

October 1st, 2026

Requirements to Test Stand in a Day

By Hunter Smith, Sr Solutions Engineer

The half-inch critical solenoid valve used on a rocket engine test stand you work on just died, and the replacement has a 16-week lead time. The original valve had been tested by the supplier and was guaranteed to meet a critical requirement of response time ≤ 250 ms. So now the stand is not operational. You have an old solenoid valve from a different vendor, but that vendor never tested the response time. If you can qualify the old valve to meet the specs by tomorrow, you can get the stand back in action and keep your test campaign on time.

Here are the test requirements:

- Cycle a half-inch solenoid valve 10,000 times at 50 psi.
- Measure open and close response time on every cycle.
- Measure seat leakage with the valve shut.
- Abort if the coil passes 120 °C.
- Pass criteria: ≤ 250 ms response.

The re-qualification requirements can mean a substantial software project. Alongside the plumbing, instrumentation, and wiring, the qualification rig needs software to acquire and scale signals, sequence the valve, measure each response, check limits, and record the results. It also needs an operator interface that makes it clear what the stand is doing. Connecting those pieces and verifying that they work together is a significant part of getting the test ready to run.

You have enough hardware challenges to meet the deadline. Test software should not be one of them. On RevelTest there are just three configurations: the IO, the

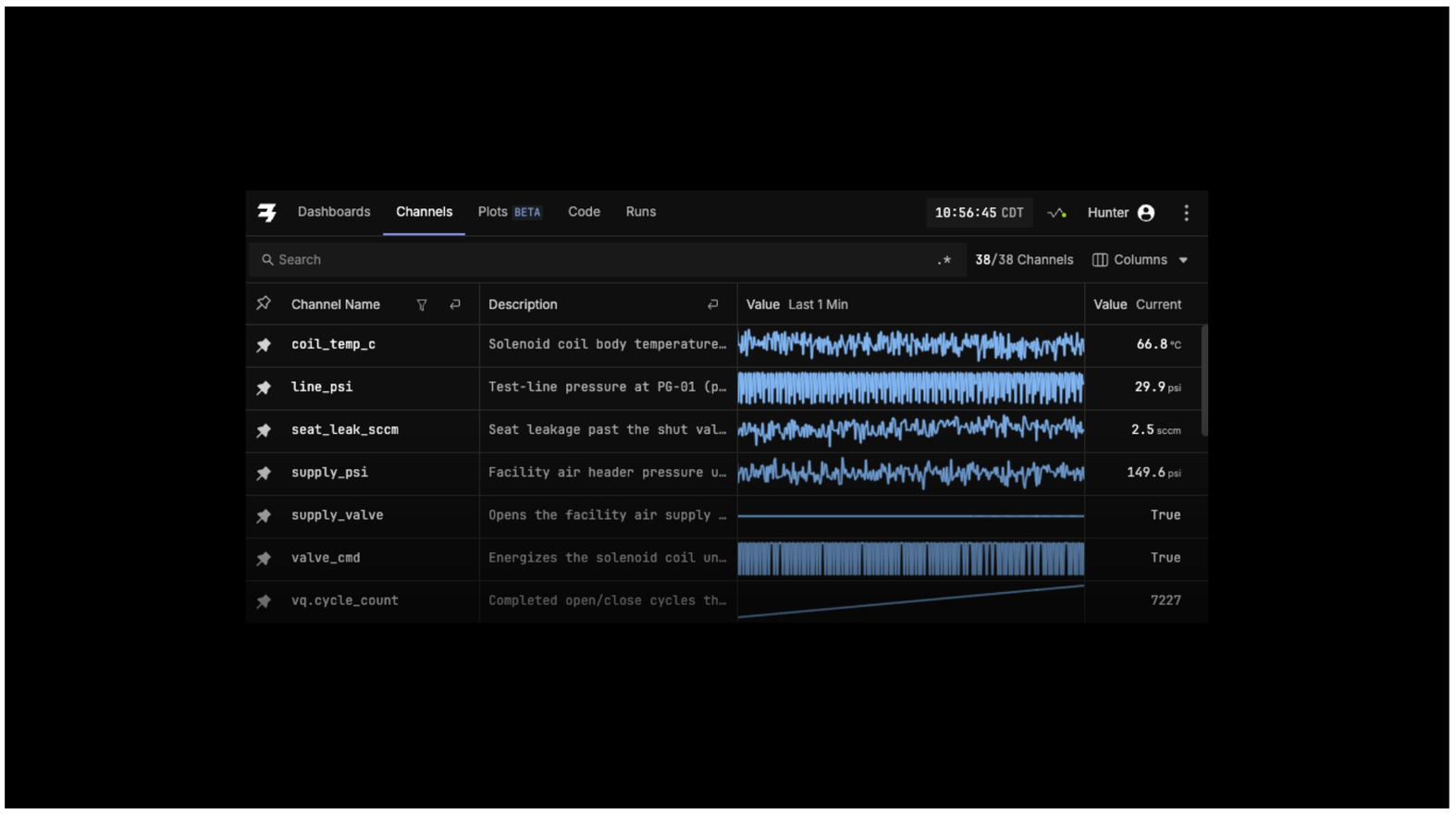
dashboard, and the test logic. The rest works out of the box.

1. The IO

In this example, an NI cDAQ chassis reads the sensors and drives the coil of the valve under test. A Beckhoff PLC controls an existing facility air supply valve.

Connect RevelTest and it scans the network, finds the cDAQ, and detects the module in each slot. The PLC is added by its address, and you name the symbols you need straight from the PLC program. Name the channels, set the scaling, click deploy, and in less than 5 minutes you can read every sensor and command every output from your laptop.

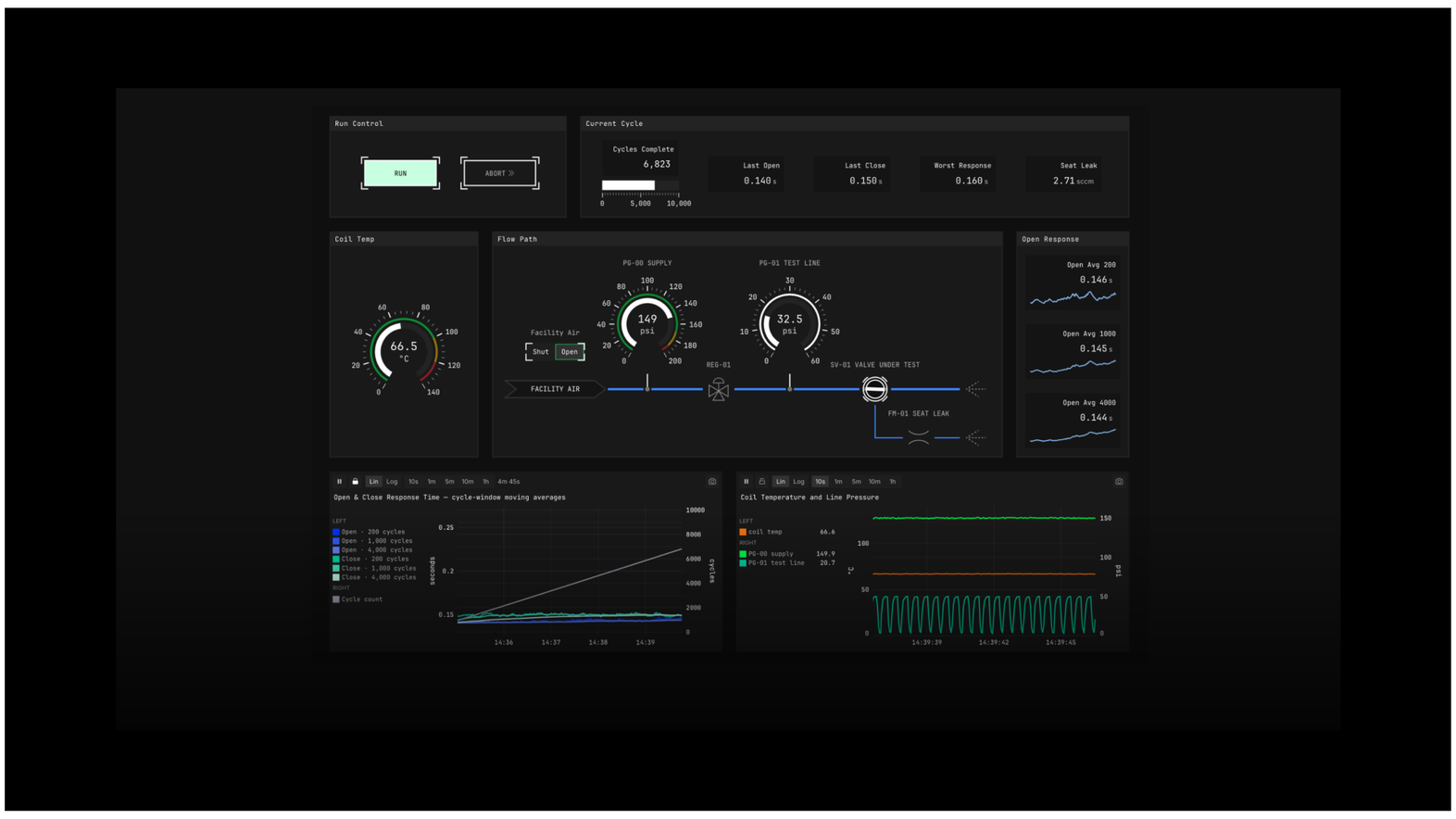
Channel	Signal	Device	Protocol
supply_psi	4-20 mA transducer	NI 9203 in a cDAQ chassis	Ethernet
line_psi	4-20 mA transducer	NI 9203	Ethernet
seat_leak_sccm	4-20 mA mass flow meter	NI 9203	Ethernet
coil_temp_c	Type-K thermocouple	NI 9213	Ethernet
valve_cmd	24 V sourcing digital out	NI 9472	Ethernet
supply_valve	PLC output	Beckhoff TwinCAT PLC	ADS over TCP



You save time and money by starting from the hardware you already own. You don’t have time to order something new; the Beckhoff PLC was already running the facility interlocks, and RevelTest reads and writes its symbols by name over ADS, with no change to the PLC program. The cDAQ was harvested from an inactive stand so you can cover the new sensors with a couple of hot-swappable c-series modules. The hardware is interchangeable, you can move a valve from the PLC to a DO card, and as long as it's still named supply_valve the rest of the system will work

exactly the same. Start with the hardware you have on hand, and have the ability to add a cRIO or LabJack if need be.

2. The dashboard



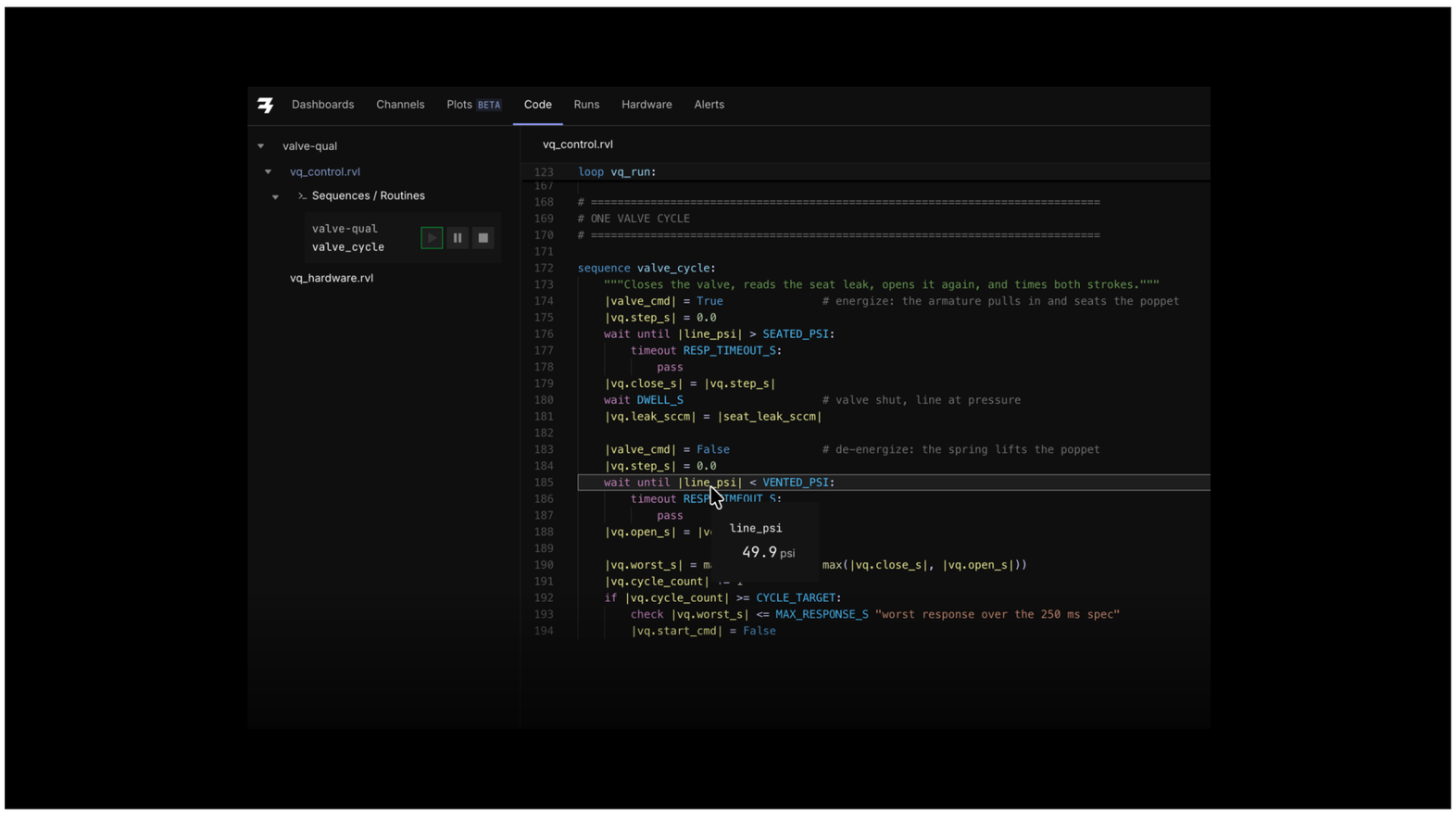
Building a dashboard is drag-and-drop. Select a node and configure its channel, all in the browser. This stand needs three regions:

- A schematic of the flow path supporting automated and manual commanding.
- Cycle count metrics for average and worst case-cycle.
- Plots with cycle time trends and system temperature and pressures.

The bottom left plot makes it easy to track cycle time trends over ten thousand cycles. This plot captures the purpose of the test that directly addressed the requirements live, rather than relying on post test analysis alone. On this valve, close response climbs about 20 ms over the first thousand cycles as the coil warms, because a hot coil draws less current and pulls in more slowly, then holds flat once the coil reaches thermal equilibrium. Open response, set by the return spring, only creeps with wear. Both stay 80 ms inside the spec. If the valve starts to slow at cycle 6,000, you know about it immediately.

Different views can use the same channels. An engineer reviewing valve performance might want a large response-time plot, while a technician operating the stand might prioritize cycle count, temperature, and abort status. Each dashboard can be arranged around those needs. At Orbital Operations, an intern built the whole stand GUI in a day.

3. The test logic in RevelCode



The valve under test is normally open, so the test is a cycle. Energize the coil and time the test line rising to pressure as the valve shuts, hold it shut and read the seat leak, de-energize and time the line venting as the valve opens, count one, and go again. Two conditions end it early: the coil reaching its insulation limit, or an operator hitting abort. That is the whole program, and it reads close to that description.

```
const CYCLE:float = 0.01              # 100 Hz: response times resolve to 10 ms

hw in  |line_psi|: float              # NI 9203, 4-20 mA transducer
# ... the other channel and constant declarations are skipped

alert coil_overtemp:
    """The coil is hotter than its insulation is rated for."""
    severity = Severity.critical
    trigger = |coil_temp_c| > MAX_COIL_TEMP_C

loop vq_run:
    # Runs every control cycle, whatever the valve is doing.
    var aborting: bool = |alert.coil_overtemp.fired| or |vq.abort_cmd|
    |vq.step_s| += CYCLE              # the clock is the control loop itself

    if |vq.start_cmd| and not aborting and |valve_cycle.status| != Status.running:
        |valve_cycle.command| = Cmd.run

    if aborting:
        |valve_cmd| = False           # de-energized, a normally-open valve vents the stand
        if |vq.start_cmd|:
            |valve_cycle.command| = Cmd.stop
            |vq.start_cmd| = False
            check |vq.cycle_count| >= CYCLE_TARGET "run aborted before the cycle target"
    ...

sequence valve_cycle:
    |valve_cmd| = True                # energize: the armature pulls in and seats the poppet
    |vq.step_s| = 0.0
    wait until |line_psi| > SEATED_PSI:
        timeout RESP_TIMEOUT_S:
            pass                      # continue: close_s records 500 ms, which fails the spec
    |vq.close_s| = |vq.step_s|
    wait DWELL_S                      # valve shut, line at pressure
    |vq.leak_sccm| = |seat_leak_sccm|

    |valve_cmd| = False               # de-energize: the spring lifts the poppet
    |vq.step_s| = 0.0
    wait until |line_psi| < VENTED_PSI:
        timeout RESP_TIMEOUT_S:
            pass                      # continue: open_s records 500 ms, which fails the spec
    |vq.open_s| = |vq.step_s|

    |vq.worst_s| = max(|vq.worst_s|, max(|vq.close_s|, |vq.open_s|))
    |vq.cycle_count| += 1
    if |vq.cycle_count| >= CYCLE_TARGET:
        check |vq.worst_s| <= MAX_RESPONSE_S "worst response over the 250 ms spec"
        |vq.start_cmd| = False
```

The logic runs at 100 Hz and the step timer counts each tick: `|vq.step_s| += CYCLE` in the state's loop block. A response time here is a count of 10ms ticks of a PTP aligned deterministic control loop, not the gap between two timestamps a script collected whenever the operating system scheduled it. A qualification that reports a response time is worth exactly as much as the clock behind it.

Acquisition rate is configured separately from the control period. Supported cDAQ inputs can capture higher-rate data for a closer look at the pressure transient after the test. Those samples provide additional detail in the recorded data; the timer shown here advances in 10 ms increments.

The loop at the top evaluates the overtemperature alert and operator abort command on every execution cycle. The abort does not wait for the sequence to reach a convenient stopping point, which matters at cycle 7,400 of 10,000.

Two check statements write the verdict into the run record. Ten thousand checks would be noise, so the run is judged on the worst response time it produced.

RevelCode compiles before deployment. The compiler rejects type errors, recursion, state machines that could transition endlessly without yielding, waits the control period cannot express, and unreachable code, so those mistakes surface at your desk rather than on the hardware.

Running it

Deploy the configuration and the runtime picks it up. Watch the first few cycles with a hand on the abort, confirm the response times land near the datasheet, then leave it running. Ten thousand cycles will take a little over an hour, and the alert watches it so a person does not have to. You can send the dashboard URL to your team so they can watch the test live.

This test showed that the old valve meets the requirements, and now you can swap it out and keep the test campaign on schedule. With RevelTest you can turn a 16 week downtime into a newly qualified old valve, and an operational stand by the end of the day.

Try it on your own hardware

If you’ve felt the pressure of keeping a test campaign on time, and want to be able to move faster with hardware, logic, and dashboards that unblock your team, reach out to Revel and see it for yourself.