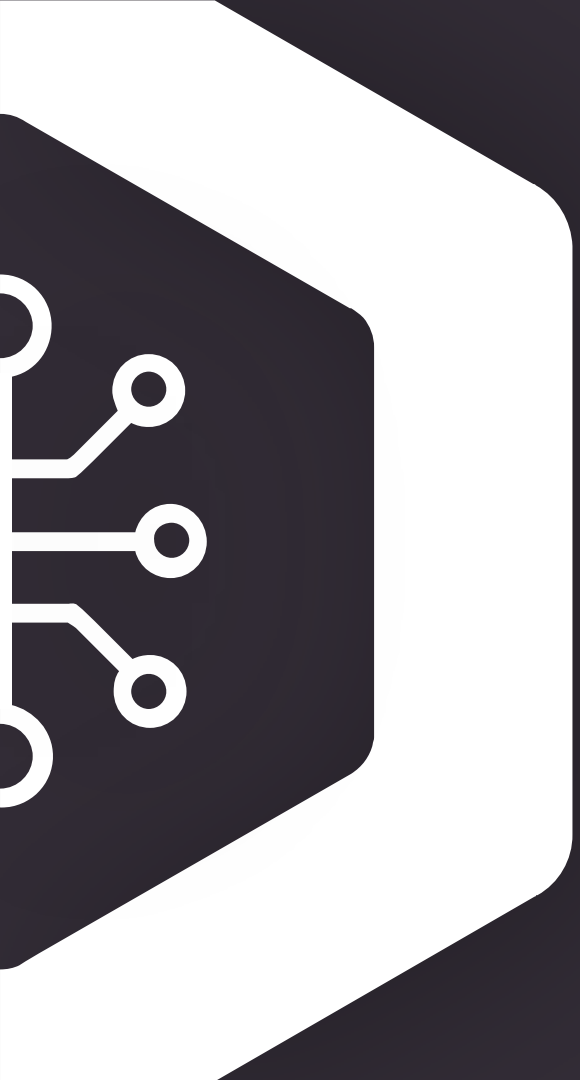


Practical Adoption of Open Source System Verilog Tools

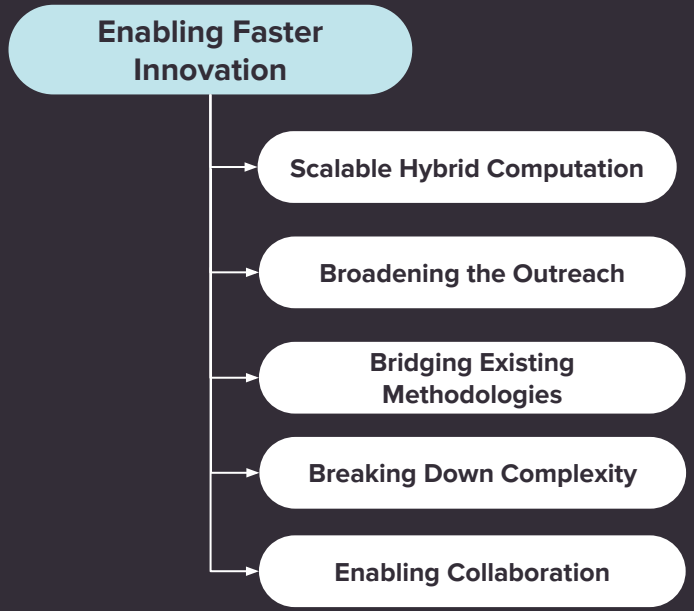
CHIPS Alliance Fall Workshop, 2021-10-12

Michael Gielda, mgielda@antmicro.com

antmicro

antmicro


CHIPS
ALLIANCE


Enabling Faster Innovation
Scalable Hybrid Computation
Broadening the Outreach
Bridging Existing Methodologies
Breaking Down Complexity
Enabling Collaboration

CHIPS ALLIANCE

antmicro

antmicro

Enabling Faster Innovation

- Scalable Hybrid Computation
- Broadening the Outreach
- Bridging Existing Methodologies
- Breaking Down Complexity
- Enabling Collaboration

WHY DO WE NEED SYSTEMVERILOG SUPPORT IN OPEN SOURCE TOOLS?

- Proprietary licensing of existing tools makes it hard to build scalable, reproducible CIs
  - Especially publicly accessible CIs in multi-org projects - OpenTitan, CHIPS Alliance
- Number of open source cores and lots of pre-existing IP implemented in SystemVerilog, e.g.
  - SweRV
  - Ibex
  - BlackParrot
  - Core-V
- Open source tools will help building a collaborative ecosystem around ASIC and FPGA design

HOW TO BUILD AN OPEN SOURCE SYSTEMVERILOG ECOSYSTEM

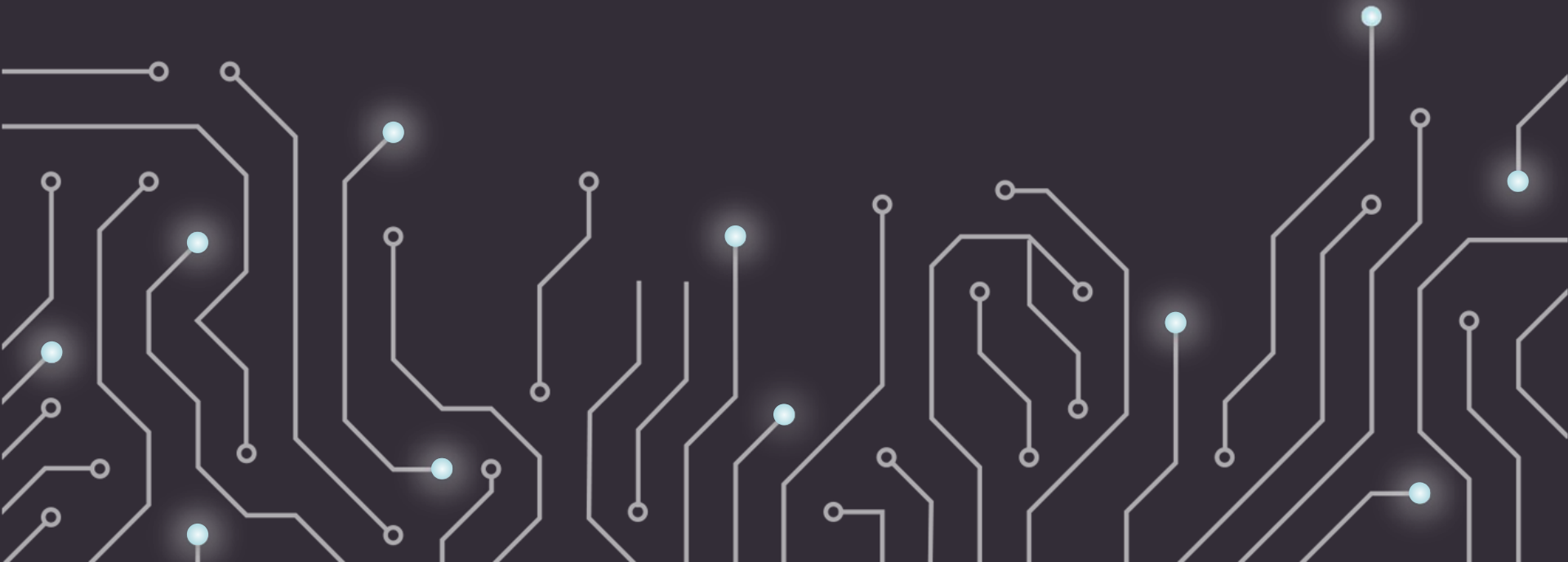
- Identify missing functionalities and features
- Reuse existing solutions
  - There are many existing projects which can be improved
- Create well documented and transparent projects
  - Include automated tests and status reporting in projects
- Cooperate with others
  - Gather information on what is needed
- **Provide incremental value**

CHIPS ALLIANCE & HDLS

- CHIPS wants to encourage all sorts of open source ASIC development - and acknowledges that the future ecosystem will be pluralistic
- workgroups for both Chisel and SystemVerilog peacefully coexist and collaborate within our Technical Steering Committee
- See https://chipsalliance.org/workgroups/ for list of workgroups, https://github.com/chipsalliance/tsc for current TSC composition

antmicro

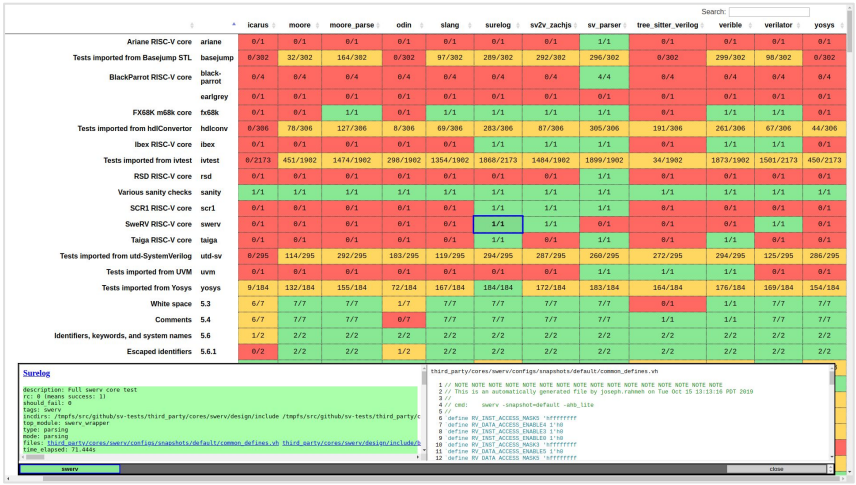
SO, WHAT DO WE MEAN BY PRACTICAL ADOPTION?



antmicro

LAY OF THE LAND: SV-TESTS

- Test suite to determine the SystemVerilog support level in various open source tools
- Pinpoint all the supported/missing SV features
- Report: symbiflow.github.io/sv-tests
- Three types of tests:
  - SystemVerilog features
  - Existing third party test suites
  - Selected open source IP cores (SweRV, Ibex etc.)



Search:

		icarus	moore	moore_parse	odin	slang	surelog	sv2v_zachjs	sv_parser	tree_sitter_verilog	verible	verilator	yosys
Ariane RISC-V core	ariane	0/1	0/1	0/1	0/1	0/1	0/1	0/1	1/1	0/1	0/1	0/1	0/1
Tests imported from Basejump STL	basejump	0/302	32/302	164/302	0/302	97/302	289/302	292/302	296/302	0/302	299/302	98/302	0/302
BlackParrot RISC-V core	black-parrot	0/4	0/4	0/4	0/4	0/4	0/4	0/4	4/4	0/4	0/4	0/4	0/4
	earlgrey	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	0/1
FX68K m68k core	fx68k	0/1	0/1	1/1	0/1	1/1	1/1	1/1	1/1	0/1	1/1	1/1	0/1
Tests imported from hdlConvertor	hdlconv	0/306	78/306	127/306	0/306	69/306	283/306	87/306	305/306	191/306	261/306	67/306	44/306
Ibex RISC-V core	ibex	0/1	0/1	0/1	0/1	0/1	1/1	1/1	1/1	0/1	1/1	1/1	0/1
Tests imported from ivtest	ivtest	0/2173	451/1902	1474/1902	298/1902	1354/1902	1868/2173	1484/1902	1899/1902	34/1902	1873/1902	1501/2173	450/2173
RSD RISC-V core	rsd	0/1	0/1	0/1	0/1	0/1	0/1	0/1	1/1	0/1	0/1	0/1	0/1
Various sanity checks	sanity	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1	1/1
SCR1 RISC-V core	scr1	0/1	0/1	0/1	0/1	0/1	1/1	1/1	1/1	0/1	0/1	0/1	0/1
SweRV RISC-V core	swerv	0/1	0/1	0/1	0/1	0/1	1/1	1/1	0/1	0/1	0/1	1/1	0/1
Taiga RISC-V core	taiga	0/1	0/1	0/1	0/1	0/1	1/1	0/1	1/1	0/1	1/1	0/1	0/1
Tests imported from utd-SystemVerilog	utd-sv	0/295	114/295	292/295	103/295	119/295	294/295	287/295	260/295	272/295	294/295	125/295	286/295
Tests imported from UVM	uvm	0/1	0/1	0/1	0/1	0/1	0/1	0/1	1/1	1/1	1/1	0/1	0/1
Tests imported from Yosys	yosys	9/184	132/184	155/184	72/184	167/184	184/184	172/184	183/184	164/184	176/184	169/184	154/184
White space	5.3	6/7	7/7	7/7	1/7	7/7	7/7	7/7	7/7	0/1	1/1	7/7	7/7
Comments	5.4	6/7	7/7	7/7	0/7	7/7	7/7	7/7	7/7	1/1	1/1	7/7	7/7
Identifiers, keywords, and system names	5.6	1/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2
Escaped identifiers	5.6.1	0/2	2/2	2/2	1/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2

antmicro Google

REUSE:
SURELOG / UHDM

- How to provide good SV support across the board?
- Surelog: open source SystemVerilog 2017 Pre-processor, Parser, Elaborator and UHDM Compiler
- Universal Hardware Data Model (UHDM) is used to exchange the information about elaborated SV design between the parser and other tools
- Integrating UHDM into Verilator and Yosys
- Can parse, synthesize and simulate OpenTitan's Ibex core directly from source - https://github.com/chipsalliance/UHDM-integration-tests

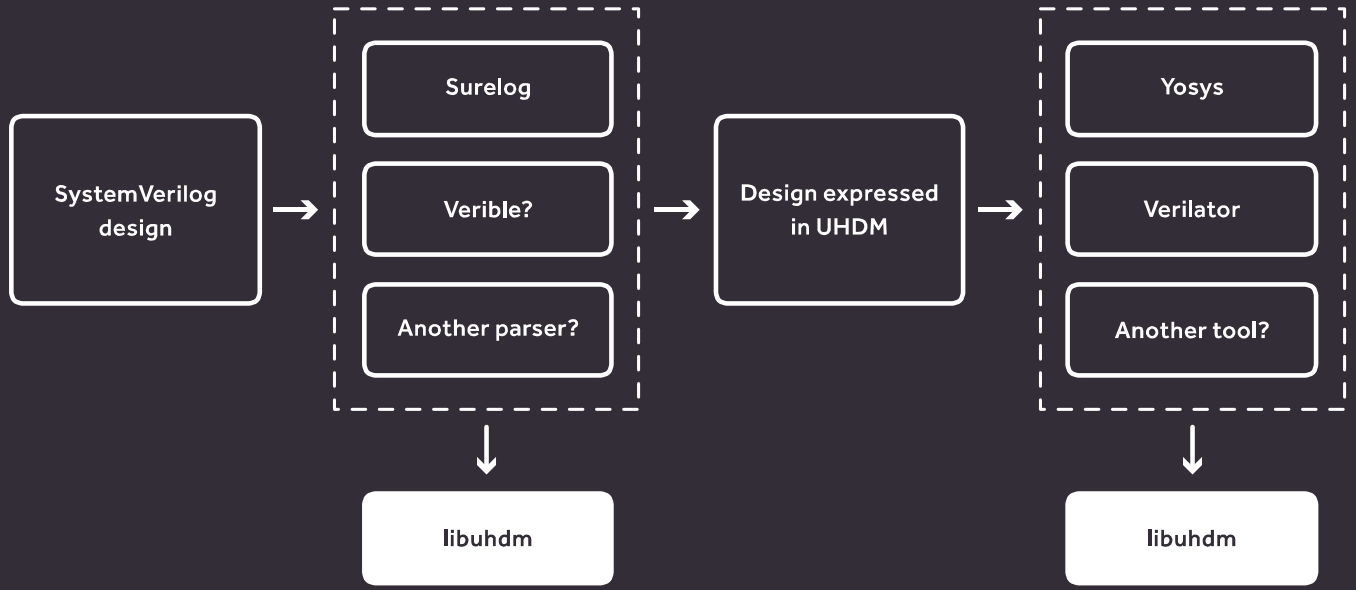
Google

lowRISC

opentitan

antmicro

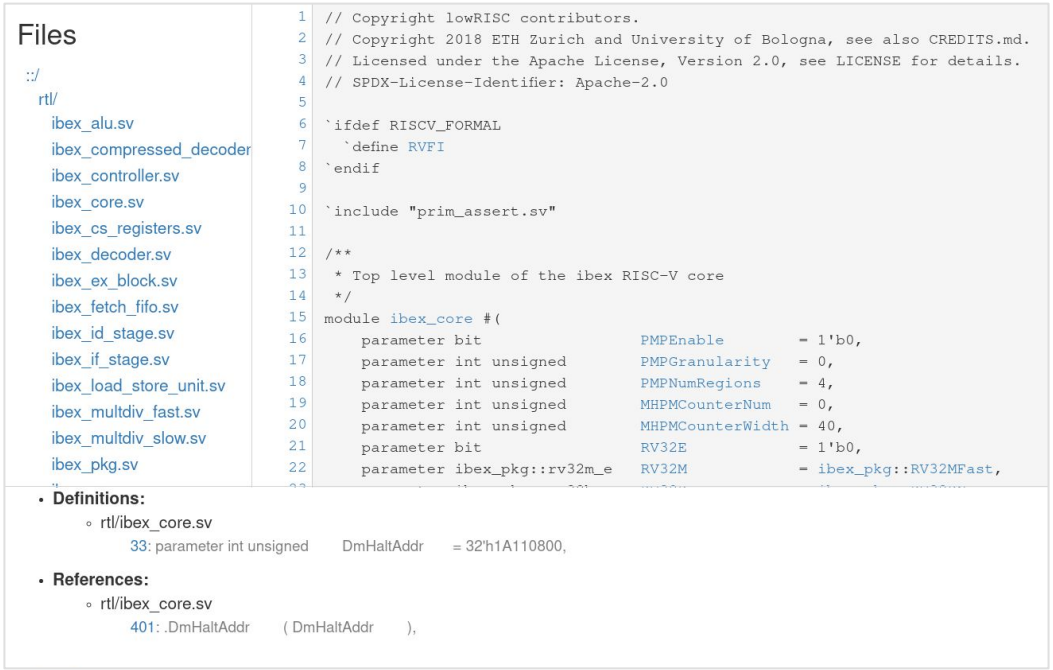
UHDM/SURELOG + VERILATOR/YOSYS



antmicro

INCREMENTAL VALUE: VERIBLE

- Open source SV linter/formatter
- Released by Google on GitHub, onboarded into CHIPS
- Actively developed by Google and Antmicro
- Used e.g. in Ibex (OpenTitan) CI
- Very versatile, with lots of practical uses



antmicro

VERIBLE - LINTING

- Linter — a static code analysis tool to spot and fix stylistic errors and bugs in SystemVerilog code
- Allows enforcement of rules on a project or company level in various SystemVerilog projects to follow authoritative style guides
- The rules vary from simple ones to more sophisticated and are highly configurable
- E.g. making sure the module name matches the file name, checking variable naming conventions

```
$ verible-verilog-lint --ruleset all core.sv
core.sv:3:11: Interface names must use lower_snake_case
naming convention and end with _if. [Style: interface-
conventions] [interface-name-style]
```

antmicro

VERIBLE - FORMATTING

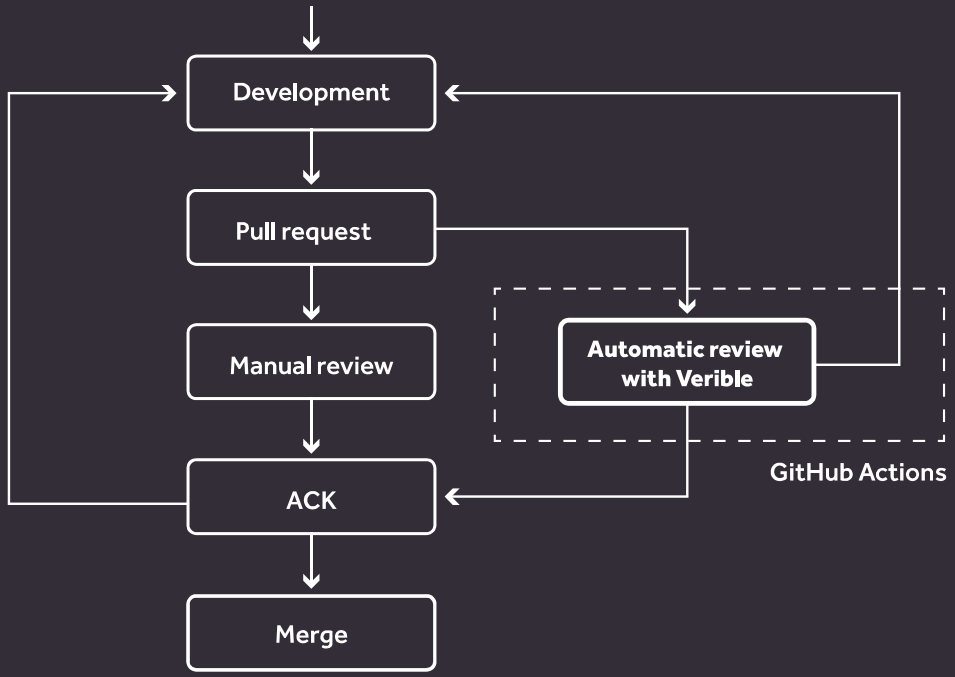
- Formatter — a complementary tool for the linter used to automatically detect various formatting issues like improper indentation or alignment
- As opposed to the linter, it only detects and fixes issues that have no lexical impact on the source code
- Linter + formatter can effectively remove all the discussions about styling, preferences and conventions from all pull requests
- Developers can then focus solely on the technical aspects of the proposed changes

```
$ cat sample.sv
typedef struct {
bit first;
        bit second;
bit
   third
        ;
  bit fourth;
bit fifth; bit sixth;
}
 foo_t;
```

```
$ verible-verilog-format sample.sv
typedef struct {
  bit first;
  bit second;
  bit third;
  bit fourth;
  bit fifth;
  bit sixth;
} foo_t;
```

antmicro

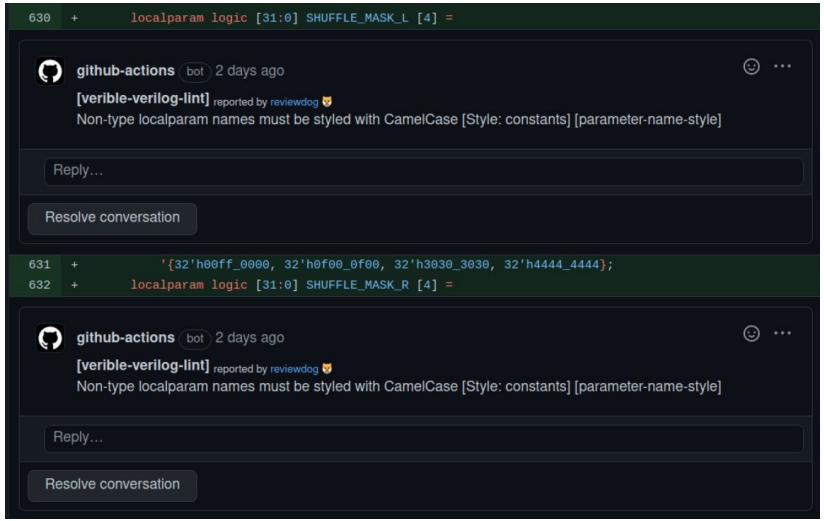
VERIBLE GITHUB ACTIONS



antmicro

VERIBLE LINTER GITHUB ACTION

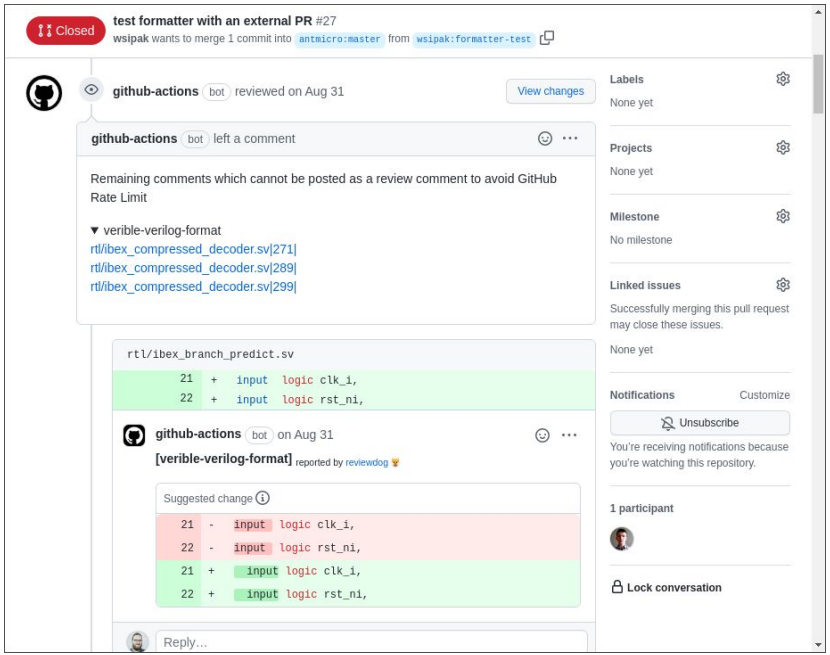
- chipsalliance/verible-linter-action
- Any GitHub-hosted open-source or private project can inform commiters about the issues detected in their code
- Wide range of applications can be devised, from faster pull requests reviews, to isolating erroneous portions of code
- Blog note



antmicro

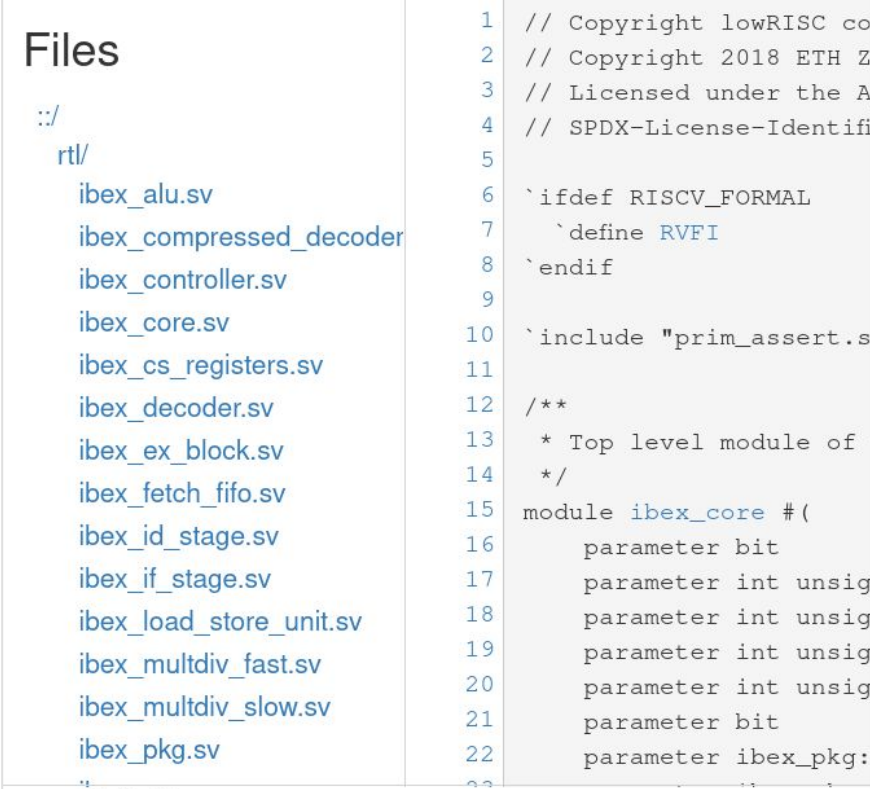
VERIBLE FORMATTER GITHUB ACTION

- chipsalliance/verible-formatter-action
- Same as above, except it automatically generates code suggestions which can just be accepted - instant benefit!



USING KYTHE FOR GENERATING INDEXED CODE DATABASE

- Verible can be used to perform many other tasks, e.g. generating a Kythe compatible indexing database
- Indexing an SV project simplifies collaboration - navigate through the source code using a web browser
- Showcase GitHub repository
- Example index webpage



COOPERATE WITH OTHERS: ADOPTION

- Our main focus in ongoing development was the OpenTitan project
- Currently pushing wider adoption at Google and other OT partners
- Now also used by some of the Core-V users, such as QuickLogic
- Starting collaboration with ZeroASIC
- Also working with the wider Ibex, BlackParrot, Core-V and SWeRV communities

Google

zero

QuickLogic

opentitan

REUSE / COLLABORATE: UVM IN OPEN SOURCE

- Lots of pre-existing IP and test benches implemented in SystemVerilog
- Lots of developers familiar with UVM
- Open source UVM is necessary long term to combine commercial ecosystem with open source tools and methodologies
- Chip-making companies can benefit from open source while keeping their existing UVM codebase
- Open source tools will enable infinitely scalable, reproducible CIs

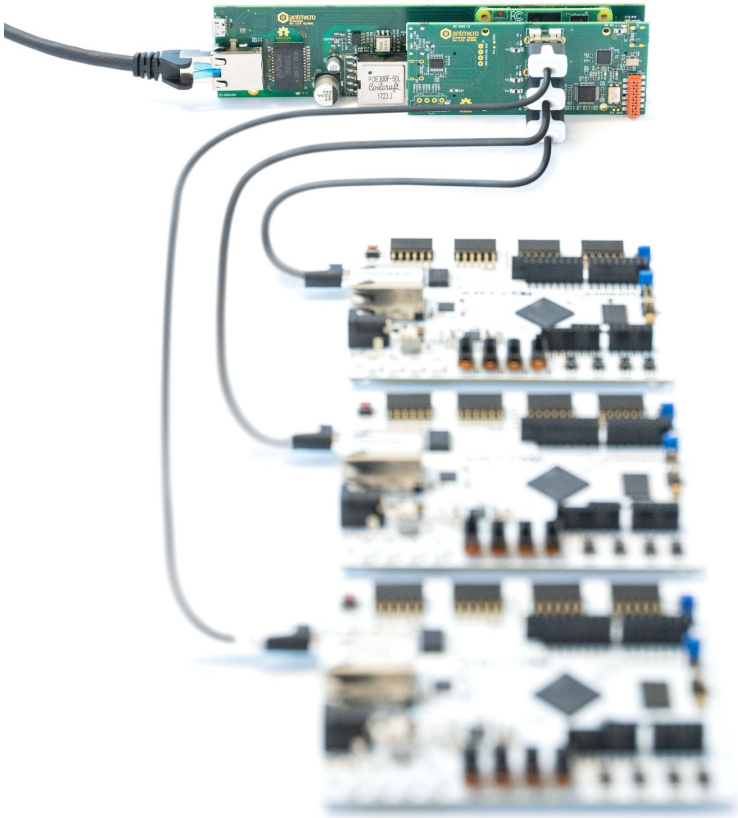
OPEN SOURCE UVM

- Ongoing work on extending Verilator with SystemVerilog features required by UVM:
  - Stratified scheduler
  - Randomize methods
  - Class support
- One of already reached milestones is development of [dynamic scheduling in Verilator]

antmicro

SCALE: GITHUB CUSTOM RUNNERS FOR CLOUD-ASSISTED CI

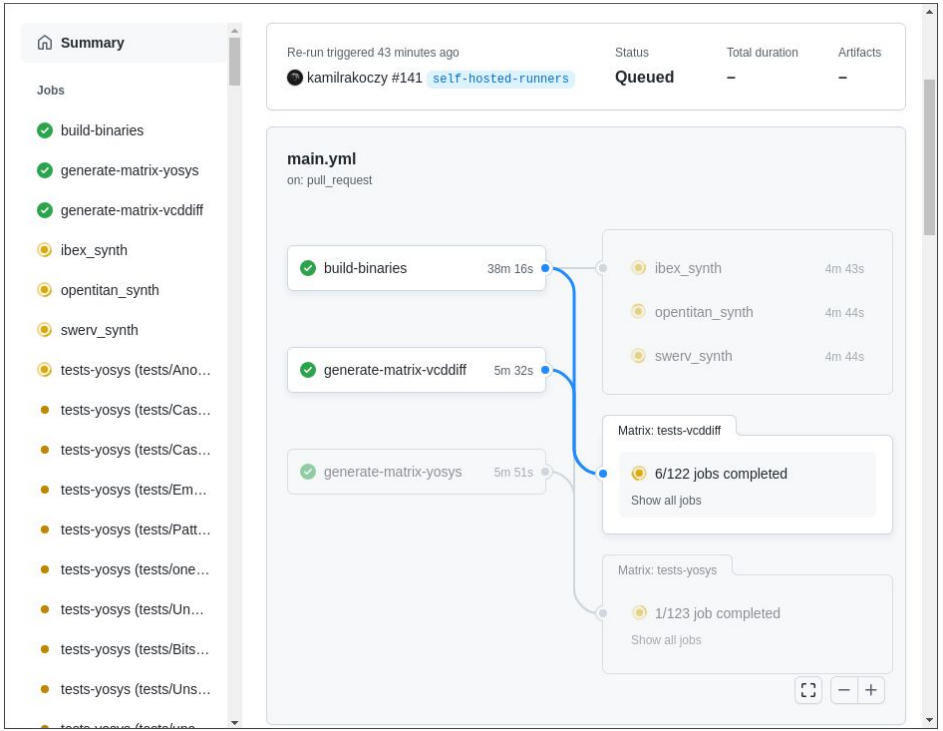
- Antmicro helps customers scale up ASIC development between teams and companies and into the cloud
- All the tools described here are meant to be used in a cloud context, on GitHub as well as private enterprise installations
- Mix and match open and closed components
- Use custom, powerful runners with scalable compute and custom peripherals e.g. FPGA boards for testing
- Read more on our blog



antmicro

CUSTOM RUNNERS IN ACTION(S) WITH GITHUB

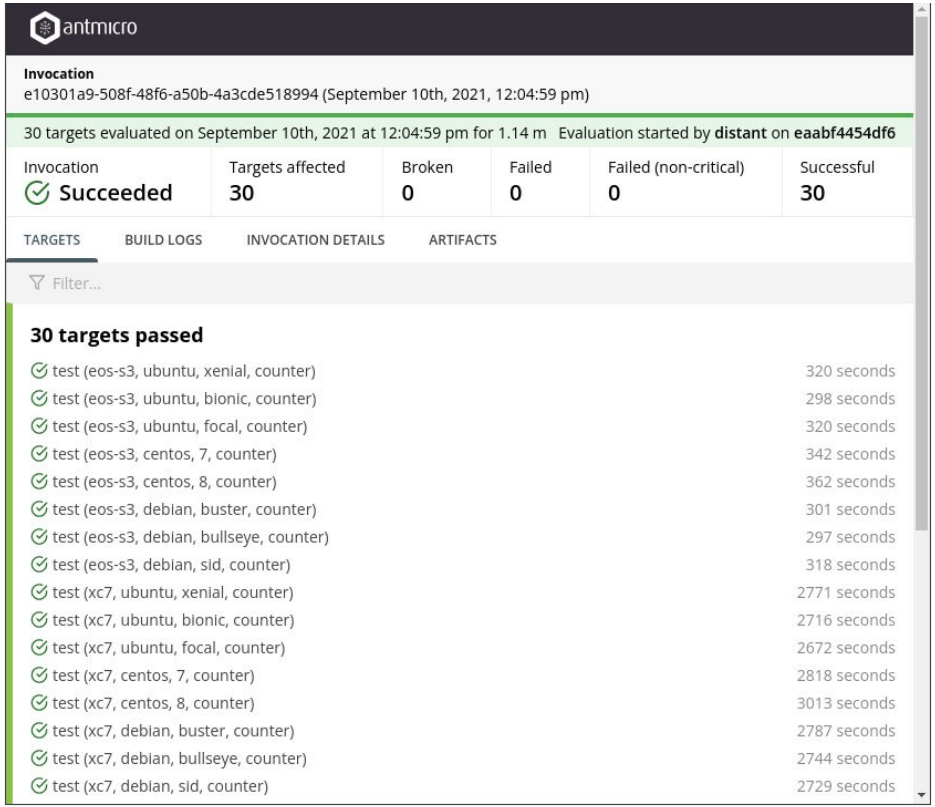
- Used for testing the UHDM integrations
- Gives us ability to do longer runs with more compute resources
- We can generate additional statistics of execution metrics, decrease resource usage and provide more insight
- Easier to find bottlenecks



antmicro

CUSTOM RUNNERS IN ACTION - DISTANT-BES

- Run private builds, but potentially share (sanitized) build logs/results
- Upload results to own servers
- Store them for as long as needed
- Create more customized dashboards and detailed views of “what went wrong”



antmicro

SUMMARY

- CHIPS Alliance is building an open source SystemVerilog tooling ecosystem - join us!
- **Practical use cases are possible already now, both in local and collaborative / cloud development**
- More features and use cases are actively being worked on
- We want to hear about your needs and projects!
- Join our mailing lists: sv-wg@lists.chipsalliance.org

WANT TO USE OPEN SOURCE SYSTEMVERILOG TOOLS?

reach out to us:

contact@antmicro.com