Testing Solutions for Siemens PLC programs based on PLCSIM Advanced

G. Sallai, D. Darvas, E. Blanco Viñuela, CERN, Geneva, Switzerland
gyula.sallai@alumni.cern, daniel.darvas@alumni.cern, enrique.blanco@cern.ch

PLC program testing

Traditional PLC program testing

Through a command line interface, the test workflow can be executed in a CI pipeline (such as Jenkins or Gitlab CI).

Automated execution on each commit.

Checks before merge.

Execution on a remote machine – no need to install the simulator on every developer workstation.

Traditional PLC program testing is often difficult, because:

- physical hardware is needed for the test execution,
- it requires manual effort to feed inputs and evaluate outputs,
- it is imprecise due to the lack of proper synchronisation.

Testing through simulators

Testing through a simulator offers several benefits:

- tests may be run on a virtual PLC: no need for physical hardware,
- may be automated through an API,
- precise time- and cycle-based synchronisation options.

For Siemens PLCs: PLCSIM Advanced

Testing workflow built on PLCSIM Advanced

Test definition

We use an intuitive, but powerful test table format with support for

- access by tag names and memory locations,
- complex expressions,
- cycle-based and time-based duration requirements,
- references between test steps.

Test project generation

Certain levels and types of testing (unit, integration, etc.) cannot be carried out on a PLC project without some modifications.

- They require isolation, which may mean intrusive modifications of the PLC program.
- They need a test driver, that must be inferred from the list of program blocks.

Our solution: automated generation of a new project for unit testing

Test execution

- The generated test project is executed on a virtual PLC.
- PLCSIM Advanced supports multiple execution modes, such as continuous, cycle-by-cycle, and time-synchronous.
- Creation of a simulator instance, PLC program download and execution are all automated through the PLCSIM C# API and TIA Portal Openness.
- The captured outputs are evaluated against the constraints in the test definition.

Continuous integration

Through a command line interface, the test workflow can be executed in a CI pipeline (such as Jenkins or Gitlab CI).

- Automated execution on each commit.
- Checks before merge.
- Execution on a remote machine – no need to install the simulator on every developer workstation.

Test report and visualisation

- The tool produces test reports in various formats, such as plaintext, HTML, timing diagrams, waveform diagrams.
- The behaviour of the system can be recorded in every cycle, enabling the display of detailed timing diagrams for each test case execution.

Use cases

Our testing workflow has demonstrated its usability in acceptance, integration and unit testing scenarios.

- Proof-of-concept CI testing workflow for the baseline objects included in the UNICOS framework.
- Automatically reproducing counterexamples obtained from our formal verification workflow (PLCverif) for the CERN SPS accelerator personnel protection system.