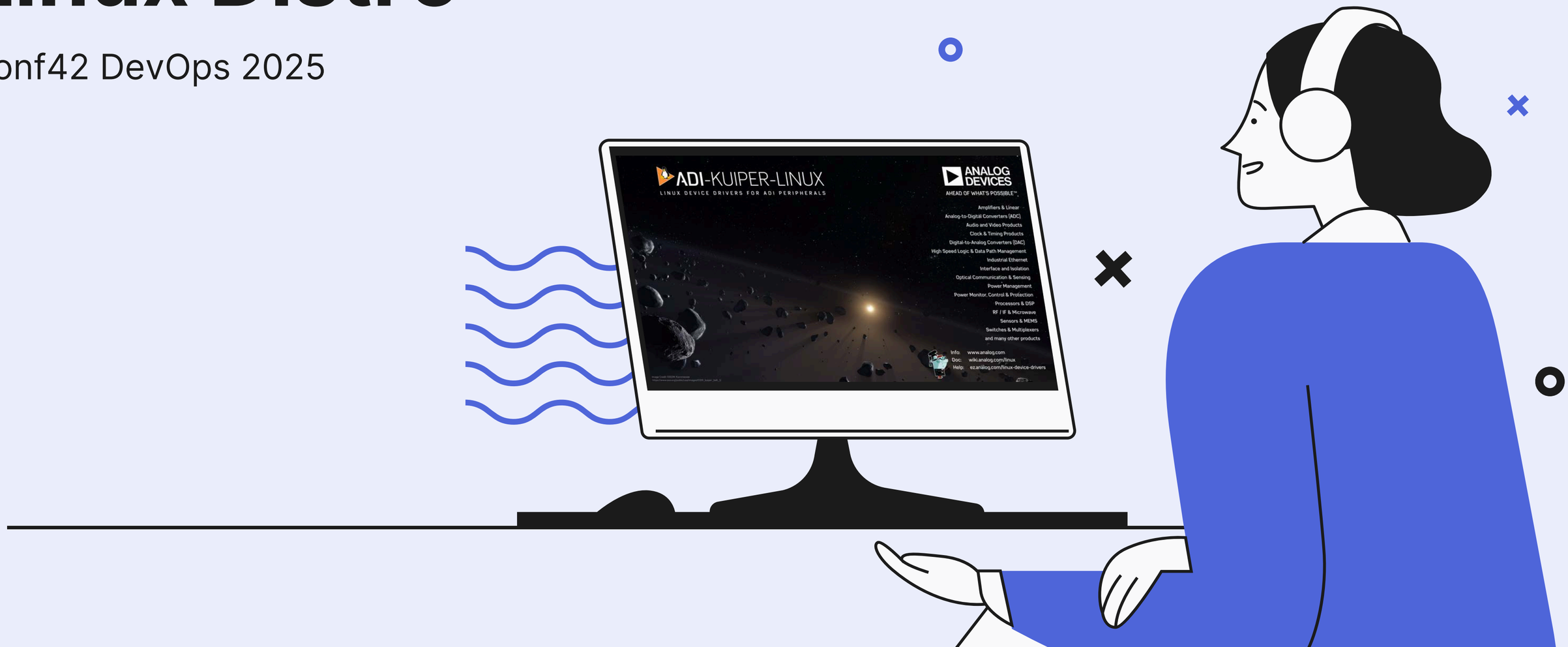
Refining the Release Strategy of a Custom Linux Distro

Conf42 DevOps 2025



What is ADI Kuiper Linux?

Free, open-source Embedded Linux Distribution

Kuiper is the primary distribution for product evaluation boards made by Analog Devices.
It includes reference designs, device drivers and a variety of development utilities.

Main Discussion Points

What we had when we started

Changes that were made in order to improve the release process

The current state of our Linux Distribution

The previous release process

No modularity

The software components were predefined and installed during the build process

Components were built during image creation

This meant longer build time

The software components were released through Kuiper Linux image

This lead to longer release cycles due to bugs affecting only one software component

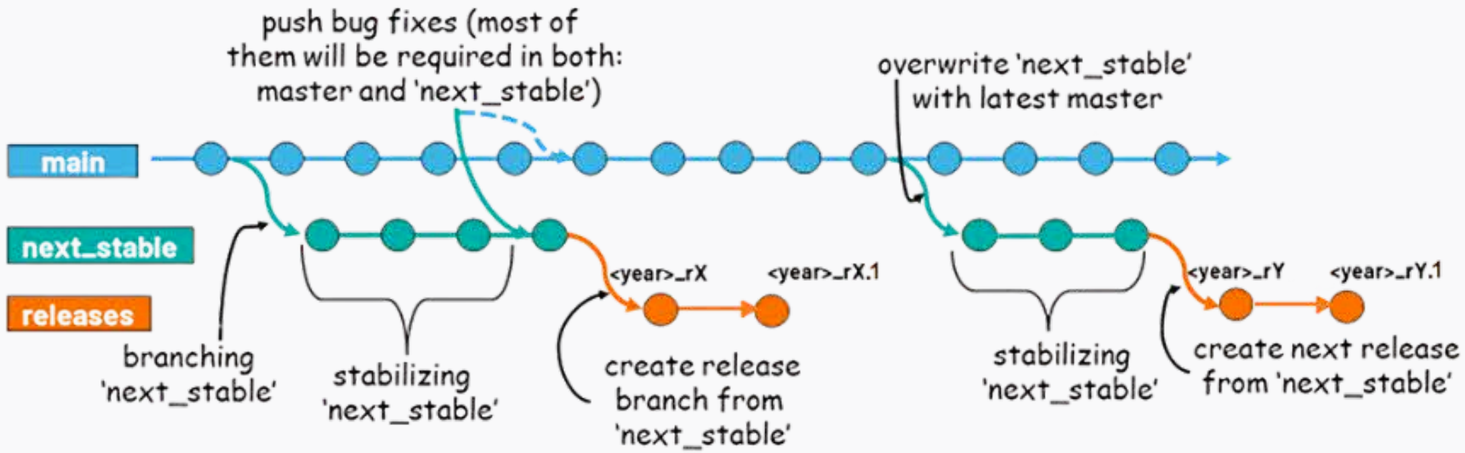
Release improvemens

- Creating a stabilization branch
- Breaking down the monolith
- Creating Debian packaging for components
- Modular configuration for custom images
- Expand component testing with custom Docker images
- CI optimizations and improving collaboration

Stabilization Branch:
Isolating Release Development

Purpose:
Isolate release stabilization efforts from ongoing development.
Avoid disruptions caused by experimental or unfinished features.

Benefits:
Faster bug fixes and focused testing.
Stable release preparation without affecting the main branch.



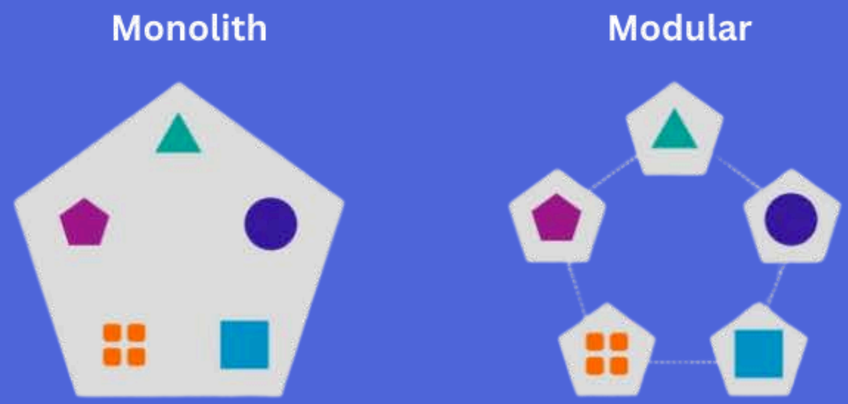
Independent Components Releases && Debian Packages

The release process is now divided into smaller, independent components

Prevent delays caused by bugs in a single component.
Faster releases and easier debugging.

Each component is released as a Debian package and uploaded to Analog Devices' Linux Package Repository.

Simplified distribution using standard Debian tools
Easier installation and rollback for users



Customizable builds through modular configuration

A new configuration file enables modularity

Users can build custom images by selecting only the tools and components they need.

New components or additional dependencies can be easily added

Users can use a dedicated extra script to include them.

Testing Software Componens with Dockerized Kuiper Images

Dual Testing Approach

Components are tested individually for standalone functionality, but also ensures compatibility within the Kuiper environment early in the process, by detecting and resolveing integration issues faster.

Lighter Docker Images

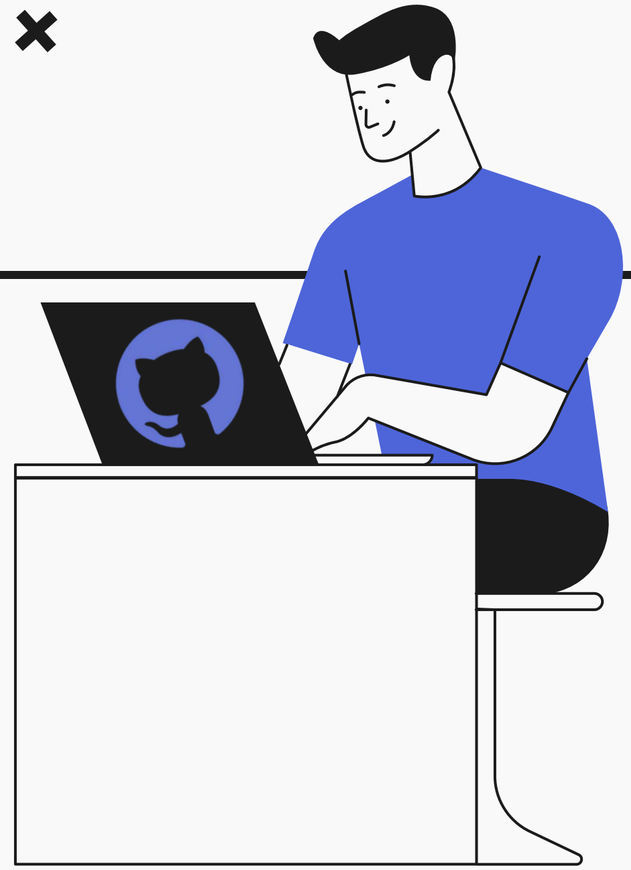
Because the image is now customizable, we can create a lighter image using only the necessary dependencies for testing, from which we can also create a Docker image.

Presentations about testing:

Secure Integration of Private Testing Infrastructure with Public GitHub Repositories - Bianca Popu

Advanced Test Harness Infrastructure for Validating ARM and FPGA-based Systems - Stefan Raus

Optimizing Builds with GitHub Actions



Moved from Jenkins to GitHub Actions

Build logs are accessible to all users.
Fresh build images available immediately after a commit is merged: this means users don’t need to wait for a release or to build the image manually.

Multiple build configurations

Four configurations are now available: a basic and a full image for each of the 32-bit and 64-bit architectures.

Improved Collaboration

Introduced a Pull Request Template for consistency and Codeowners to streamline reviews and approvals.

The current release process

Customizable builds through modular configuration

With the help of the configuration file, the software components that need to be installed are selected.

Components are built separately and exported as Debian packages

Faster and more flexible builds

Each software component has it’s own release

Prevents delays caused by bugs in a single component and improves flexibility and development efficiency

Thank you!

If you have questions you can write me
on this email address: andreea.andrisan@analog.com
or on LinkedIn: Andreea Andrisan