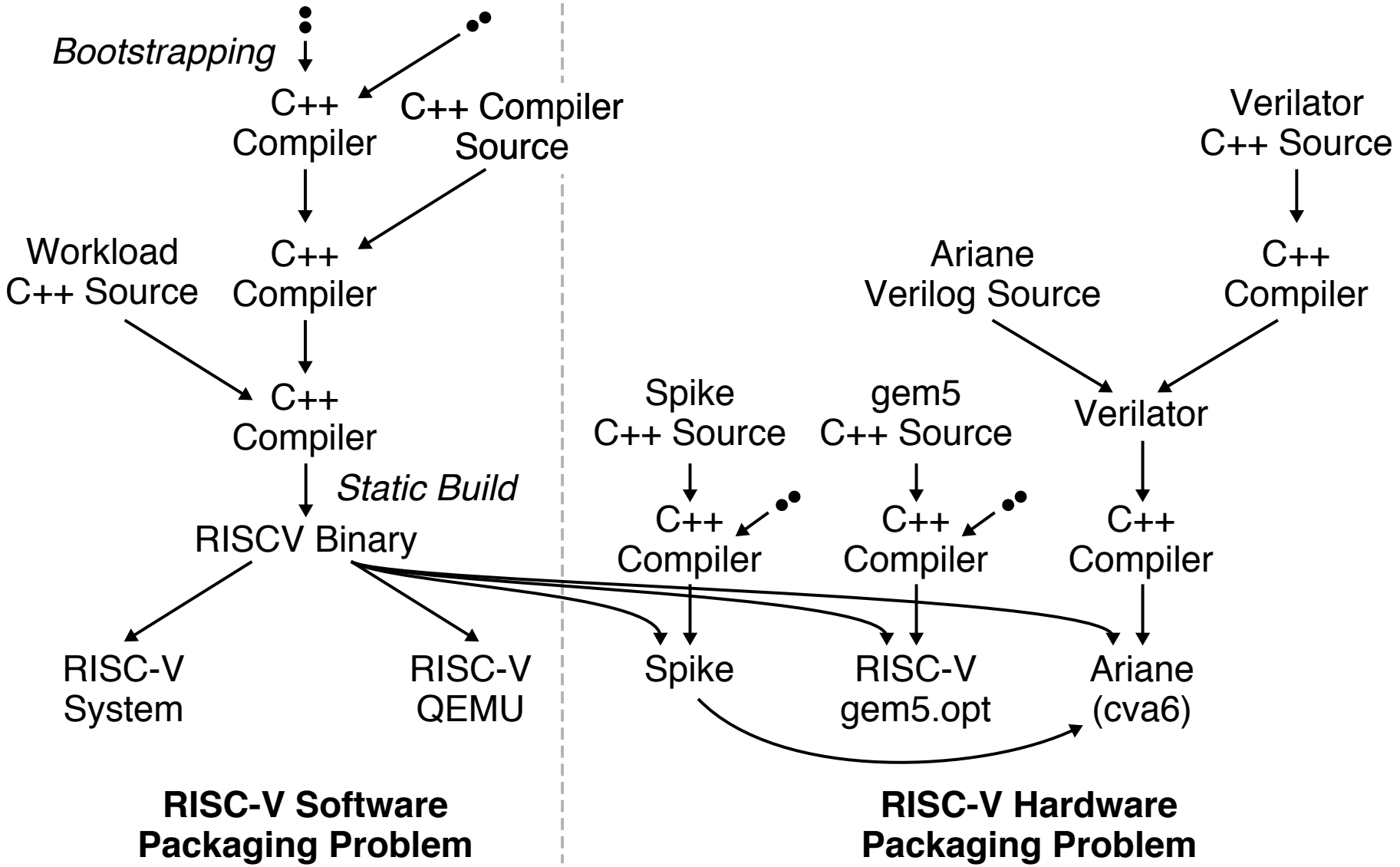
The Case for Using Guix to Enable Reproducible RISC-V Software & Hardware

Christopher Batten[1], Pjotr Prins[2]
Efraim Flashner[2], Arun Isaac[2], Jan Nieuwenhuizen[3]
Ekaitz Zarraga[4], Tuan Ta[1], Austin Rovinski[1], Erik Garrison[2]

[1] School of Electrical and Computer Engineering, Cornell University
[2] The University of Tennessee Health Science Center
[3] Joy of Source [4] ElenQ Technology

The RISC-V Packaging Problem



An Ideal RISC-V Packaging Solution?

- **Transparent –** understand entire development environment including exact build configurations and version of every dependency
- **Lightweight –** integrate into standard development environment
- **Flexible –** easily switch between different development environments
- **Isolated –** isolate entire development environment to prevent accidently “leaking” user’s environment into an experiment
- **Portable –** build workloads for native execution and/or target multiple ISAs for cycle-level simulation
- **Fast –** leverage precompiled packages when available
- **Distribution Agnostic –** enable researchers to use any distribution
- **Extensible –** extensions through a general-purpose language

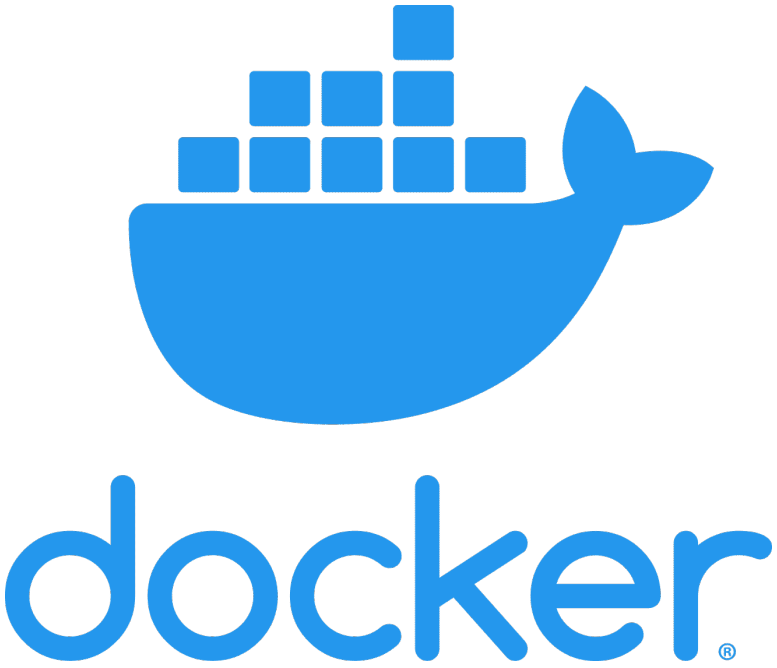
Why not just use Docker?

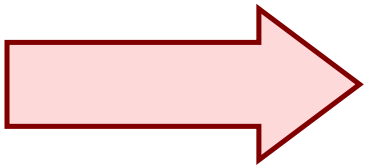
```
FROM ubuntu:20.04
ENV DEBIAN_FRONTEND=noninteractive
RUN apt -y update
RUN apt -y upgrade
RUN apt -y install build-essential git m4 scons zlib1g zlib1g-dev
                   libprotobuf-dev protobuf-compiler libprotoc-dev ...
RUN pip install mypy
```

```
% git clone https://gem5.googlesource.com/public/gem5
% docker pull gcr.io/gem5-test/ubuntu-20.04_all-dependencies:v21-2
% docker images
REPOSITORY                                          SIZE
gcr.io/gem5-test/ubuntu-20.04_all-dependencies  1.38GB
% docker run -u $UID:$GID --volume /home/cb535/gem5:/gem5
    --rm -it gcr.io/gem5-test/ubuntu-20.04_all-dependencies:v21-2
I have no name!@bbfd8a86240b:/$
```

- Transparent?
- Lightweight?
- Flexible?
- Isolated?
- Portable?
- Fast?
- Distribution Agnostic?
- Extensible?

Containers are Like Smoothies





A smoothie tastes great ...
but how much do we know about what is really in the smoothie?
Can someone else make the exact same smoothie?

Adapted from L Courtès, FOSDEM'20

&

Guix

Talk Outline

Motivation

Guix Background

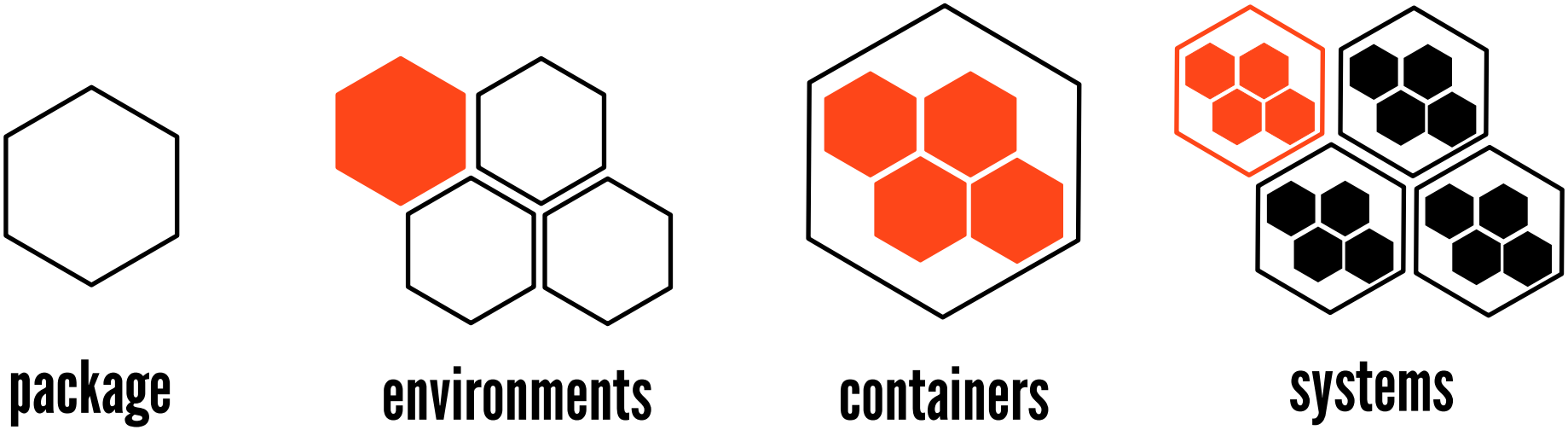
Guix for RISC-V Software

Guix for RISC-V Hardware

Guix for RISC-V Demo

What is Guix?

General toolbox for software deployment



- Guix is a functional, transactional package manager
- Guix is an environment manager
- Guix is a reproducible container generator
- Guix is a complete operating system constructor and manager

Adapted from L Courtès, FOSDEM'20

Guix Hello World

```
% guix pull
% guix install hello
% guix package --list-installed
hello 2.12 /gnu/store/x2byq2a04pi...1mqikz07i1m-hello-2.12

% which hello
~/.guix-profile/bin/hello

% readlnk $(which hello)
/gnu/store/x2byq2a04pi...1mqikz07i1m-hello-2.12/bin/hello

% hello
Hello, world!

% guix remove hello
```

Guix is more than a package manager!

The Guix `hello` Package

```
(define-public hello
(package
  (name "hello")
  (version "2.12.1")
  (source (origin
            (method url-fetch)
            (uri (string-append "mirror://gnu/hello/hello-" version
                                ".tar.gz"))
            (sha256
             (base32
              "086vqwk2wl8zfs47sq2xpjc9k066ilmb8z6dn0q6ymwjzlm196cd"))))
  (build-system gnu-build-system)
  (synopsis "Hello, GNU world: An example GNU package")
  (description
   "GNU Hello prints the message \"Hello, world!\" and then exits.  It
serves as an example of standard GNU coding practices.  As such, it
supports command-line arguments, multiple languages, and so on.")
  (home-page "https://www.gnu.org/software/hello/")
  (license gpl3+)))
```

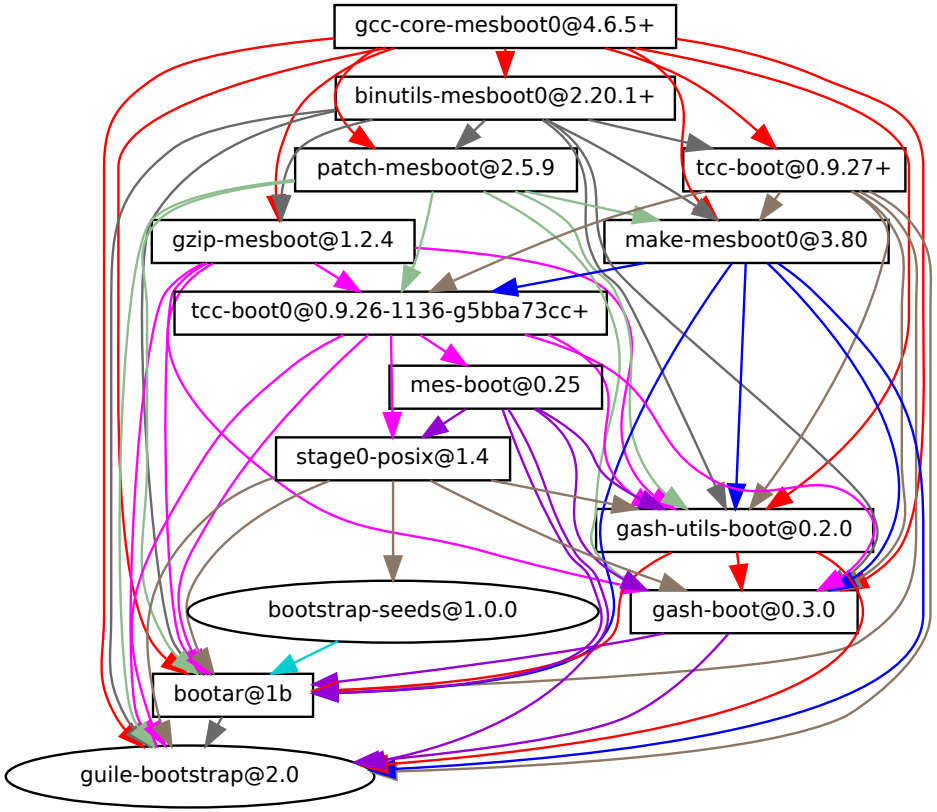
&

Guix

Talk Outline

Bootstrapping RISC-V for Guix

- Stage0-POSIX
  - Hex0: Raw ELF file in hex
  - Hex1: Hex0, one char labels
  - Hex2: Hex1, proper labels, etc
  - M0/M1: simple macro system
  - M2-Planet: Simple C subset that uses M0 as output
- GNU Mes
  - Mutually hosted Scheme interpreter in simple C subset and C compiler in Scheme



- TinyCC
  - Simple C compiler, assembler, linker that can be compiled with MesCC
  - Able to compile older versions of gcc

Hex0 for RISC-V

https://github.com/oriansj/bootstrap-seeds/blob/master/POSIX/riscv64/hex0_riscv64.hex0

392B binary seed from fully transparent assembly source

```
# :_start ; (0x0600078)
  13 0A 00 00    # RD_S4 MV                ; Initialize register
  83 35 01 01    # RD_A1 RS1_SP !16 LD     ; Input file name

  ; Open input file and store FD in s2
  93 08 80 03    # RD_A7 !56 ADDI          ; sys_openat
  13 05 C0 F9    # RD_A0 !-100 ADDI        ; AT_FDCWD
  13 06 00 00    # RD_A2 MV                ; read only
  73 00 00 00    # ECALL
  13 09 05 00    # RD_S2 RS1_A0 MV         ; Save fd in for later

  ; Open output file and store the FD in s3
  13 05 C0 F9    # RD_A0 !-100 ADDI        ; AT_FDCWD
  83 35 81 01    # RD_A1 RS1_SP !24 LD     ; Output file (argument 3)
  13 06 10 24    # RD_A2 !577 ADDI         ; octal 00001101
  93 06 00 1C    # RD_A3 !448 ADDI         ; Set read, write, execute permission on user
  73 00 00 00    # ECALL
  93 09 05 00    # RD_S3 RS1_A0 MV         ; Save fd in for later
```

The Guix `smithwaterman` Package

```
(define-public smithwaterman-static
  (package
    (inherit smithwaterman)
    (name "smithwaterman-static")
    (arguments
     (substitute-keyword-arguments
       (package-arguments smithwaterman)
         ((#:make-flags flags ''())
          #~(cons "CFLAGS=-static" #$flags))))))

% guix build --target=riscv64-linux-gnu \
    smithwaterman-static
/gnu/store/fmdn4a1aa0z61dycd2956c9nbzahs2ac
            -smithwaterman-static-0.0.0-2.2610e25
```

Status Update on Porting Guix Packages to RISC-V

- Many application-level packages work out of the box
- `gcc-7` and up, `llvm-11` and up, `clang`
- `perl`
- `python2`, `python3`
- `ruby2.7`
  WIP: `ruby 2.5/2.6/3.0/3.1`
- `gccgo`
  WIP: `bootstrapping go-1.16/1.17`
- WIP: `rust`, `java`, `julia`, `ghc`

```
% guix install qemu
% DIR=$(guix build
   --target=riscv64-linux-gnu nano)
% echo $DIR
/gnu/store/1bzp...9j4f-nano-6.2
% ln -sf $DIR/bin/nano nano
% qemu-riscv64 ./nano

% DIR=$(guix build
    --target=riscv64-linux-gnu gzip)
% echo $DIR
/gnu/store/d8pm...675r-gzip-1.10
% ln -sf $DIR/bin/gipz gzip
% echo "CARRV 2022" > carrv.txt
% qemu-riscv64 ./gzip -kf carrv.txt
% gunzip -c carrv.txt.gz
```

&

Guix

Talk Outline

Motivation

Guix Background

Guix for RISC-V Software

Guix for RISC-V Hardware

Guix for RISC-V Demo

The Guix `spike` Package

`https://git.savannah.gnu.org/cgit/guix.git/tree/gnu/packages/virtualization.scm#n1012`

- Fetch specific version using git tag
- Patch `riscv/dts.cc` to use Guix package for `dtc`
- Leverage Guix built-in support for GNU build systems
- Captures all dependencies (`dtc`, `python-wrapper` for testing)
- Package is upstreamed enabling fast binary substitutions

The Guix `gem5` Package

`https://git.genenetwork.org/guix-bioinformatics/guix-bioinformatics/src/branch/master/gn/packages/virtualization.scm#L21`

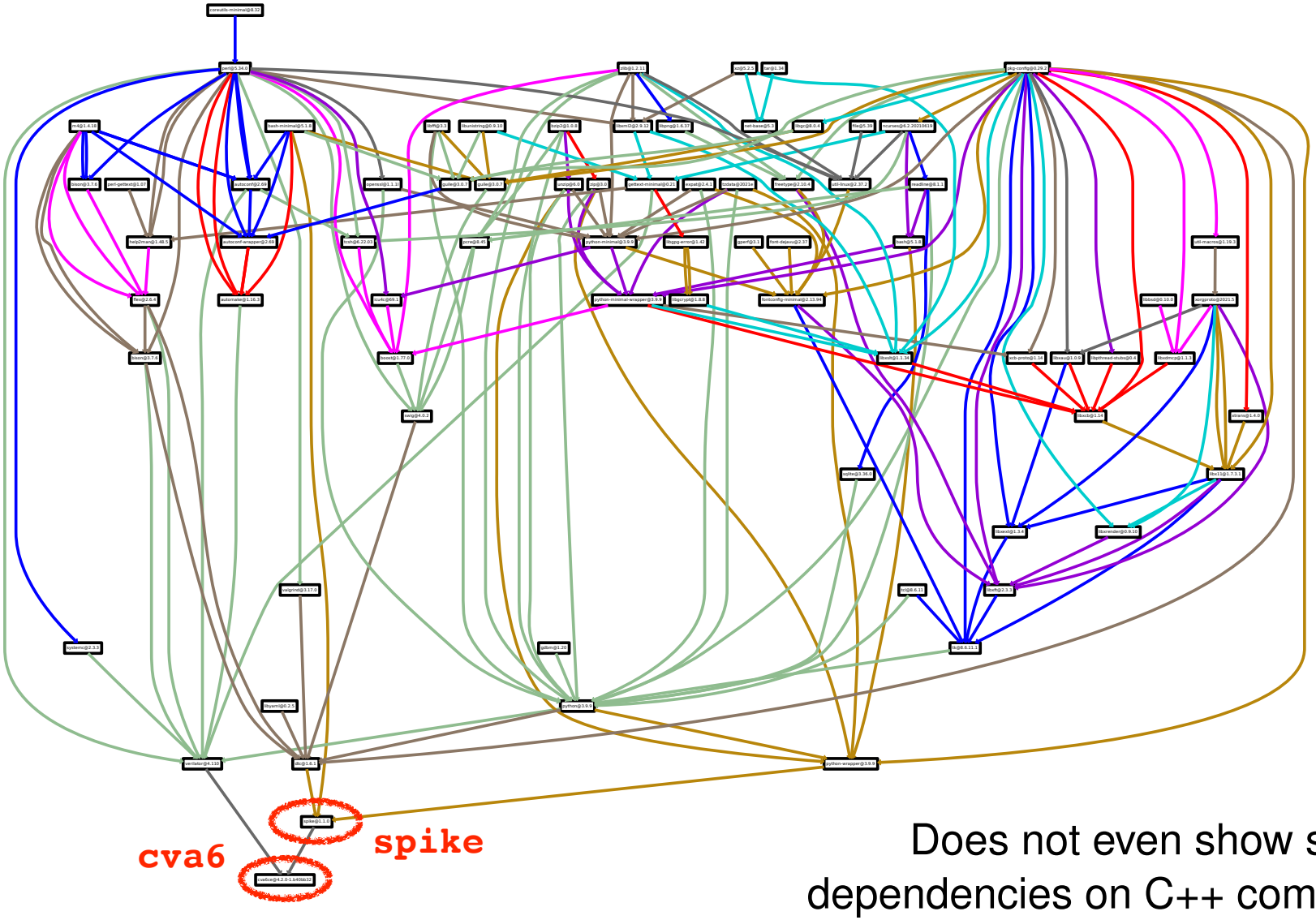
- Fetch specific version using git tag
- Eliminate non-deterministic use of `__DATE__` and `__TIME__`
- Patch `Makefile/SConstruct` to use Guix packages such as `pybind11`, `zlib`, `libpng`
- Leverage Guix built-in support for SCons build systems
- Builds for multiple architectures (e.g., x86, ARM, RISC-V)
- Installs binaries for each simulator suffixed with architecture
- Installs default configurations
- Captures all dependencies
- Provides a derived package to install a single architecture

The Guix `cva6` (Ariane) Package

`https://git.genenetwork.org/guix-bioinformatics/guix-bioinformatics/src/branch/master/gn/packages/riscv.scm#L41`

- Fetch specific version using git commit
- Patch `Makefile` to eliminate ad-hoc environment variable used to specify the location of `libfesvr.a`
- Leverage Guix built-in support for GNU build systems
- Patch Ariane Verilog to print to stdout correctly
- Eliminates dependency on pre-compiled RISC-V toolchain binary blob
- Captures dependency on Guix `spike` package for `libfesvr.a`
- Captures dependency on Guix `verilator-4.110` package since Ariane cannot use most recent version of Verilator
- Install binary named `ariane`

The Guix `cva6` Package Dependency Graph



Does not even show some dependencies on C++ compiler!

The Guix `cva6` Package Hash

```
% guix install cva6
% readlink $(which ariane)
/gnu/store/j9awlgnksin6shkjh691bpmb3miq9sm0-cva6-4.2.0-1.b40bb3
  /bin/ariane
```

This hash in the `/gnu/store` captures:

- all direct dependencies (e.g., spike, verilator)
- all implicit dependencies (e.g., C++, Autotools, etc)
- all recursive dependencies (e.g., Python, dtc, valgrind, flex, bison, etc)
- even the compiler used to build the compiler!
- every command line option and environment variable

&

Guix

Talk Outline

Motivation

Guix Background

Guix for RISC-V Software

Guix for RISC-V Hardware

Guix for RISC-V Demo

Guix for RISC-V Demo

```
% guix install smithwaterman
% smithwaterman -p TGATTGTACCAAA TGATCATGTACCA

% guix install qemu
% guix install spike
% guix install gem5-riscv
% guix install cva6

% DIR=$(guix build \
        --target=riscv64-linux-gnu smithwaterman-static)
% ln -sf $DIR/bin/smithwaterman sw

% DIR=$(guix build \
        --target=riscv64-linux-gnu riscv-pk
% ln -sf $DIR/bin/pk pk
```

Guix for RISC-V Demo

```
% qemu-riscv64 ./sw -p TGATTGTACCAAA TGATCATGTACCA

% spike ./pk ./sw -p TGATTGTACCAAA TGATCATGTACCA

% gem5.opt \
    $GUIX_PROFILE/share/gem5/configs/example/se.py \
    --cmd=./sw \
    --options="-p TGATTGTACCAAA TGATCATGTACCA"

% ariane +max-cycles=100000000 +time_out=100000000 \
    ./pk ./sw -p TGATTGTACCAAA TGATCATGTACCA
```

&

Guix

Take-Away Points

- Packging RISC-V software and hardware can be challenging
- Guix is a mature toolbox for software deployment including support for packages, environments, containers, and systems
- Guix can potentially offer a compelling option for packaging in the RISC-V ecosystem