Linux Plumbers Conference 2022

>> Dublin, Ireland / September 12-14, 2022

How to introduce KUnit to physical device drivers?

Kernel Testing & Dependability Micro Conference

Tales L. Aparecida <tales.aparecida@gmail.com>
B.Sc. Computer Science - Unicamp/Brazil

Abstract

Unit testing is a great way to ensure code reliability, leading to organic improvements, as it's often possible to integrate it with developers' workflows. It is also of great help when refactoring, which should be a primordial task in large code bases. When it comes to the Linux kernel, the KUnit framework looks very promising, as it works natively from inside the kernel, and provides an infrastructure for running tests easily.

We are seeing a growing interest in unit testing on the DRM subsystem, with amazing initiatives to add KUnit tests to the DRM API. Moreover, three GSoC projects under the X.Org Foundation umbrella target unit tests for AMDGPU display drivers, as it is currently the largest one in the kernel. It is, thus, of great importance to discuss problems and possible solutions regarding the implementation of KUnit tests, especially for hardware drivers.

Bearing this in mind, and as part of our GSoC projects [1], we introduce unit testing to the AMDGPU driver departing from the Display Mode Library (DML), which is a library focused on mathematical calculations for DCN (Display Core Next); we also explore the addition of new tests to DCE (Display Controller Engine). Since AMD's CI already relies on IGT GPU Tools (a test suite for DRM drivers) we also propose an integration between it and KUnit which allows for DRM KUnit tests to be run through IGT as well.

In this talk, we present the tests' development process and the current state of KUnit in GPU drivers. We discuss the obstacles we faced during the project, such as generating coverage reports, mocking a physical device, and especially in regards to the implementation of tests for the AMDGPU driver stack, with the additional difficulties associated with making them IGT compatible. Finally, we want to discuss with the community lessons learned using KUnit in GPU drivers and how to reuse these strategies for other GPU drivers and also drivers in other subsystems.

[1] https://summerofcode.withgoogle.com/programs/2022/organizations/xorg-foundation

What this is about

This talk is the result of multiple **Google Summer of Code** Projects under the **X.Org Foundation[1]** umbrella

- Developed by
  - Maíra Canal
  - Magali Lemes
  - Isabella Basso
  - Tales Aparecida
- Mentored by
  - Rodrigo Siqueira (AMD)
  - Melissa Wen (Igalia)
  - André Almeida (Igalia)

[1] https://summerofcode.withgoogle.com/programs/2022/organizations/xorg-foundation

What this is about

Goal: Learn how to write unit tests for physical device drivers using AMD display driver as the subject.

The AMD display driver is part of the DRM subsystem and offers support for many generations of GPUs, which results in its huge codebase.

DRM - Direct Rendering Manager

Expected challenges

I. Initial expectation to find a steep learning curve to write our first KUnit tests **mocking a GPU**.

II. Worry about how we should organize the code to encourage the community to write more tests while also aligning with the driver's maintainers guidelines

Development

Converted DRM **kselftests** into **KUnit** tests

- Started in a LKCamp + Flusp 2021 Hackathon
- This tests are part of the official test suite for any DRM change
  - This suite of tests was integrated into IGT GPU Tools

Students groups respectively from State University of Campinas (Unicamp) + University of São Paulo (USP)

Development

I. First tests for functions clearly independent from the GPU to learn KUnit basics
   - First encounter with static functions that could be tested
II. More tests, this time for a given generation, but still no hardware mocking required, just some test fixtures
III. Finally, some tests based on regressions, and still no physical device mocking, but more static testing required

KUnit + gcov: Test Coverage

- Shows how much source code is run when the tests are executed
- Helps to track progress
- Potential projects for beginners

LCOV - code coverage report

			Hit	Total	Coverage
Current view:	top level - drivers/gpu/drm/amd/display/dc	Lines:	37	673	5.5 %
Test:	coverage.info	Functions:	1	58	1.7 %
Date:	2022-08-30 20:32:10				

Filename	Line Coverage			Functions	
dc_dmub_srv.c		10.9 %	37 / 340	4.0 %	1 / 25
dc_edid_parser.c		0.0 %	0 / 18	0.0 %	0 / 3
dc_helper.c		0.0 %	0 / 295	0.0 %	0 / 27
dc_link.h		0.0 %	0 / 14	0.0 %	0 / 2
dm_services.h		0.0 %	0 / 6	0.0 %	0 / 1

Generated by: LCOV version 1.14

Directory	Line Coverage			Functions	
drivers/gpu/drm/amd/display/dc/dml		5.3 %	83 / 1567	4.0 %	6 / 149
drivers/gpu/drm/amd/display/dc/dml/calcs		1.7 %	74 / 4278	15.9 %	7 / 44
drivers/gpu/drm/amd/display/dc/dml/dcn20		2.9 %	219 / 7554	9.3 %	7 / 75

Retrospective

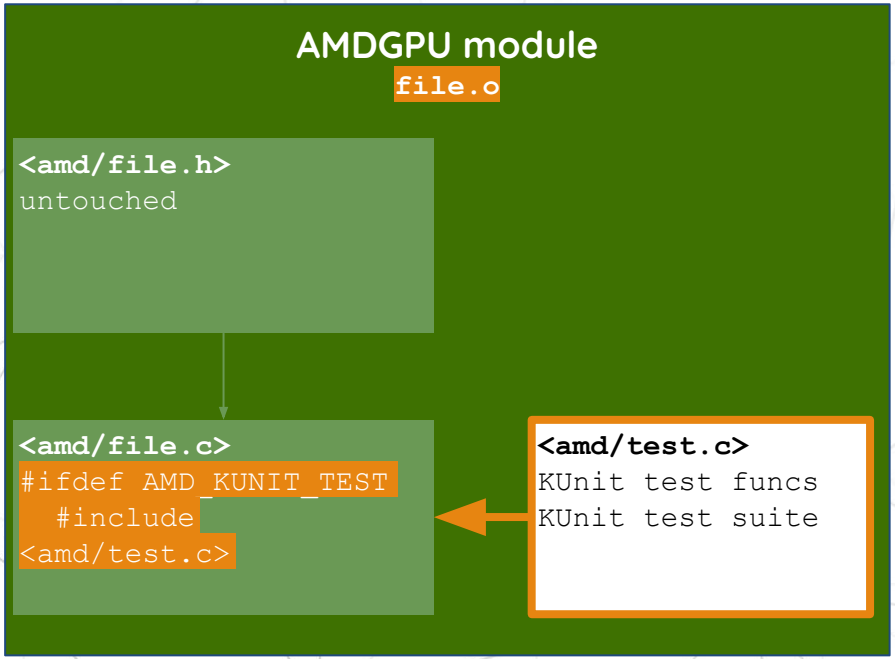
- A lot of code is actually hardware-agnostic!
  - Device mocking is probably necessary in some cases, but there are a lot of low hanging fruit without it
- Static functions might require tests
  - Not everyone agrees with testing static functions
  - It can be tricky to reach all code paths otherwise

There are many alternatives for testing static functions

1. test them inside the driver's module
   a. Keep them static
   b. Stop being static
2. test them in a standalone testing module
   a. Stop being static and export them
   b. Stop being static and export the test cases
   c. Keep them static and export the test cases

Tests suite injected into the driver

keep it static



All tests run on load

Define tests cases and suites in a file that is appended to the tested file footer.

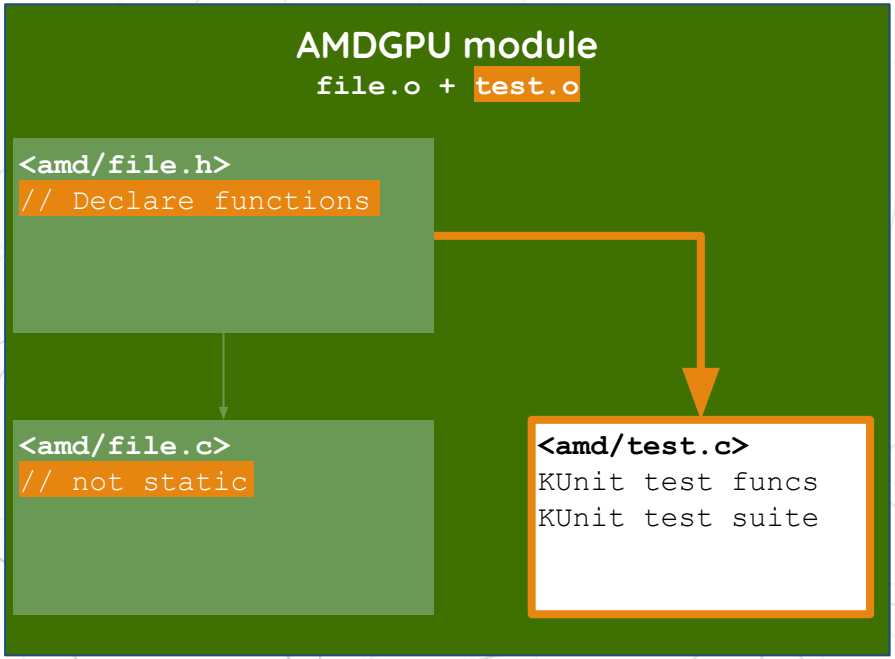
Don't need any exported symbols.
Don't need declaration in header files.
Works for any functions in the file, even static.

No need to edit makefile

Documented at
https://docs.kernel.org/dev-tools/kunit/tips.html#testing-static-functions

Tests suite inside the driver

function is no longer static



All tests run on load

Define tests cases and suites in a file that is linked on compilation to the tested module.

Don’t need any exported symbols.
Need functions declared on header
Works for any functions in the header file.

Minor edit to makefile

Export driver symbols [conditionally]

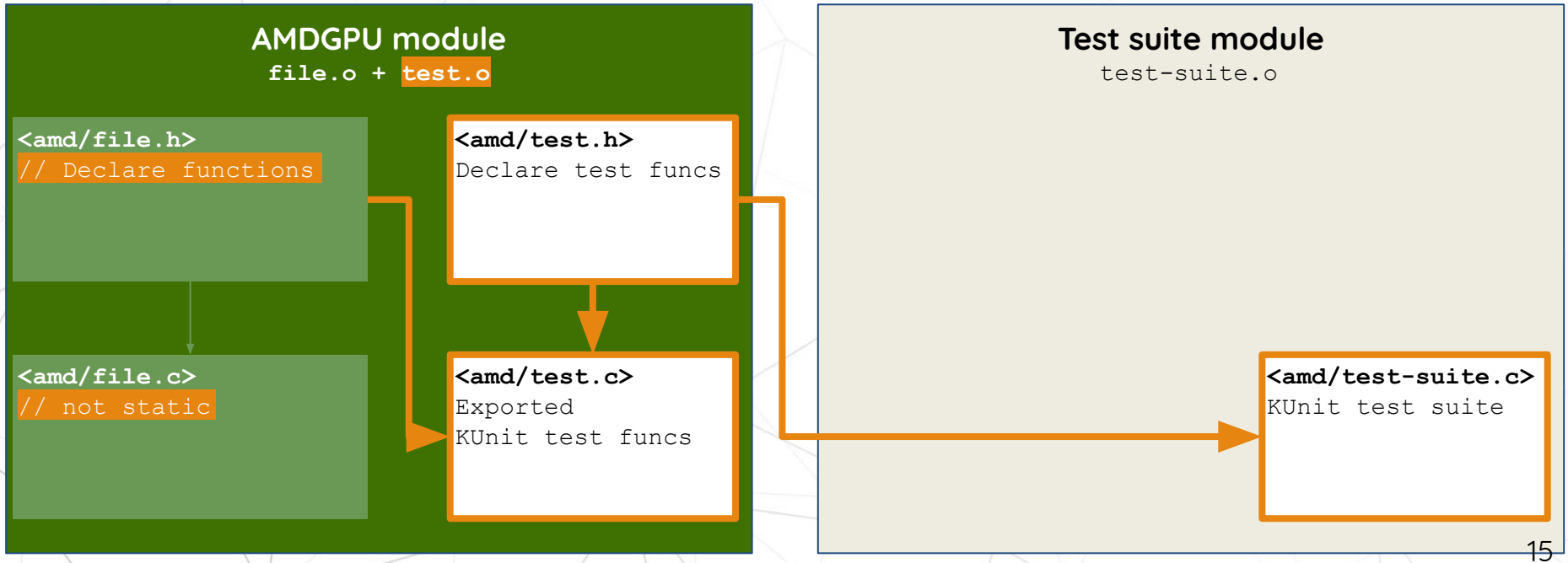
function is no longer static

If you want to run test modules for declared functions, you only need to export when testing

Export test cases symbols

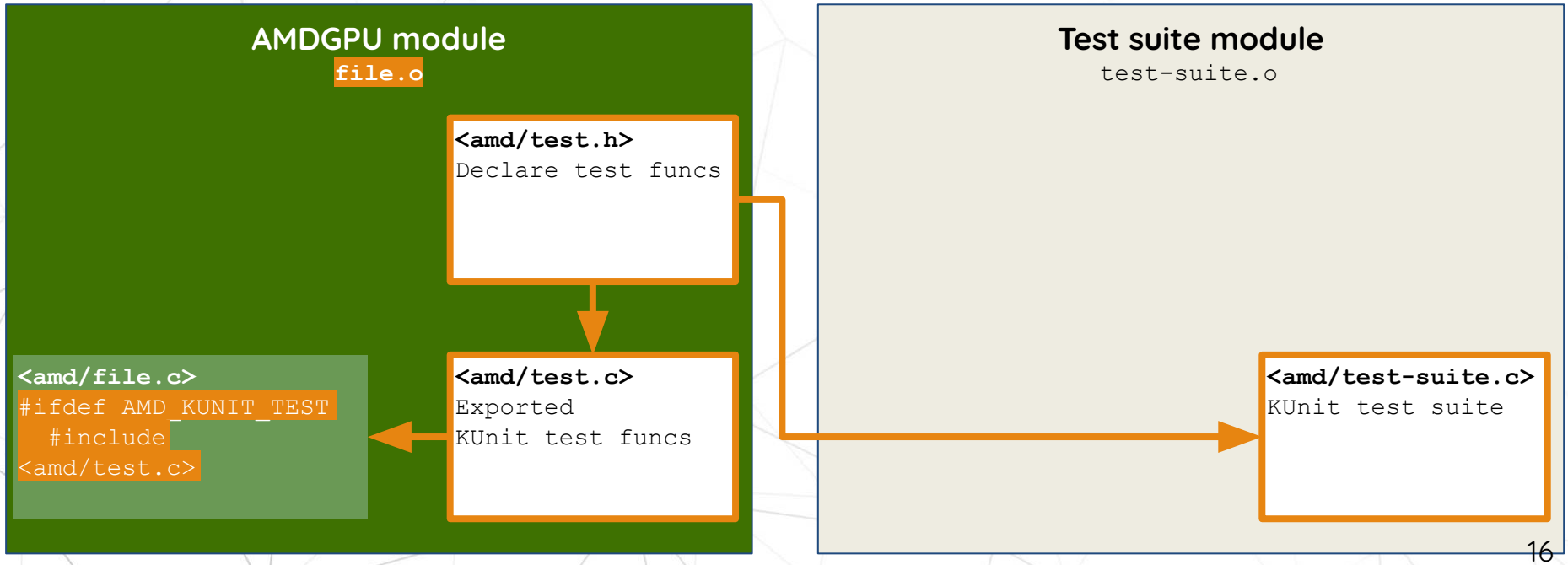
function is no longer static

If you want to run test modules for declared functions, but don't want conditional exporting



Inject and export tests cases
keep it static

If you want to tackle both issues, run test modules and include static functions



Acknowledgements

- Google Summer of Code
  - X.Org Foundation
- Mentors from AMD and Igalia
- Community (DRM, Kunit engineers)
- The Linux Foundation

Linux Plumbers Conference 2022

>> Dublin, Ireland / September 12-14, 2022

See also my teammates at XDC

October 4-6

“KUnit sorcery and the uncanny nature of FPU in the DRM”

Maíra Canal <mairacanal@riseup.net>

Magali Lemes <magalilemes00@gmail.com>

Isabella Basso <isabbasso@riseup.net>

Let's discuss

- Unit tests for static functions
- Standalone test modules VS

  test inside driver's modules
- Other topics

Should we test static functions?

a) yes.

b) no.

c) no, convert them if you want to test

What to unit test?

a) only exported symbols

b) symbols at least declared in header (.h) files

c) anything, including static functions

IGT + Kunit?

a) I prefer to run KUnit inside IGT

b) I prefer running them separately

c) I don’t mind

Reach us

Tales Aparecida <tales.aparecida@gmail.com>

Maíra Canal <mairacanal@riseup.net>

Magali Lemes <magalilemes00@gmail.com>

Isabella Basso <isabbasso@riseup.net>