

A decorative graphic on the left side of the slide, consisting of several overlapping, curved, green bands that create a sense of depth and movement, resembling a stylized tree trunk or a flowing liquid.

mentor
embedded

“Which targets does Clang support?”

EuroLLVM 2014: Lightning Talk

Jonathan Roelofs

jonathan@codesourcery.com

mentor.com/embedded

**Mentor
Graphics**[®]

Android is a trademark of Google Inc. Use of this trademark is subject to Google Permissions.
Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

“Which targets does Clang support?”

“Some stuff doesn’t seem to be documented at ALL.... what are the valid inputs to the ‘-arch’ ... option? ... This really is frustrating.”

– Tim Hill ^[1]

“Which targets does Clang support?”

“I read the man page ... but I haven't been able to find a list of what ‘-march’ options are available.... Could someone point me to a list of supported options?”

– Tim Nackos [2]

“Which targets does Clang support?”

“I think the best way to get the answer is reading the source” – a’Q [3]

“Which targets does Clang support?”

Clearly we need a better answer!

Prior Work

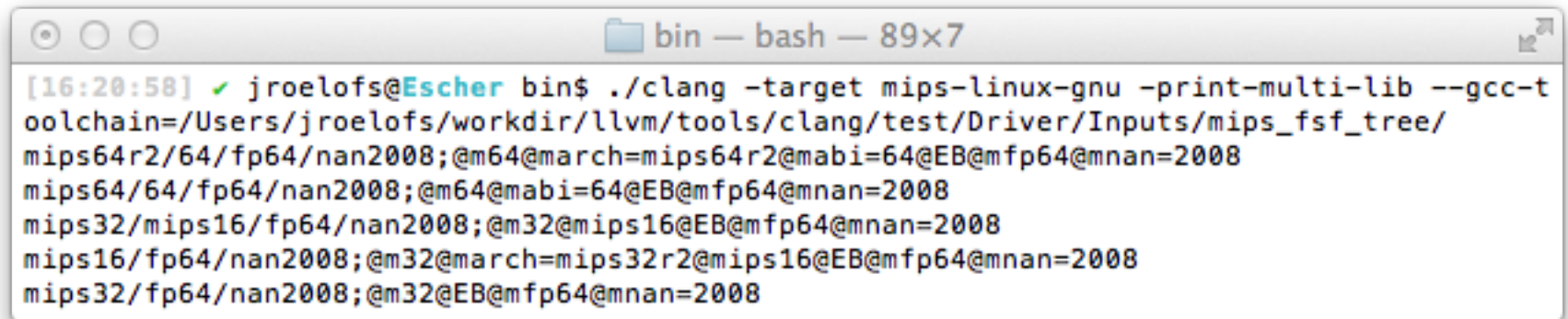
```
bin — bash — 60x33
[14:41:04] ✓ jroelofs@Escher bin$ ./llc --version
LLVM (http://llvm.org/):
  LLVM version 3.5.0svn
  DEBUG build with assertions.
  Built Mar  4 2014 (15:58:16).
  Default target: x86_64-apple-darwin13.1.0
  Host CPU: corei7

Registered Targets:
  aarch64      - AArch64 (ARM 64-bit little endian target)
  aarch64_be  - AArch64 (ARM 64-bit big endian target)
  arm         - ARM
  cpp         - C++ backend
  hexagon     - Hexagon
  mips        - Mips
  mips64      - Mips64 [experimental]
  mips64el    - Mips64el [experimental]
  mipsel      - Mipsel
  msp430      - MSP430 [experimental]
  nvptx       - NVIDIA PTX 32-bit
  nvptx64     - NVIDIA PTX 64-bit
  ppc32       - PowerPC 32
  ppc64       - PowerPC 64
  ppc64le     - PowerPC 64 LE
  r600        - AMD GPUs HD2XXX-HD6XXX
  sparc       - Sparc
  sparcv9     - Sparc V9
  systemz     - SystemZ
  thumb       - Thumb
  x86         - 32-bit X86: Pentium-Pro and above
  x86-64      - 64-bit X86: EM64T and AMD64
  xcore       - XCore
[14:41:11] ✓ jroelofs@Escher bin$
```

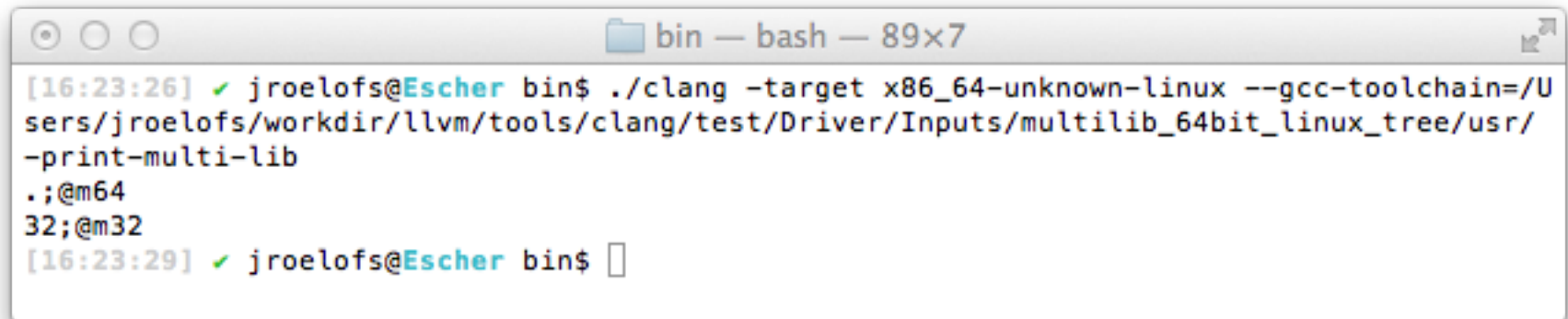
Prior Work

```
$> clang -target <foo> --print-multi-libs
```

(Based on patches I submitted earlier this spring)



```
bin — bash — 89x7
[16:20:58] ✓ jroelofs@Escher bin$ ./clang -target mips-linux-gnu -print-multi-lib --gcc-toolchain=/Users/jroelofs/workdir/llvm/tools/clang/test/Driver/Inputs/mips_fsf_tree/mips64r2/64/fp64/nan2008;@m64@march=mips64r2@mabi=64@EB@mfp64@mnan=2008
mips64/64/fp64/nan2008;@m64@mabi=64@EB@mfp64@mnan=2008
mips32/mips16/fp64/nan2008;@m32@mips16@EB@mfp64@mnan=2008
mips16/fp64/nan2008;@m32@march=mips32r2@mips16@EB@mfp64@mnan=2008
mips32/fp64/nan2008;@m32@EB@mfp64@mnan=2008
```



```
bin — bash — 89x7
[16:23:26] ✓ jroelofs@Escher bin$ ./clang -target x86_64-unknown-linux --gcc-toolchain=/Users/jroelofs/workdir/llvm/tools/clang/test/Driver/Inputs/multilib_64bit_linux_tree/usr/-print-multi-lib
.;@m64
32;@m32
[16:23:29] ✓ jroelofs@Escher bin$
```

Universal Driver

“Clang is inherently a cross compiler.... However, actually cross compiling in practice involves much more than just generating the right assembly”

– Daniel Dunbar [4]

Proposed Solution

Target Triple: <arch><sub>-<vendor>-<sys>-<abi>

--print-supported-archs

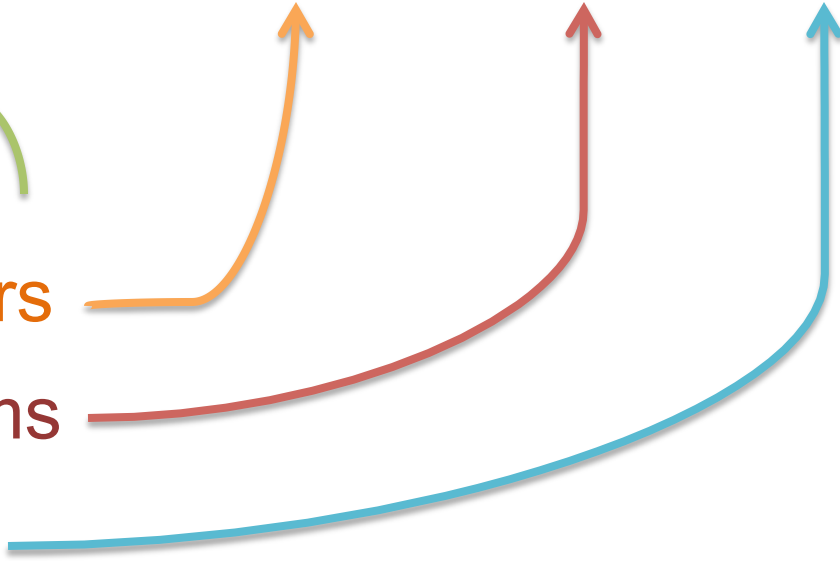
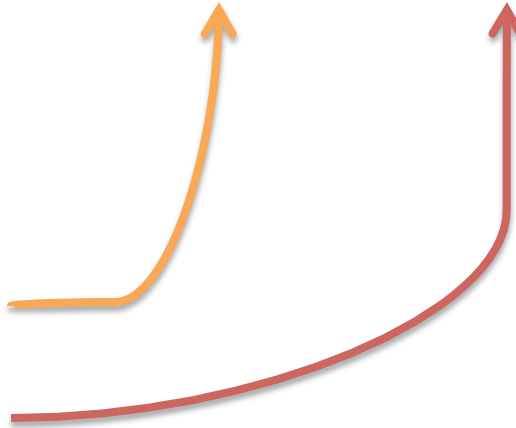
--print-supported-vendors

--print-supported-systems

--print-supported-abis

--print-multi-libs

--print-available-targets



Proposed Solution: Examples

```
$> clang --print-supported-archs
```

```
x86
```

```
...
```

```
$> clang -march x86 --print-supported-systems
```

```
auroraux
```

```
darwin
```

```
macosx
```

```
...
```

```
$> clang -march x86 --print-available-systems
```

```
linux
```

Proposed Solution: Examples

```
$> clang --print-supported-targets
```

```
x86-linux-gnu
```

```
ppc-apple-darwin
```

```
arm-none-eabi
```

```
$> clang --print-available-targets
```

```
x86-linux-gnu
```

```
$> clang -target ppc-apple-darwin foo.c
```

```
Sorry, but the toolchain for: ppc-apple-darwin  
has not been installed.
```

Conclusion

It should be simple to ask Clang which targets it could support, and of those, which ones it does support.

Thank you!

Backup Slides

Bibliography

- [1] <http://lists.cs.uiuc.edu/pipermail/cfe-dev/2014-March/036002.html>
- [2] <http://lists.cs.uiuc.edu/pipermail/cfe-dev/2010-December/012465.html>
- [3] <http://stackoverflow.com/questions/15036909/clang-how-to-list-supported-target-architectures/18576360#18576360>
- [4] <http://clang.llvm.org/UniversalDriver.html>