PRODUCT

Flight Controller

Aerospace Defense Flight Controller Zynq Ultrascale Mpsoc

SiliconCentric | Independent Verification & Validation

I

PRODUCT

Introduction

Our client is an aerospace-and-defense supplier. Their product is a flight-control computer, still in development — engineering units built, qualification layout not yet frozen. The design is a standalone Zynq UltraScale+ MPSoC: Cortex-A53 cores host mission and telemetry functions, programmable logic hosts the bus controllers, and the internal lockstep RPU is the companion controller that owns the flight-critical control loop. We collaborated with them on this project by providing independent verification and validation of the board.

- Compute topology: one MPSoC package — A53 application cluster, FPGA fabric for MIL-STD-1553 and ARINC-429 controllers, and the internal Cortex-R5 lockstep RPU running the control loop.
- Under validation: dual-redundant IMUs on independent SPI buses, air-data sensors (pitot and static pressure) on a third SPI, a dual-redundant MIL-STD-1553 bus, four ARINC-429 channels, two RS-422 telemetry links, eight actuator driver channels, four pyro-initiator channels, GNSS on UART, and a 28 V isolated power stage.
- Program state: mid-development — engineering units on the bench, one more layout iteration available before qualification build.
- Independence was non-negotiable for the prime contractor: an assessment with no technical, managerial, or financial stake in the design, from a party that neither drew nor routed it.
- Scope stayed bounded to independent validation — no feature development, no control-law work, no flight-software authoring.
- Working material was the hardware design package: schematics, ICDs, BOM — with no access to flight-software source.

2

PRODUCT

Hardware Topology

Topologically this is a standalone MPSoC whose internal RPU fills the companion-controller role that older flight computers assigned to a discrete processor. Scope enclosed the board, its buses, and its safe-state behavior; control laws, flight software, and any DO-254/DO-178C sign-off sat outside.

- Topology classification: standalone SoC with internal RPU companion; APU and RPU exchange state through on-chip memory with IPI mailboxes, and the RPU exclusively owns the actuator and pyro output registers.
- The RPU's job: the 1 kHz control loop, sensor voting across the redundant IMUs, actuator command output, pyro arming logic, and the safe-state watchdog.
- In-scope buses: MIL-STD-1553 dual-redundant, ARINC-429 x4, RS-422 x2, SPI x3 (IMU A, IMU B, air data), UART (GNSS), plus the fabric-hosted bus controllers themselves.
- On-chip resources in scope: the A53 caches and MMU, the GIC interrupt fabric, the coherency interconnect, DDR with ECC, OCRAM and the RPU TCMs, the QSPI boot flash, and the DMA engines — validated as board-level behavior rather than assumed from silicon heritage data.
- Envelope in scope: −40 °C to +71 °C operating, 28 V input per MIL-STD-704 transient profiles, and vibration-under-monitoring at the levels the airframe spec demanded.
- Out of scope: flight software, control-law verification, and certification sign-off — we produce the hardware-integration evidence; the client's DER and certification authority judge it.
- Zynq UltraScale+ is one of the six architectures Polaris already runs on, so the OS baseline was inherited and effort focused on this board's delta rather than silicon bring-up.

3

PRODUCT

Problem Statement

Engineering units existed, the flight-software team's schedule was locked to a prime-contractor milestone, and one layout iteration remained. Every device on the board had heritage or vendor data; the board as a system had none. A flight computer that glitches an ordnance channel does not get a second chance in flight test.

- Proven in isolation, never as a board: the 1553 transceivers, ARINC drivers, IMUs, and MPSoC each carried qualification-grade vendor data, but nothing had exercised this board's bus routing, its isolated 28 V power stage, its fabric controller timing, and its redundant sensor paths as one system.
- Firmware coverage is feature-shaped, not board-shaped: flight software exercises the buses and sensors the control laws need — the second RS-422 link, two of four ARINC channels, the pyro built-in-test path, and the 1553 bus-B failover sat outside every planned software test.
- No stress, no extremes, no duration: no −40 °C cold-start campaign, no 71 °C soak under full bus load, no MIL-STD-704 transient and brown-out sequence, no multi-day burn-in, no vibration run with continuous electrical monitoring — the exact conditions the airframe guarantees this board will see.
- Nothing runs alongside development: hardware validation was a qualification-phase gate, so any board fault would first appear during flight-software integration — the costliest place in the program to debug it.
- DO-254 expects hardware verification evidence with requirement traceability; DO-178C expects the software team to build on hardware whose behavior is characterized, not assumed.
- The pyro and actuator stages demanded proof of glitch-free behavior through every power and configuration transition — an FPGA-based design adds a bitstream-load window that pure-MCU heritage designs never had.
- A single field failure here is a lost vehicle and a grounded fleet.

PRODUCT

Customer Requirements

4

The client's requirements were direct: validate the whole board, under stated limits, with evidence structured for their DO-254 verification case — assessor-ready, not engineer-ready.

- Operating temperature: the board shall operate from −40 °C to +71 °C, including cold start at −40 °C.
- Supply voltage: 28 V nominal per MIL-STD-704, steady-state window 22 V to 29 V. The board shall survive transient spikes to 80 V for 50 ms without damage; over-voltage protection shall clamp above 32 V continuous.
- Brown-out: the board shall ride through input sag to 18 V per MIL-STD-704 transient profiles; below 18 V it shall enter safe state with actuator and pyro outputs inhibited and no record loss.
- Over-current: each actuator and pyro channel shall have independent over-current protection tripping within 5 ms; a fault on one channel shall not disturb any other channel. Pyro outputs shall be glitch-free through every power and FPGA configuration transition.
- Humidity: the board shall operate at up to 95% RH non-condensing and shall not be exposed beyond that.
- Full-board functional validation: all A53 cores, the lockstep RPU, every fabric bus controller, all DDR regions, both IMU paths, air data, all four ARINC channels, both 1553 buses, both RS-422 links, every actuator and pyro channel, and the power stage — exercised as one system.
- Coverage independent of firmware scope: every subsystem validated regardless of what the control laws touch, through a HAL built from the design package with no flight-software dependency.
- Stress matrix: −40 °C to +71 °C sweep with cold-start cycling, MIL-STD-704 voltage transients and brown-out, 96-hour burn-in under full bus traffic, vibration with continuous monitoring, and bus-failover injection on 1553 and the redundant IMUs.
- Continuous validation alongside development: CI integration so every flight-software candidate revalidates on the engineering units in parallel with development.
- Regulatory evidence: timestamped, operator-attributed, version-pinned records with bidirectional requirement-test-result traceability, formatted for the DO-254 verification case.
- A web console: the client's test conductors run campaigns without knowing the validation stack, the same interface as every other program.
- An immutable, append-only evidence store forming a defensible chain of custody across engineering and qualification units.
- Setup in minutes: bench procedures that take a conductor from cabling to running campaign repeatably.

PRODUCT

Architecture

The validation OS, HAL, and client run on the flight-control board; the server, dashboard, and SDK run on an external x86 host. Execution is isolated from data processing, and the MPSoC keeps its resources for representative behavior — which here meant the 1 kHz loop timing measurements were taken with the fabric and RPU loaded the way flight would load them.

- Polaris OS (the validation OS) — a reproducible Linux image for the A53 side booting to a known state, giving the program a hardware baseline that cannot drift between benches.
- HAL — the one board-specific layer, exercising the fabric bus controllers, all three SPI sensor buses, the RS-422 links, actuator and pyro built-in-test paths, and the power-stage telemetry behind a uniform interface that keeps the aerospace suites portable to the qualification build.
- Client — the on-target agent driving tests through the HAL; gRPC on Linux, FlatBuffers to the RPU's FreeRTOS image; three timestamps per event (origin, relay, server); load small enough that loop-timing evidence reflects flight configuration.
- Server — orchestration, scheduling, and the append-only records store on the x86 host, running two engineering units in parallel and hosting the dashboard and API.
- Dashboard — the test conductor's console: provision a unit, configure and launch, watch live bus, rail, and loop-timing telemetry, review time series after; every state-changing action operator-attributed, which the program office required.
- Python SDK — automation and CI; drives everything the console does; the C/C++ bindings saw use here for a timing-critical injection harness.
- The OS and HAL are the two board-specific layers and were heavily modified for this board and this SoC; the client, server, dashboard, and Python SDK were configured to the client's requirements, but their core structure and working style remain the same across every board and every industry.

6

PRODUCT

Design

Design was driven by the 1 kHz hard deadline, the −40 °C cold-start requirement, the ordnance-safety regime around the pyro channels, and a traceability standard where every record must survive a certification review.

- Validation OS design — secure boot through the full MPSoC chain, read-only rootfs, A/B partitions with rollback; the deliberate trade-off is boot time (bitstream authentication costs seconds), accepted because an unauthenticated test image is inadmissible evidence in this vertical. The RPU runs FreeRTOS in lockstep with fault-injection hooks.
- HAL design — partitioned by subsystem, every function mapped to a schematic sheet and ICD paragraph; bus controllers and sensor paths are interrupt-driven because polling would blur the failover-timing measurements. A babbling 1553 remote terminal can storm the fabric interrupts, so hardware rate limiting is built into the capture block.
- Client design — origin timestamps latched in fabric registers at the event, not at software pickup, so failover sequences order correctly at microsecond scale; on power-fail the client abandons aggregation and preserves raw records, trading analysis convenience for complete evidence.
- Server design — three-timestamp causality reconstructs bus-failover chains across units; retention sized for 96-hour burn-ins on two units; the store runs on an air-gapped program network, so the cross-site sync feature was configured off.
- Dashboard design — loop-period histograms, 1553 word-error counters, per-channel ARINC status, rail telemetry, and pyro built-in-test state are first-class views; run health and drop counts on every screen, because a conductor must see evidence gaps immediately.
- SDK design — the client scripted transient-injection campaigns and the nightly acceptance gate; the direct API path served bench-level single-bus checks, at the stated cost that such runs generate no records.
- OS and HAL design was board-specific and heavily reworked; the four upper components were configured to the program's workflow while their structure and working style stayed constant.

7

PRODUCT

Implementation

The build effort went into the OS delta, a HAL written paragraph-by-paragraph against the ICDs, and the aerospace campaigns; client, server, dashboard, and SDK were configured, not constructed.

- Validation OS — Yocto image for the MPSoC: inherited boot chain (FSBL, ATF, U-Boot) with program-specific signing keys, board device tree covering the fabric controllers and sensor buses, minimal kernel; OTA with A/B rollback; authenticated boot to validation-ready in under 40 seconds; per-build SBOM with CVE tracking; RPU FreeRTOS image built in the same tree.
- HAL — implemented subsystem by subsystem and bench-proven as written: register interfaces to the fabric 1553 and ARINC controllers, interrupt and DMA paths for IMU capture, drive of the actuator and pyro built-in-test loops, MIL-STD-704 transient hooks into the programmable supply; the design-package pass surfaced a finding of its own — the ICD specified ARINC channel 3 as high-speed while the schematic's RC filtering only supported low-speed, a document conflict caught before it could present as an intermittent bus fault.
- Client — C++ daemon plus fabric timestamp capture, event and polling paths, graceful shutdown without record loss, gRPC upstream, FlatBuffers to the RPU.
- Server — telemetry receiver, versioned schema, scheduler, REST API, dashboard hosting, two-unit parallelism, air-gapped deployment.
- Dashboard — unit provisioning, campaign configuration, live bus and loop-timing streams, time-series review, export templated for the DO-254 verification case, operator attribution throughout.
- SDK — Python bindings, C bindings for the injection harness, example scripts, and CI wiring gating every flight-software candidate.
- Board-specific: OS delta and HAL. Configuration: client transports, server deployment posture, dashboard views and export, SDK harness glue.

PRODUCT

Test - IV&V

8

The strategy ran stack-first: prove the platform, prove the OS/HAL on this board, then run the board through one-shot, long-running, event-based, and monitoring campaigns, with all dashboard-launched work committed to the append-only store.

- Stack self-verification: 100% test coverage of the client, server, dashboard, and Python SDK — proven before it produced evidence about the flight computer.
- OS/HAL verification: A/B boot and rollback exercised, OTA path verified end to end, the secure boot chain checked on accept and reject paths including a tampered bitstream.
- Bench setup: unit cabled to the host, programmable 28 V supply and chamber configured, provisioning from the dashboard — a written conductor procedure, repeatable in under thirty minutes.
- Functional campaign: 1553 bus-A/bus-B suites with failover injection, per-channel ARINC word suites, RS-422 loopback and margin, dual-IMU cross-comparison, air-data channel verification, actuator and pyro built-in-test suites, GNSS acquisition checks.
- CPU and memory campaign: cache and coherency stress across the A53 cluster, MMU and GIC interrupt-latency measurement under full bus load, DDR pattern and ECC single/double-bit error injection, OCRAM/TCM and QSPI margin checks, and DMA integrity under concurrent sensor capture.
- Clock, reset, and power-domain campaign: PLL lock and clock-tree behavior verified across the temperature range, per-domain warm and cold reset sequencing exercised, and watchdog-expiry recovery into safe state confirmed with actuator and pyro outputs inhibited throughout.
- Debug and trace posture: CoreSight trace and PMU counters on engineering units anchored the loop-timing and interrupt-latency evidence; JTAG lockdown via eFuse configuration was verified on the qualification-intent path, since an open debug port on an ordnance-carrying board is itself a finding.
- One-shot tests as the acceptance gate: one pass/fail per subsystem per software candidate.
- Long-running: the 96-hour full-traffic burn-in with snapshots caught IMU-B cross-axis disagreement drifting with temperature, isolated to a rail-ripple sensitivity, corrected with added filtering in the final layout.
- Event-based: silent runs recording only on bus errors, loop-deadline misses, rail excursions, or safe-state transitions — this mode, with a scope on the outputs, documented the program's main defect: pyro channels 2 and 3 glitching high for 1.8 µs during FPGA bitstream load, a configuration-window hazard on an ordnance path, eliminated by adding hardware inhibits gated on DONE.
- Monitoring: passive telemetry — rails, temperatures, loop-period statistics, bus error counters — under a flight-representative workload binary through thermal and vibration runs.
- Instrumentation correlation: oscilloscope and logic-analyzer captures anchored the pyro glitch, the cold-start behavior at −40 °C (one unit required two boot attempts below −35 °C, traced to a slow-starting oscillator), and the MIL-STD-704 transient responses to the records.
- Recording rule: every run launched from the dashboard is recorded, always, into the append-only store; the Python SDK can run 100% of what the dashboard can run — module, peripheral, or system level — for quick independent checks, and SDK-launched runs of this kind are not written to the records store. The store is the evidence; the SDK is the bench tool.
- Traceability: every requirement traced bidirectionally to tests and records, schemas version-pinned, runs reproducible for the certification review.

9

PRODUCT

Deliverables

Delivery comprised the full validation platform fitted to the flight computer — source, records, and procedures, with no lock-in.

- Validation OS image with the complete Yocto build system, signed boot-chain recipes, and per-build SBOM.
- Board-specific BSP and HAL source for every subsystem, maintained against the design package into the qualification layout.
- Compiled client for the A53 side plus the RPU FreeRTOS build and fabric timestamp blocks.
- Server and dashboard as deployment-ready containers configured for the air-gapped program network.
- Complete suites — functional, failover, thermal, MIL-STD-704 transient, burn-in, vibration-monitoring — with recorded engineering-unit baselines.
- Python SDK, C injection bindings, documentation, example scripts, and the CI integration gating every flight-software candidate.
- The append-only records store delivered as program property — the hardware-integration evidence for the DO-254 case.
- The independent validation report: the pyro configuration-window glitch, the IMU-B ripple sensitivity, the cold-start oscillator finding, the ARINC ICD conflict, metrics, pass/fail, coverage.
- Conductor bench procedures for acceptance and re-runs; full documentation; no lock-in.

10

PRODUCT

Conclusion

The flight computer went into its qualification layout with its worst behaviors already found and designed out — an ordnance channel that glitched during configuration, a sensor that drifted with rail ripple, an oscillator that hesitated in the cold. Each was corrected in the remaining layout iteration rather than discovered in flight test.

- The pyro glitch alone justified the engagement: found on the bench with a scope for the cost of a hardware inhibit, instead of during a flight-test anomaly investigation with the fleet grounded — and every finding is backed by traceable, conductor-attributed records the certification review can use.
- Validation now runs continuously in the program's CI against every flight-software drop, and the suites carry to the qualification units with only the HAL delta re-verified. The caveat stands — we produce the evidence; the client's DER and certification authority judge it.

II

PRODUCT

Lessons Learned

The engagement hardened our rule set for FPGA-based safety controllers: configuration-window behavior is now a first-class campaign, not a corner case.

- Every output stage on an FPGA-fabric board gets a bitstream-load glitch campaign in the first bench week — DO-254 discusses verification of intended function at length, but the hazard here lived in the 200 ms before the design existed, and only output-stage scoping during configuration finds it.
- The APU–RPU boundary needs its boot-order contract tested explicitly: the RPU had to own the pyro registers before the A53 cluster came up, and we found one reset ordering where U-Boot briefly tri-stated the fabric pins the RPU believed it held — on future MPSoC engagements the reset-and-ownership matrix is validated before any functional suite runs.

CONTACT

SiliconCentric

EMAIL sandesh@siliconcentric.com

PHONE (669) 356-1998

WEB https://www.siliconcentric.com

ADDRESS San Jose, CA, USA