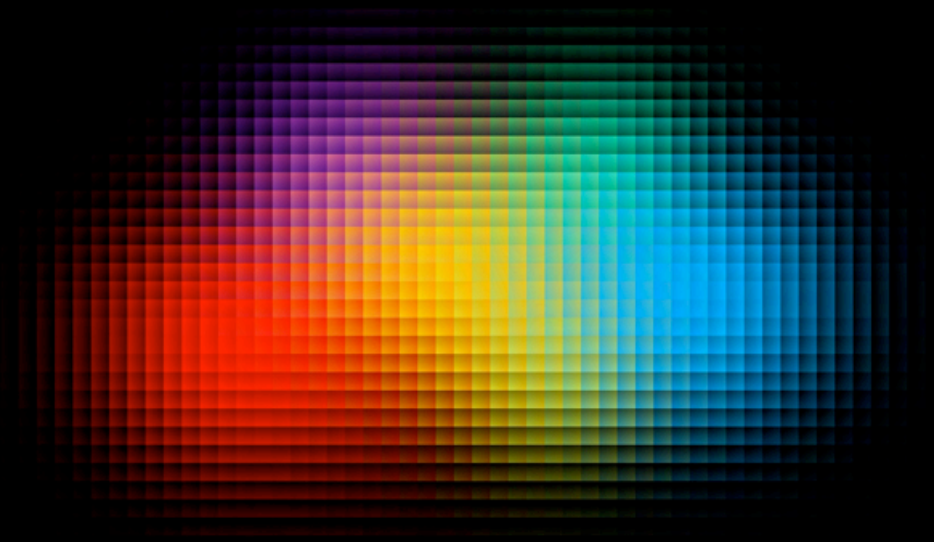
Advantages of this Design

- One truth for instructions:
  - New features (e.g. AVX-512) added in one place
  - Assembler, disassembler, and compiler support all agree



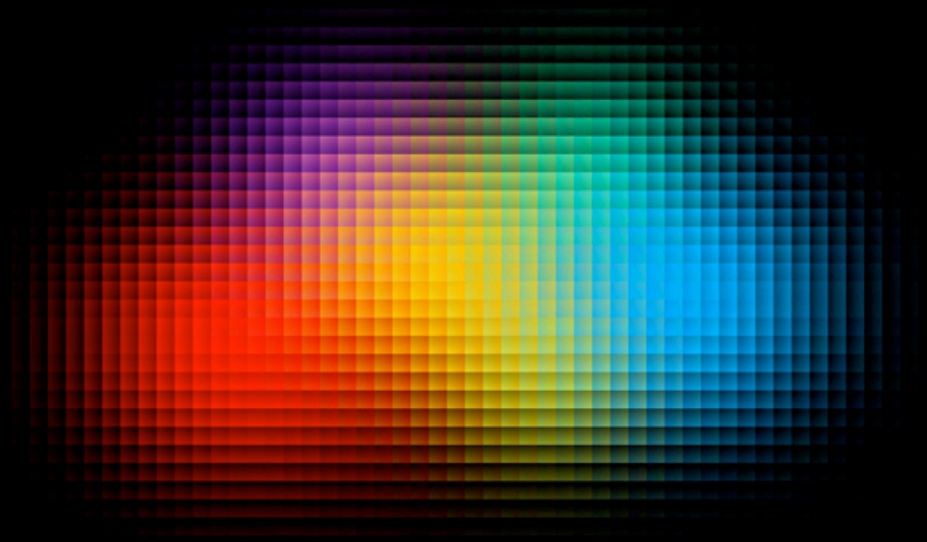
Advantages of this Design

- 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



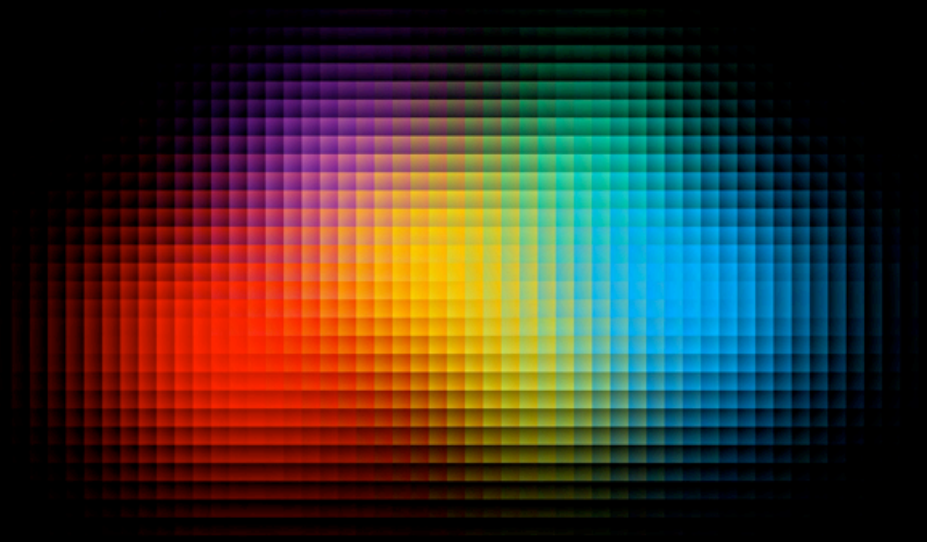
Advantages of this Design

- 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

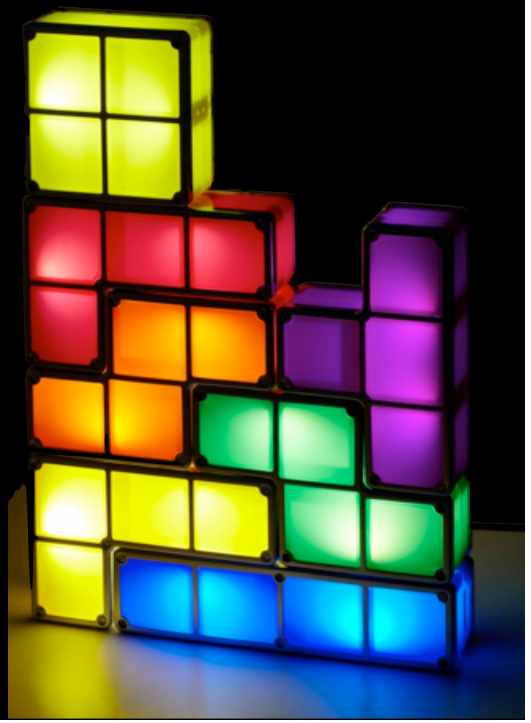


Advantages of this Design

- 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

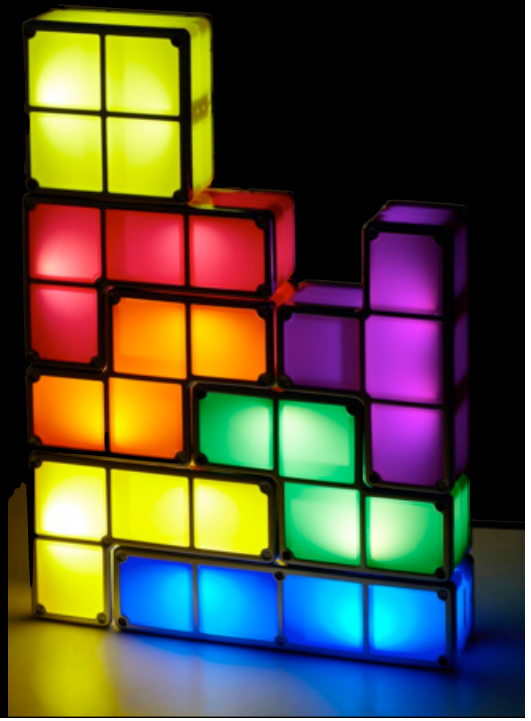


Compiler Infrastructure?



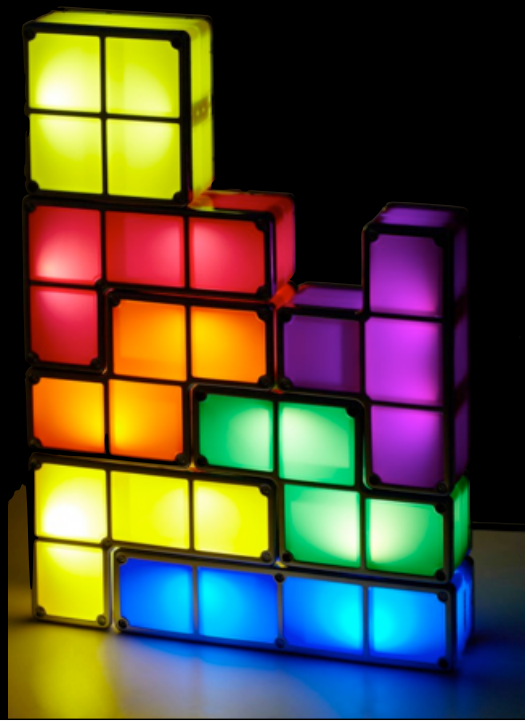
Compiler Infrastructure?

- Library-based design
  - Modularity
  - Proper layering
  - Testability



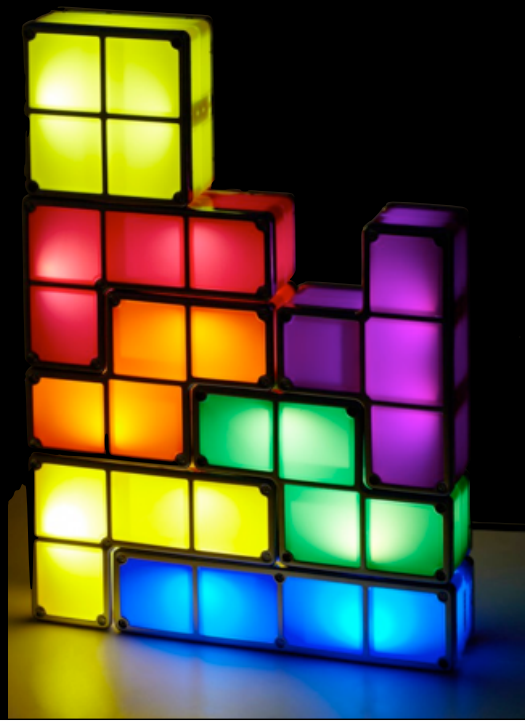
Compiler Infrastructure?

- Library-based design
  - Modularity
  - Proper layering
  - Testability
- Follows “textbook” compiler design
  - Frontend, optimizer, backend
  - … with enforced layers



Compiler Infrastructure?

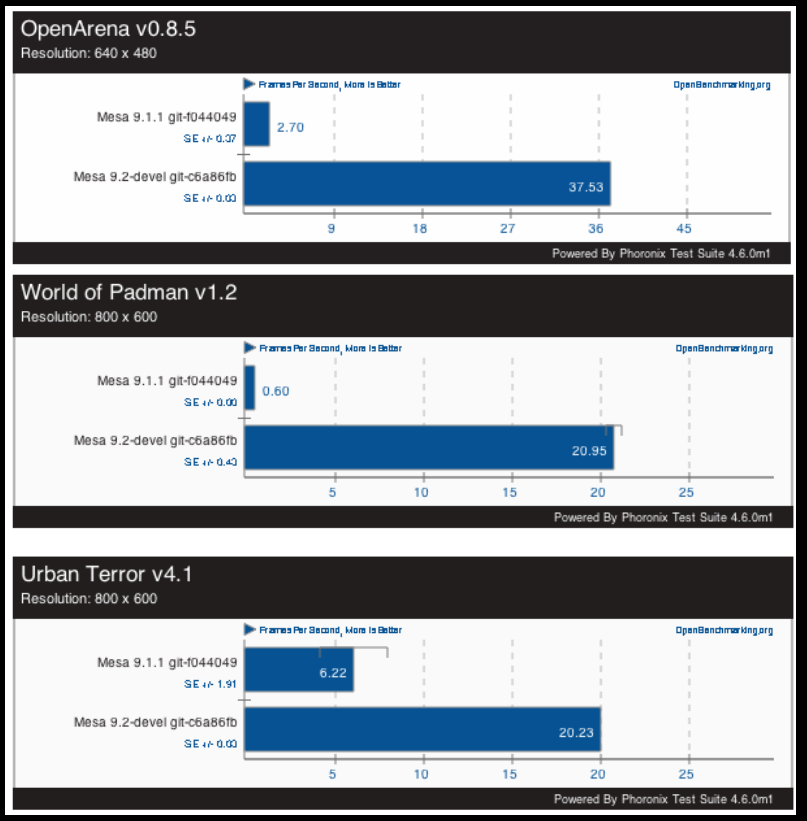
- Library-based design
  - Modularity
  - Proper layering
  - Testability
- Follows “textbook” compiler design
  - Frontend, optimizer, backend
  - … with enforced layers
- Enables building things we never anticipated!




Applications of LLVM

mesa 3d - LLVMpipe Software Rasterizer

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

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

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!

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

Compile just about anything to Javascript!

Epic Citadel

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

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

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

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

LLVM D compiler

LLVM Pascal Compiler

Intel SPMD Program Compiler

Clang Compiler

http://clang.llvm.org


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)

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

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

http://clang.llvm.org/diagnostics.html

Clang has great diagnostics

```
$ clang t.c
t.c:8:36: error: invalid operands to binary expression ('int' and 'struct A')
  X = X + func(X ? ((SomeA.F + 40) + SomeA) / 42 + SomeA.F : Ptr->F);
                   ~~~~~~~~~~~~~~~ ^ ~~~~~
```

http://clang.llvm.org/diagnostics.html

Clang has great diagnostics

```
$ clang t.c
t.c:8:36: error: invalid operands to binary expression ('int' and 'struct A')
  X = X + func(X ? ((SomeA.F + 40) + SomeA) / 42 + SomeA.F : Ptr->F);
                   ~~~~~~~~~~~~~~~ ^ ~~~~~
```

```
$ clang t.c
t.c:9:7: error: invalid operands to binary expression ('int' and 'struct A')
  X = MAX(X, *Ptr);
      ^~~~~~~~~~~~
t.c:2:24: note: instantiated from:
#define MAX(A, B) ((A) > (B) ? (A) : (B))
                   ~~~ ^ ~~~
```

http://clang.llvm.org/diagnostics.html

Clang has great diagnostics

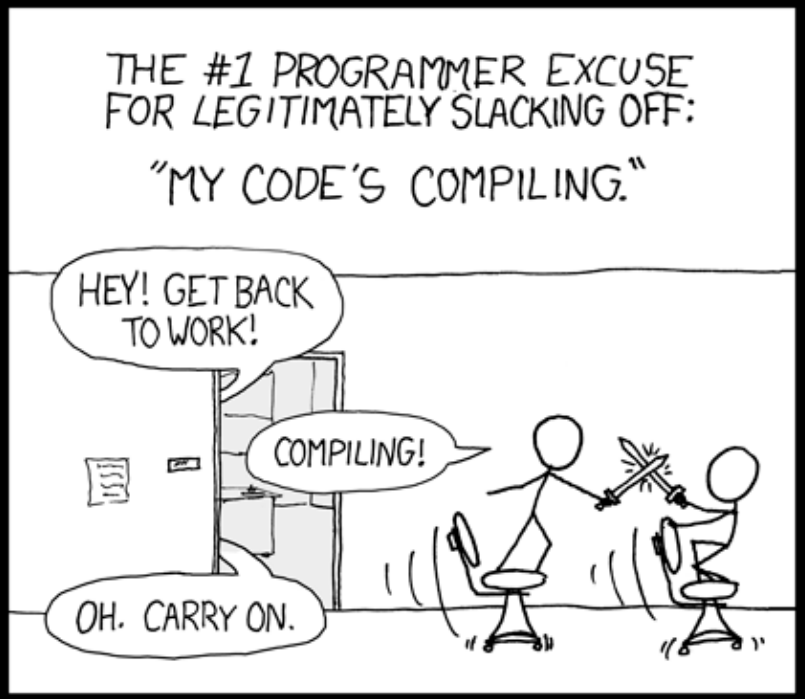
```
$ clang t.c
t.c:8:36: error: invalid operands to binary expression ('int' and 'struct A')
  X = X + func(X ? ((SomeA.F + 40) + SomeA) / 42 + SomeA.F : Ptr->F);
                   ~~~~~~~~~~~~~~ ^ ~~~~~
```

```
$ clang t.c
t.c:9:7: error: invalid operands to binary expression ('int' and 'struct A')
  X = MAX(X, *Ptr);
      ^~~~~~~~~~~~
t.c:2:24: note: instantiated from:
#define MAX(A, B) ((A) > (B) ? (A) : (B))
                   ~~~ ^ ~~~
```

```
$ clang t.cpp
t.cpp:5:3: error: no template named 'vector'; did you mean 'std::vector'?
  vector<int> V;
  ^~~~~~
  std::vector
```

http://clang.llvm.org/diagnostics.html

Clang compiles fast



Clang compiles fast

http://xkcd.com/303/

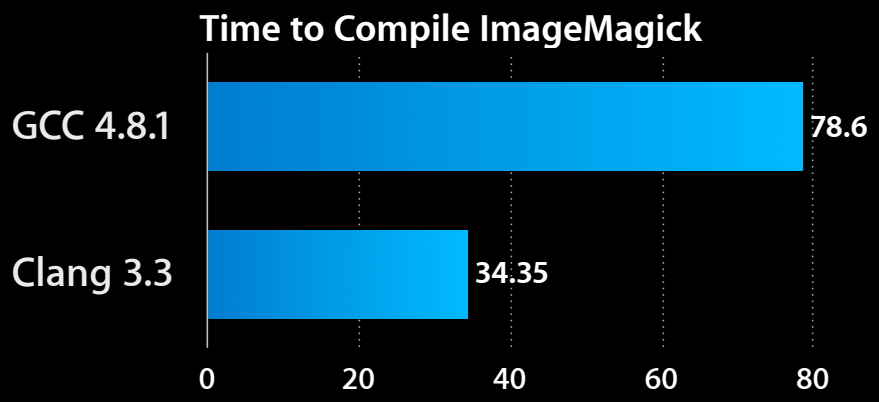
Clang compiles fast

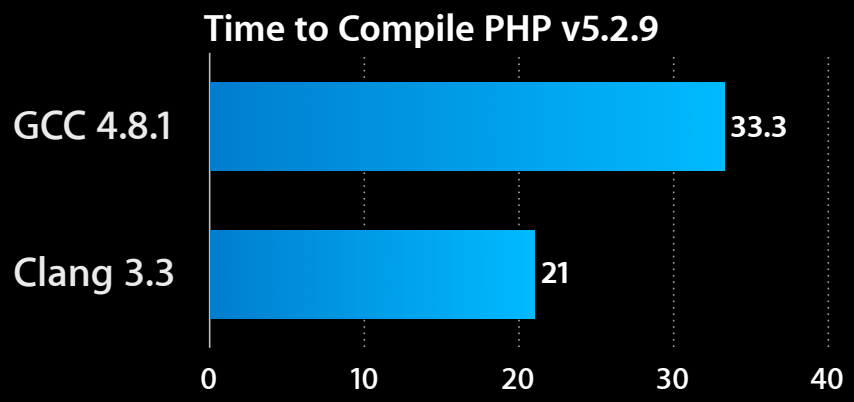
http://xkcd.com/303/

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

Clang compiles fast

http://xkcd.com/303/

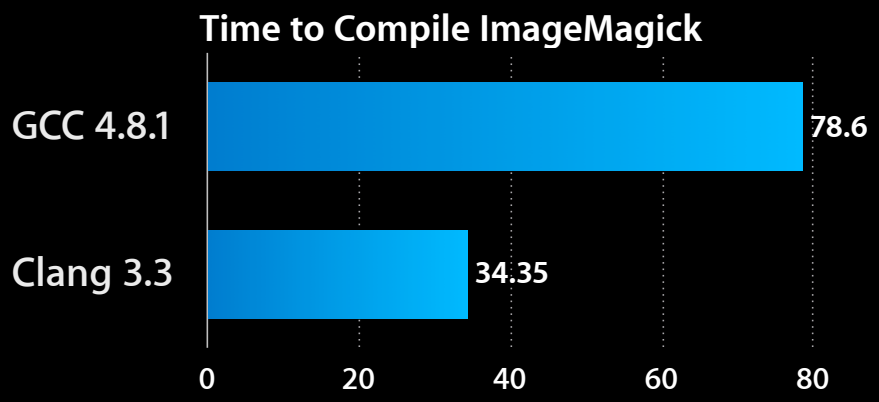




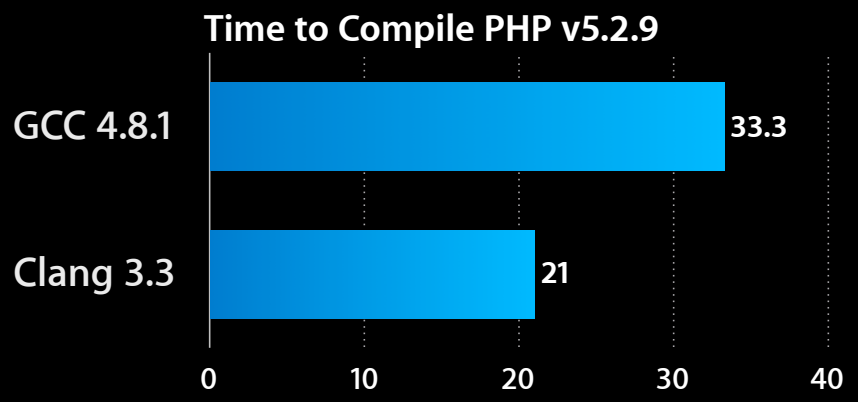
http://www.phoronix.com/scan.php?page=article&item=intel_haswell_llvm33

Clang compiles fast

http://xkcd.com/303/



2x Faster!



50% Faster!

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

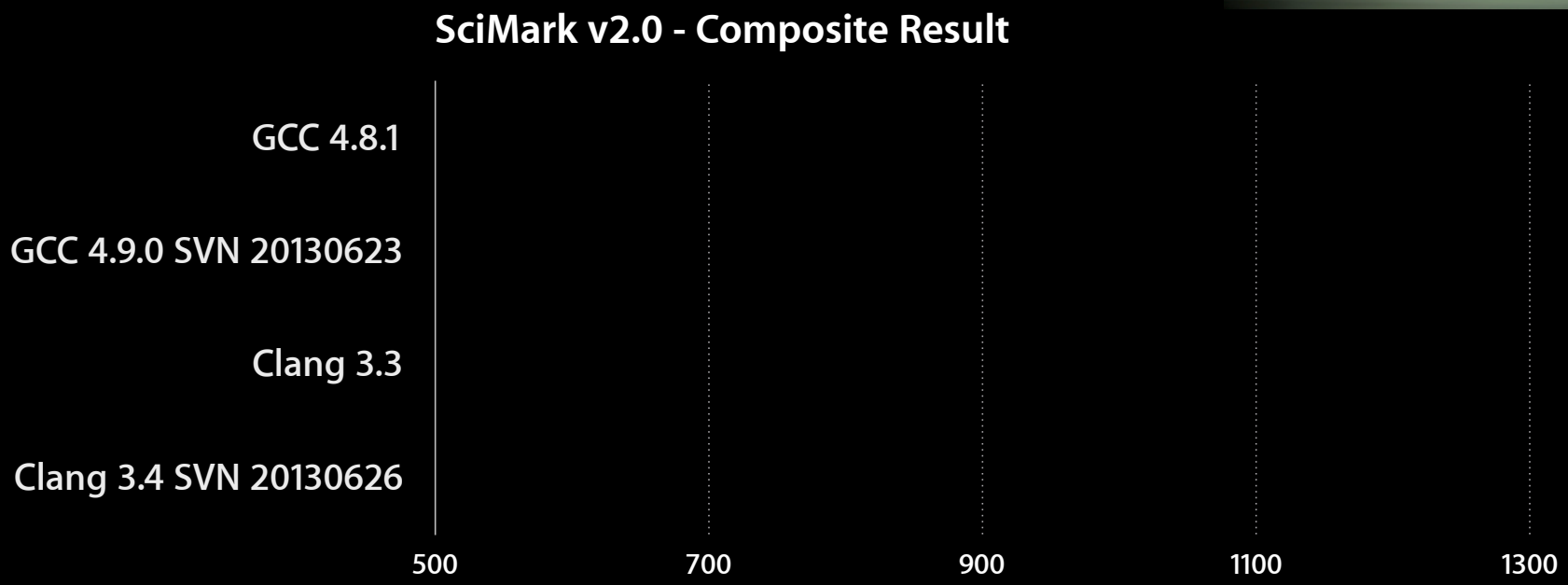
Generates fast code

Generates fast code

SciMark v2.0 - Composite Result

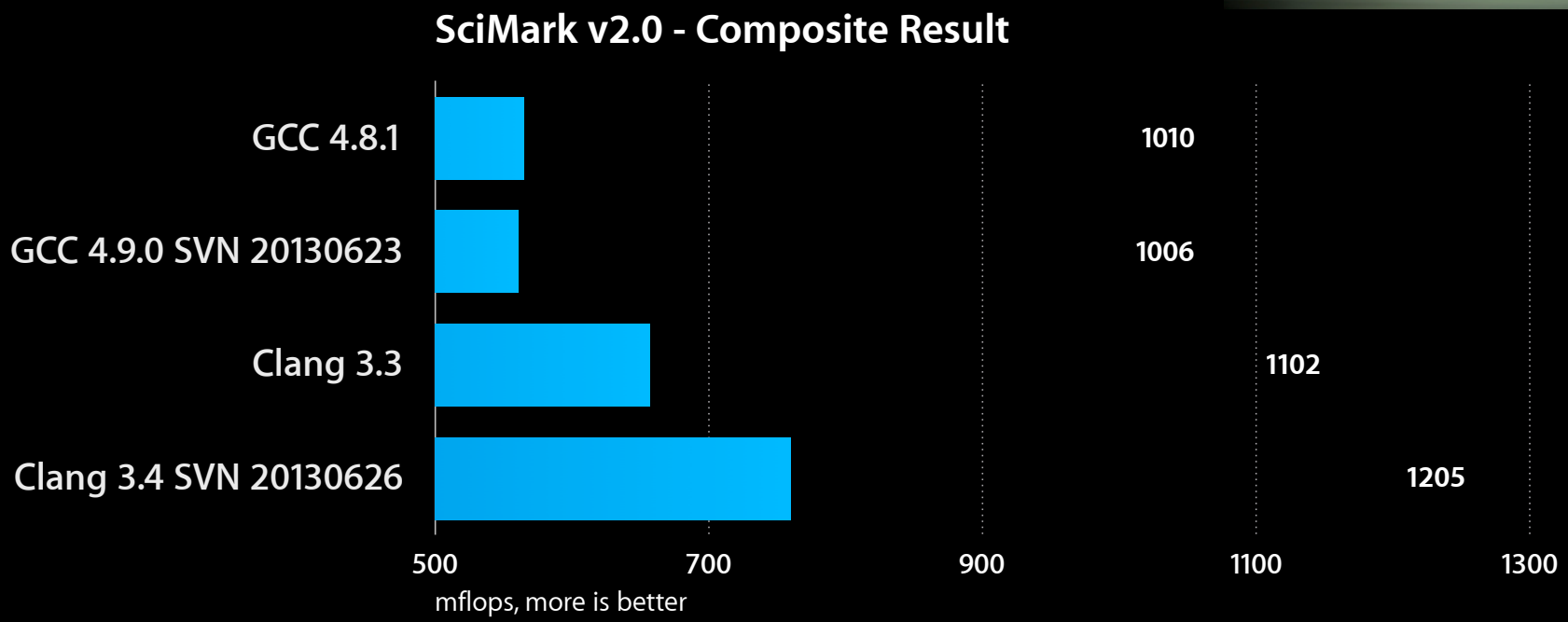
http://www.phoronix.com/scan.php?page=article&item=llvm_clang34_first

Generates fast code



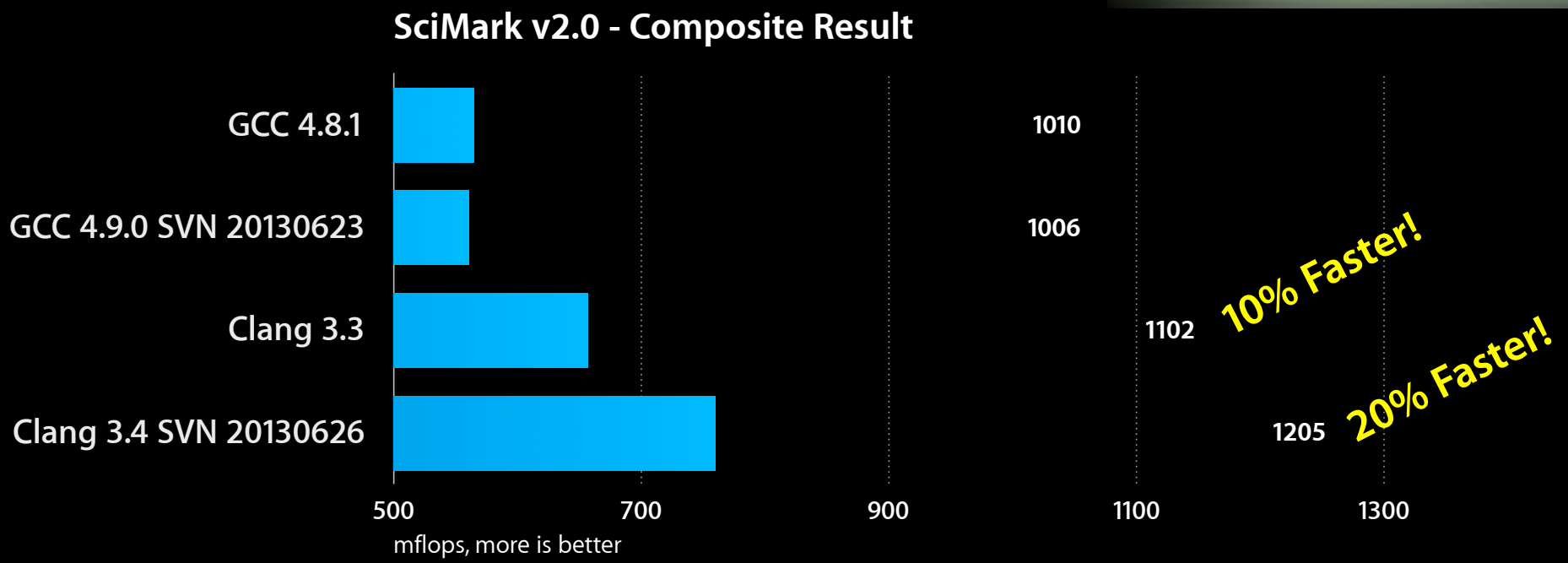
http://www.phoronix.com/scan.php?page=article&item=llvm_clang34_first

Generates fast code



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

Generates fast code



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

Clang Applications

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
```

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 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

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...

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

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/

LLVM Compiler Infrastructure

High technology in service of great applications and tools

http://llvm.org/