Vitis CI/CD & Containerization

Ben Hawks, Mariana Gonzalez Velarde, Andrew Whitbeck
Fermi National Accelerator Laboratory - Batavia, IL USA

FERMILAB-POSTER-25-0222-AD-CSAID

ACORN Motivation

The Accelerator Control Operations Research Network (ACORN) project will modernize the Fermilab accelerator control system. The ACORN data acquisition and control (DAC) hardware architecture is based around deploying more than 1000 System-on-Chip (SoC) devices. This distributed architecture requires custom firmware to implement applications across a variety of devices, all of which must be developed and maintained efficiently. The scale and diversity of the deployment makes firmware and on-device software integration testing critical to system reliability. Here we describe a Kubernetes-based CI/CD pipeline to scalably support ACORNs hardware development and maintenance. Below is a summary of the SoC development flow.

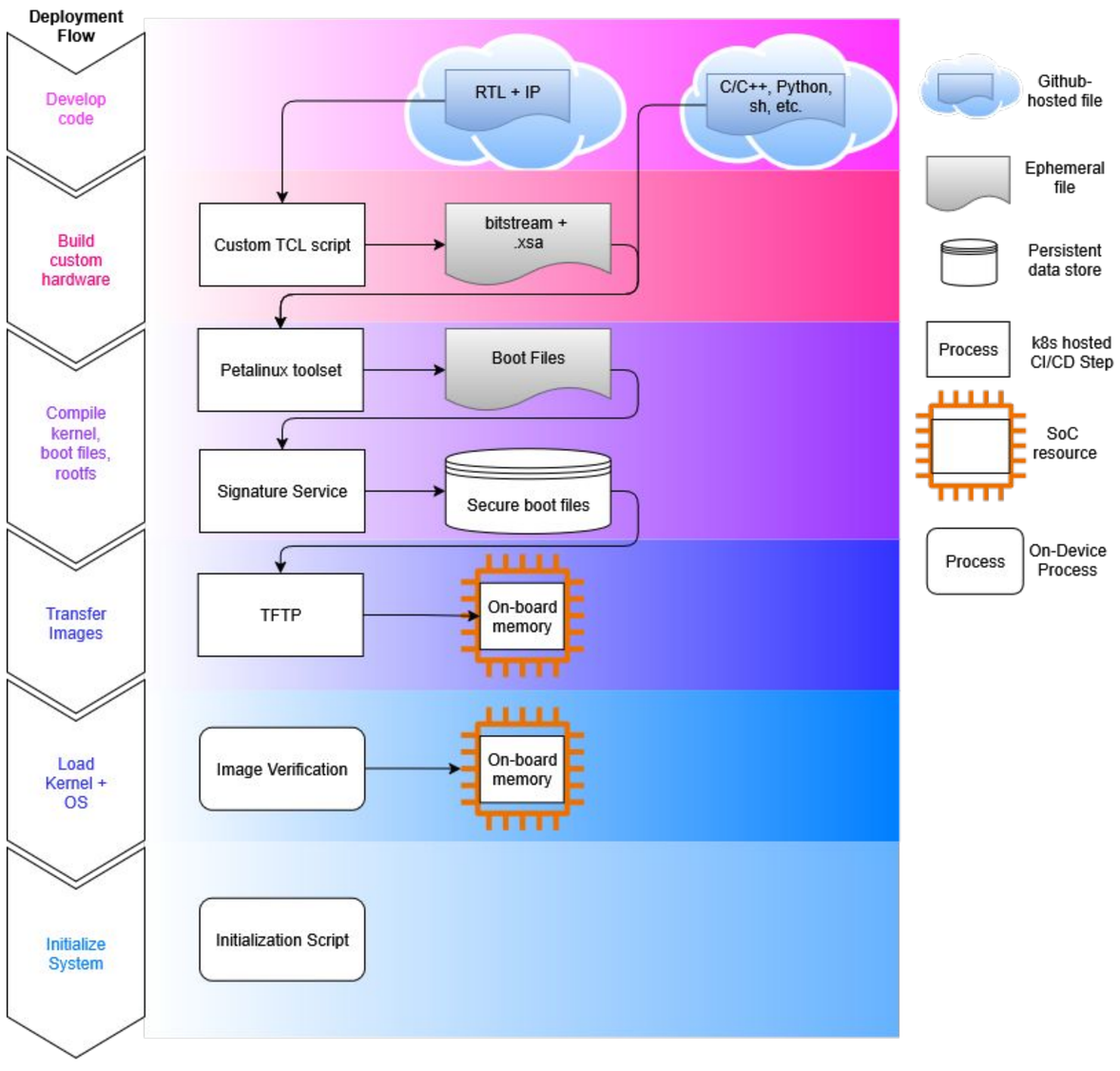

Figure 2: Proposed Deployment Workflow for ACORN DAC Devices

We have successfully implemented a proof of concept version of this workflow, implementing the Github Actions and firmware build steps of the process. We are currently in the process of building out the rest of the workflow that leverages the Yocto build pipeline, as well as firmware signing and hardware deployment.

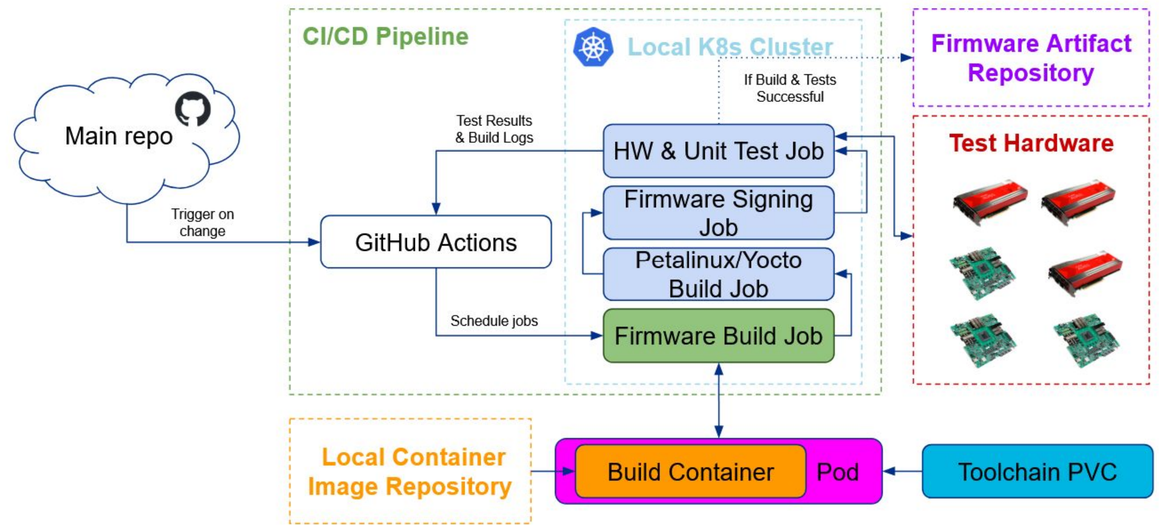

Figure 1: Proposed ACORN DAC CI/CD Workflow and Infrastructure

Github CI/CD & K8s Orchestration

We leverage **GitHub Actions** integrated with **Kubernetes** to provide scalable, automated CI/CD for our FPGA and software build workflows. A custom container image bundles the required OS, build tools, and the GitHub Actions Runner agent. Using **actions-runner-controller**, we orchestrate ephemeral self-hosted runners directly in Kubernetes, enabling on-demand execution, automatic scaling, and reproducible environments. This architecture minimizes idle resources, accelerates feedback cycles, and ensures consistent build results across the pipeline.

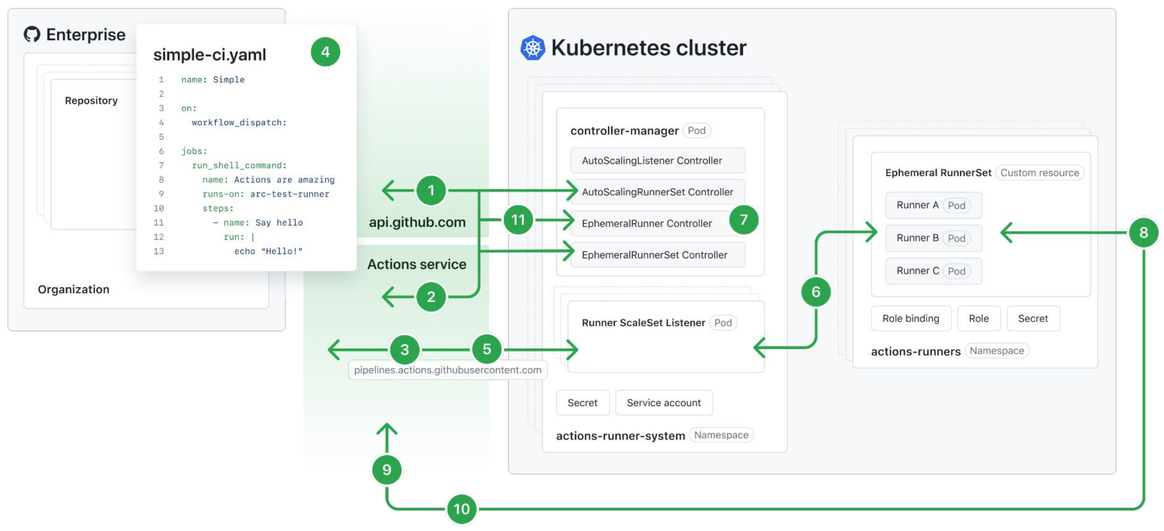

Figure 3: GitHub's actions-runner-controller autoscaling workflow [1]

Vitis Containerization

In order to run firmware builds on **Kubernetes** and **GitHub Actions**, we needed to containerize **AMD's Vitis FPGA Toolchain**. This is achieved by using Jason Moss' "xilinx-docker" image [2], which contains all the Vitis software dependencies, as a base, installing further required tooling to support firmware builds and github actions, and deploying the customized image to the Kubernetes cluster image repository.

Due to the toolchain size, AMD Vitis and **Yocto** Artifact Caches are installed onto Kubernetes Persistent Volume Claims, which are then mounted as volumes into the appropriate Kubernetes pods as they're deployed.

References:
[1] GitHub 2025. Actions Runner Controller - GitHub. https://docs.github.com/en/actions/concepts/runners/actions-runner-controller
[2] Jason Moss. 2025. xilinx-docker. https://gitlab.com/rjmoss/xilinx-docker

This manuscript has been authored by FermiForward Discovery Group, LLC under Contract No. 89243024CSC000002 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics.

FERMILAB-POSTER-25-0222-AD-CSAID