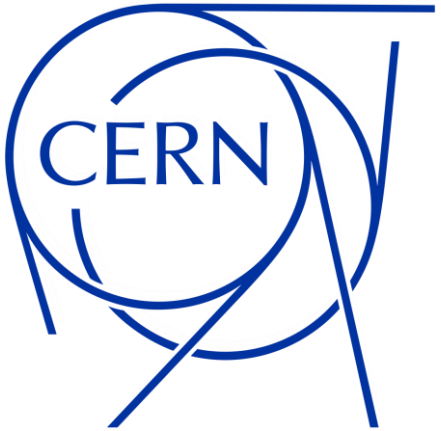


The new CI4FPGA service

Gitlab CI for FPGA/SoC Projects

Carmen Marcos[1], Christos Gentsos[1]*, Konstantinos Evangelou[2], Ismael Posada Trobo[2] ([1]: IT-CA-GES, [2]: IT-PW-WA)

21 May 2025

Why CI Matters for Your FPGA Projects

Wait, what is CI again?

CI involves regularly integrating code changes from multiple contributors into a shared codebase, followed by automatic testing and building of the result. This practice is a cornerstone of modern development, particularly for automated repository-level checks.

Key Benefits for FPGA Development

Consistent Automated Systems

Robust framework for various tests (unit, system-level, linting, code coverage) and builds (bitstream, drivers, utilities)

Triggered at key points in the workflow

Efficient Job Handling

Easy launching of job sets, despite long build times

⇨ frequent usage ⇨ less breakage, quicker regression bug-hunting

Enhanced Collaboration

Facilitates common development practices and seamless code sharing

Ideal for IP core libraries and team projects.

Beyond the Lab Machine: Scaling Challenges

Single lab machine: not that hard

👍 Get a beefy machine, install tools, gitlab-runner, it works

👎 Each team has to roll their own

- Machines often left unmaintained
- Still significant barrier to entry
- Fragmented efforts
- Unusable at scale

Scaling for a broad audience on k8s clusters… not that easy

- Huge toolchains, O(100GB)
- Can’t use publicly available flavors, lack of RAM
- Many releases ⇨ OS compatibility
- Tricks needed to use on containers
- Maintaining and monitoring clusters not FPGA work

Project Purpose and Objectives

Scalable VM-based k8s clusters

Develop scalable, maintained VM-based clusters for CI jobs, adapted to EDA tool resource requirements

Easy-to-Use Docker Images

Provide user-friendly Docker images for EDA tools, including simulators, toolchains and utilities

Comprehensive Documentation

Create clear and thorough docs to support the 100+ (then projected) users of the system

Purpose:

Accessible CI for all FPGA designers at CERN

Scalable VM-based k8s clusters

Core plat

Robust

- Kuberne
- Located

handling

e) access

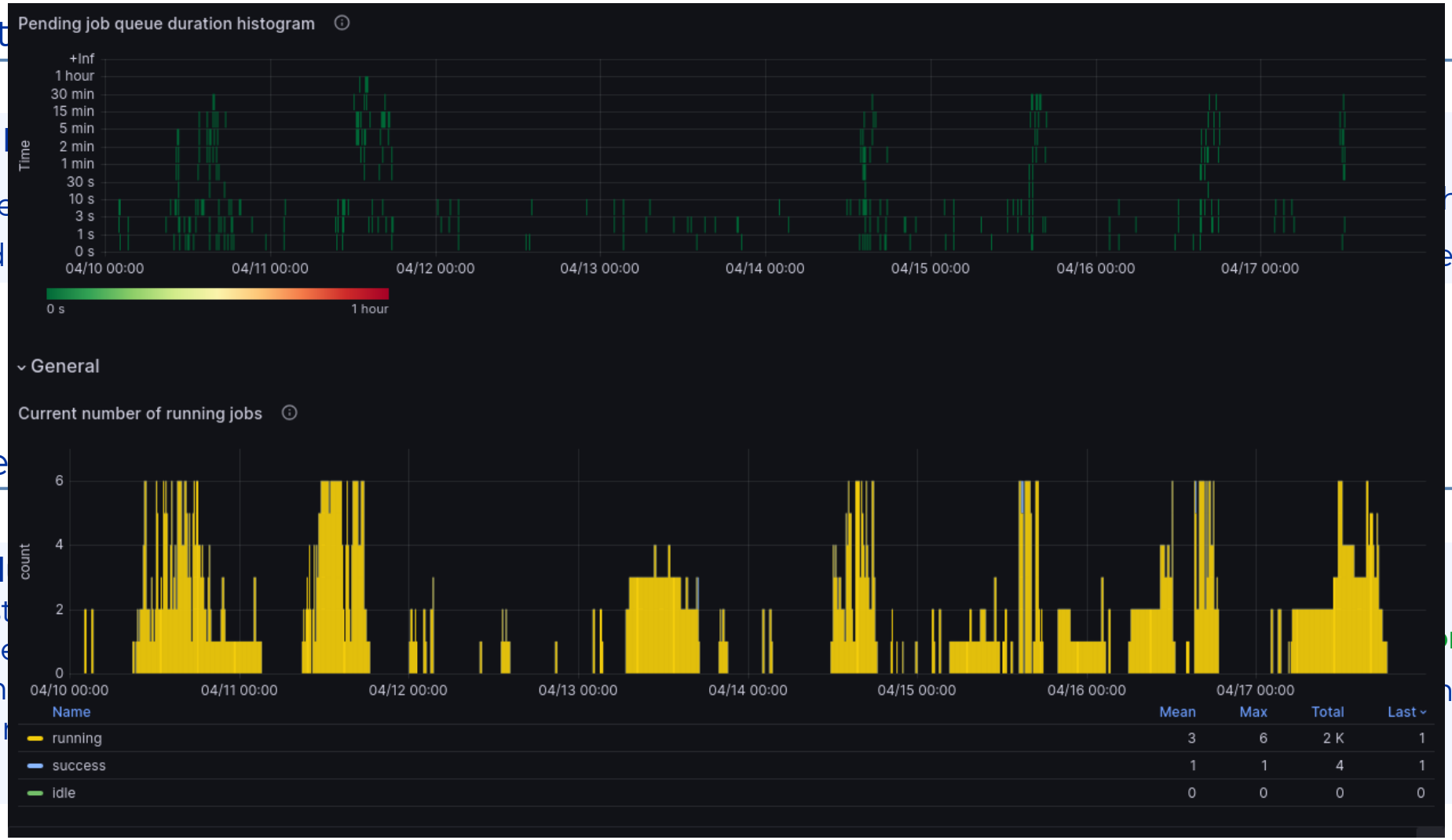
Optimize

Dual Cl

Two dist
resource
- fpga-m
- fpga-la

nds

nsuring



Easy-to-Use Docker Images

🛠 Pre-configured Docker images

- Extensive library: images for all needed tools, multiple versions per tool (85 images and counting)
- Simple to use: no need for any setup instructions in job scripts (license, paths)
- Careful: measures to prevent depriving real users from licenses

📚 Comprehensive User Documentation

- Automatically generated from image sources, ensuring consistency
- User-focused: Extensive FAQ

List of tools available as images

The following tools are available in the registry.cern.ch/ci4fpga project:

Toolchains

- Xilinx Vivado
- Xilinx ISE
- Microsemi Libero
- Intel Quartus
- Lattice Diamond

Simulators

- Riviera-PRO
- GHDL (GCC backend)
- GHDL (mcode backend)
- NVC
- QuestaSim
- ModelSim
- Icarus Verilog
- Altair DSim

Lint (and lin-related) tools

- VHDL Style Guide
- Aldec ALINT-PRO
- Verible

Documentation generation

- AMIQ EDA DVT
- Doxygen

Assorted tools

- YosysHQ OSS CAD Suite
- PetaLinux
- VisualElite
- Synopsys Synplify

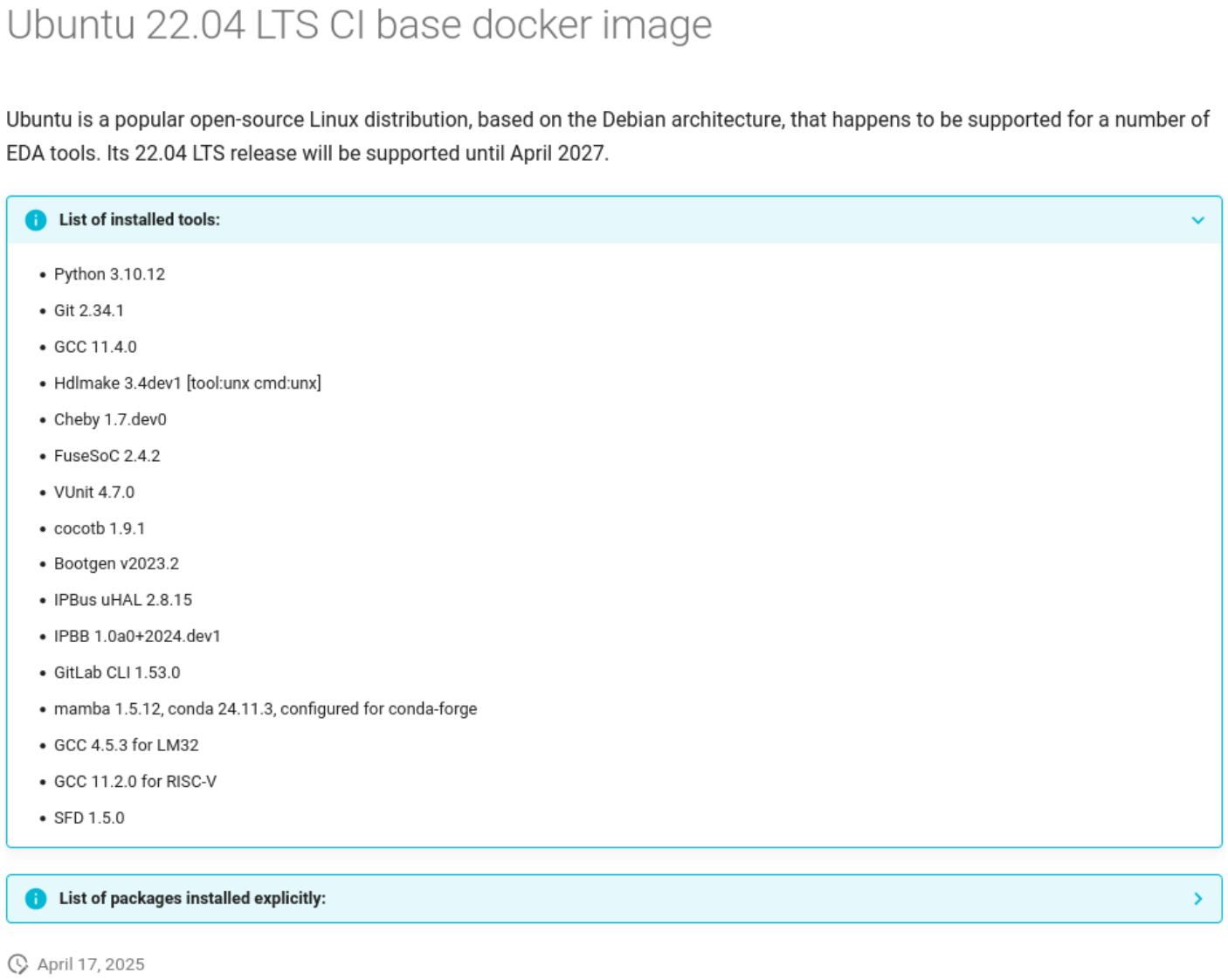
Easy-to-Use Docker Images

🛠 Multiple releases

- Supported OS used for each release
- OS for each release clearly documented

🛠 Base OS images rich in utilities

- Base images feature commonly used tools, like
  - hdlmake
  - Cheby
  - Vunit
  - Cocotb
  - Embedded toolchains
- Versions clearly documented



Easy-to-Use Docker Imag

Build summary for Xilinx Vivado 2024.2 libraries with Riviera-PRO 2025.04-beta

🛠 Pre-compiled simulation libraries

- Simulator images come with pre-compiled simulator libraries for most toolchains
- Greatly simplifies simulation tasks
- Simulation logs parsed and presented in the documentation

Library	Language	Mapped Library Name	Error(s)	Warning(s)
secureip	verilog	secureip	0	775
unisim	vhdl	unisim	0	3
unimacro	vhdl	unimacro	0	0
unifast	vhdl	unifast	0	0
unisim	verilog	unisims_ver	0	261
unimacro	verilog	unimacro_ver	0	17
unifast	verilog	unifast_ver	0	5
simprim	verilog	simprims_ver	0	613
xpm	vhdl	xpm	0	0
xpm	verilog	xpm	0	42
xilinx_vip	verilog	xilinx_vip	0	6
common_cpp_v1_0	cpp	common_cpp_v1_0	0	0
xtlm	systemc	xtlm	0	1
xscl	systemc	xscl	0	1
emu_perf_common_v1_0	systemc	emu_perf_common_v1_0	1	0
axi_tlm_ext_v1_0	systemc	axi_tlm_ext_v1_0	0	1
axi_tg_sc_v1_0	systemc	axi_tg_sc_v1_0	1	0
noc_sc_v1_0_0	systemc	noc_sc_v1_0_0	1	1
noc_sc_v1_0_0_legacy	systemc	noc_sc_v1_0_0_legacy	1	0
remote_port_c_v4	c	remote_port_c_v4	0	0

Defining / Moving CI Jobs

```
build-job:
  tags:
- fpga-mid
  image: registry.cern.ch/ci4fpga/vivado:2022.2
  script:
- vivado -mode batch -source build.tcl
  artifacts:
paths:
        - "*.rpt"
        - "*.bit"
```

Simple to begin with CI

- Assuming you have a command / script you can run, something like that is all it takes
- If you already have a CI implementation, only things need to be changed are the tag and the image
  - Feedback on the migration has been great; in some cases, it just took 5 minutes
- No need to register anywhere, just go to the documentation, everything should be there – please open a ticket if it isn't

Existing Users

- 119 users are already using CI4FPGA with 130 projects; more than 20k jobs completed so far

```
14/05/25-15:38:02, start: image registry.cern.ch/ci4fpga/ghdl:dev, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:03, start: image registry.cern.ch/ci4fpga/petalinux:2018.1, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:03, start: image registry.cern.ch/ci4fpga/doxygen:1.12.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:04, start: image registry.cern.ch/ci4fpga/doxygen:1.12.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:05, start: image registry.cern.ch/ci4fpga/vsg:3.26.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:27, end: image registry.cern.ch/ci4fpga/doxygen:1.12.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:41, end: image registry.cern.ch/ci4fpga/ghdl:dev, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:47, end: image registry.cern.ch/ci4fpga/vsg:3.26.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:38:48, end: image registry.cern.ch/ci4fpga/petalinux:2018.1, triggered as web by [redacted] from job [redacted]
14/05/25-15:39:16, end: image registry.cern.ch/ci4fpga/petalinux:2018.1, triggered as web by [redacted] from job [redacted]
14/05/25-15:39:19, end: image registry.cern.ch/ci4fpga/petalinux:2018.1, triggered as web by [redacted] from job [redacted]
14/05/25-15:39:25, start: image registry.cern.ch/ci4fpga/ghdl:4.1.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:39:44, end: image registry.cern.ch/ci4fpga/doxygen:1.12.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:40:05, end: image registry.cern.ch/ci4fpga/ghdl:4.1.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:40:50, start: image registry.cern.ch/ci4fpga/vivado:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-15:40:50, start: image registry.cern.ch/ci4fpga/vivado:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-15:40:50, start: image registry.cern.ch/ci4fpga/doxygen:1.12.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:42:28, end: image registry.cern.ch/ci4fpga/doxygen:1.12.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:50:42, end: image registry.cern.ch/ci4fpga/vivado:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-15:52:49, end: image registry.cern.ch/ci4fpga/vivado:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-15:53:47, start: image registry.cern.ch/ci4fpga/vsg:3.26.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:53:47, start: image registry.cern.ch/ci4fpga/vsg:3.26.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:53:48, start: image registry.cern.ch/ci4fpga/petalinux:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-15:53:51, start: image registry.cern.ch/ci4fpga/petalinux:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-15:54:20, end: image registry.cern.ch/ci4fpga/vsg:3.26.0, triggered as web by [redacted] from job [redacted]
14/05/25-15:54:56, end: image registry.cern.ch/ci4fpga/vsg:3.26.0, triggered as web by [redacted] from job [redacted]
14/05/25-16:10:55, end: image registry.cern.ch/ci4fpga/petalinux:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-16:11:04, end: image registry.cern.ch/ci4fpga/petalinux:2024.2, triggered as web by [redacted] from job [redacted]
14/05/25-16:18:46, start: image registry.cern.ch/ci4fpga/vivado:2024.2, triggered as push by [redacted] from job [redacted]
14/05/25-16:18:47, start: image registry.cern.ch/ci4fpga/vivado:2024.2, triggered as push by [redacted] from job [redacted]
```

Future Plans

Governance

- Strategy for which images should be actively maintained, which should be frozen and when, missing
- If we only continue to add images without putting some aside as “legacy”, documentation will become unusable after some years
- Somebody needs to decide (how to decide)

Other ideas

- Identify shareable CI job “recipes”, share centrally?
- Add clusters w/more RAM? (needs official request for resources)
- ???

Many thanks!

Any questions?