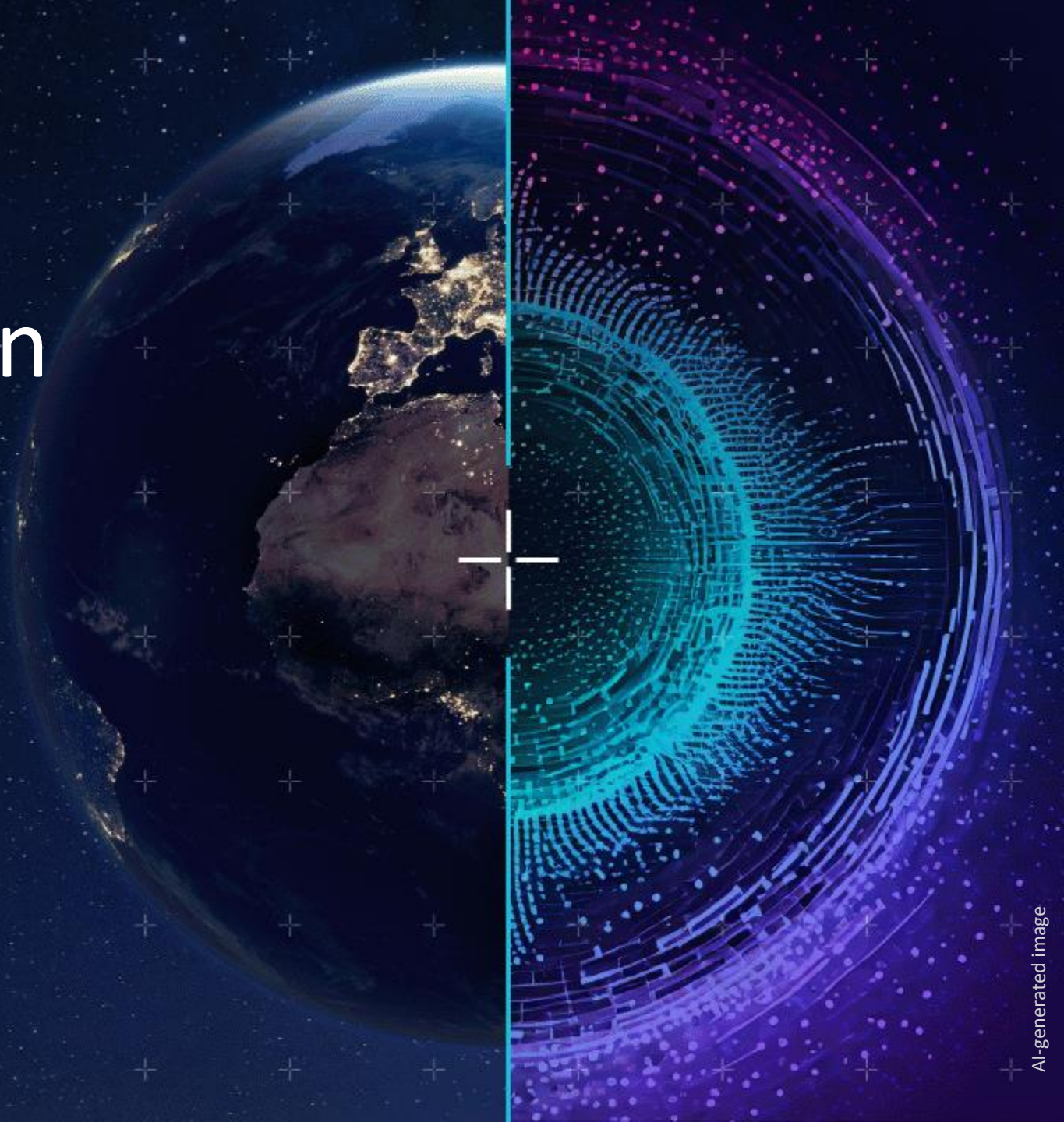
arm

Multilib Configuration Files



Peter Smith

April 2024

© 2024 Arm

AI-generated image

What is multilib?

- Compiler driver can support more than one target with the same installation.
- Pre-compiled libraries and header files differ per target.
- On Linux and similar platforms `/lib`, `/lib32` and `/lib64` directories for 32-bit and 64-bit programs.
- Embedded toolchains support many different CPU and ABI variants.
- Multilib is a mapping of command line options to include and library directories.

© 2024 Arm

Multilib example clang (-m elf_x86_64)

```
-dynamic-linker /lib64/ld-linux.so.2
/lib/x86_64-linux-gnu/crt1.o
/lib/x86_64-Linux-gnu/crti.o
/usr/lib/gcc/x86_64-linux-gnu/12/crtbeginS.o
-L/usr/lib/gcc/x86_64-linux-gnu/12
-L/usr/lib64
-L/lib/lib64
-L/usr/lib/llvm-14/lib
-L/lib
-L/usr/lib
/lib/x86_64-linux-gnu/12/crtendS.o
/lib/x86_64-Linux-gnu/crtn.o
```


© 2024 Arm

Multilib example clang -m32 (-m elf_i386)

```
-dynamic-linker /lib/ld-linux.so.2
/usr/lib32/Scrt1.o
/usr/lib32/crti.o
/usr/lib/gcc/x86_64-linux-gnu/11/32/crtbeginS.o
-L/usr/lib/gcc/x86_64-linux-gnu/11/32
-L/usr/lib32
-L/lib/lib32
-L/usr/lib/llvm-14/lib
-L/lib -L/usr/lib
/usr/lib/gcc/x86_64-linux-gnu/11/32/crtendS.o
/usr/lib32/crtn.o
```


© 2024 Arm

Multilib implementation by example

```
 // Find multilibs with subdirectories like armv7-a, thumb, armv7-a/thumb.
MultilibBuilder ArmV7Multilib = MultilibBuilder("/armv7-a")   // /armv7-a directory when
                                    .flag("-march=armv7-a") // -march=armv7a is present and –mthumb is absent
                                    .flag("-mthumb", /*Disallow=*/true);
MultilibBuilder ThumbMultilib = MultilibBuilder("/thumb")                       // /thumb directory when
                                    .flag("-march=armv7-a", /*Disallow=*/true) // -mthumb is present and
                                    .flag("-mthumb"); // require –mthumb       // -march=armv7-a is absent
MultilibBuilder ArmV7ThumbMultilib = MultilibBuilder("/armv7-a/thumb")          // /armv7-a/thumb dir when
                                    .flag("-march=armv7-a").flag("-mthumb");   // both –mthumb and armv7-a
  MultilibBuilder DefaultMultilib = MultilibBuilder("") // no directory when neither –mthumb nor –march=armv7-a
                                    .flag("-march=armv7-a", /*Disallow=*/true)
                                    .flag("-mthumb", /*Disallow=*/true);
  MultilibSet AndroidArmMultilibs = // Set containing all 4 possible combinations
      MultilibSetBuilder()
          .Either(ThumbMultilib, ArmV7Multilib, ArmV7ThumbMultilib,
                  DefaultMultilib)
          .makeMultilibSet()
          .FilterOut(NonExistent);
```


© 2024 Arm

Use case for configuration file based multilib

- An embedded toolchain can have hundreds of multilibs
  - Target architecture (v6-m, v7-m, v8-m, v8.1-m.main, … AArch64)
  - Little or big endian
  - Floating point calling convention
  - Exceptions/no-exceptions
  - Position-independent or not
- Many multilib instances will satisfy a set of command-line options
  - Want to select the most optimal.
- Many possible embedded toolchains
  - Different targets (Arm, RISCV etc)
  - Different C-libraries (picolibc, newlib)
- Not practical for every toolchain to hard-code their multilibs upstream.

© 2024 Arm

Configuration file based multilib

- Design at https://github.com/llvm/llvm-project/blob/main/clang/docs/Multilib.rst
- Defer the creation of Multilib and MultilibSet until run-time
  1. Normalize command line options into **Flags** (-mcpu, -march are normalized to --target)
  2. Load multilib.yaml from sysroot
  3. Generate additional **Flags** from multilib **Mappings** section.
  4. Match **flags** against multilib variants to select one or more variants.
  5. Generate -isystem and –L options from selected multilib variants.
- Normalization is done in driver with a limited set of options.
- In use with LLVM embedded Toolchain for Arm

© 2024 Arm

Example multilib.yaml config file

```
Variants:
-Dir: thumb/v6-m
  Flags: [--target=thumbv6m-none-unknown-eabi]
-Dir: thumb/v7-m
  Flags: [--target=thumbv6m-none-unknown-eabi, -mfpu=fpv4-sp-d16]
Mappings:
-Match: --target=thumbv([7-9]|[1-9][0-9]+).*
  Flags: [--target=thumbv7m-none-eabi]
```


© 2024 Arm

Experience and future plans

- Some Multilib variants do not map to any command-line option
  - Can work around by changing sysroot
  - Plan for a `--multilib=<string>` where `<string>` is described in multilib.yaml
- Dependencies between C++, resource and C library directories
  - #include_next requires C++ then resource then C on include path.
  - Multilib with multiple variants expands to
    - Variant1 C++
    - …
    - VariantN C++
    - Variant1 resource
    - …
    - VariantN resource
    - … for C library
  - Currently using exclusive match

© 2024 Arm

Conclusion

- Multilib configuration can be achieved at run-time using configuration files.
- In active use in Arm's LLVM based toolchains.
- More work to do to handle more complex cases.
- Seeking feedback from other toolchain implementations.

© 2024 Arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה
ధన్యవాదములు

© 2024 Arm

The Arm trademarks featured in this presentation are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

www.arm.com/company/policies/trademarks

© 2024 Arm