Several rpfpga instruments together

Two or three Red Pitaya boards running different designs, connected and controlled as one experiment

Rok Žitko
Jozef Stefan Institute, Ljubljana, Slovenia
rok.zitko@ijs.si

November 10, 2026

Abstract

A Red Pitaya runs one rpfpga design at a time: a waveform generator, a lock-in amplifier, a lock box, a time tagger, a photon counter, an analyser. A laboratory experiment usually needs several of them at once: a sequencer that drives the experiment, a detector that reads it out, a lock that holds a laser, a reference that keeps time. This guide is about using two or three boards, each running a different design, as one instrument: which of their pins may be connected to which, with what levels and cables; how time and frequency relate between boards with their own crystals; how fast one board can react to another; how to control and record several boards from one program; a catalogue of combinations with their wiring and code; and how to test a combination before trusting it. Each design's own manual describes the design; this guide describes the connections between them. Combinations marked Tested were wired and measured on the boards used here; those marked Not tested follow from the designs' documented interfaces and have not been tried together.

Contents

1 Why several boards **3**
1.1 The designs and their roles . . . 3
1.2 Ground rules . . . 3
1.3 What is tested . . . 4

2 The instruments' interfaces **4**
2.1 awg: the sequencer . . . 4
2.2 counter: gated photon counting and decisions . . . 4
2.3 spin: a pulsed RF transceiver . . . 5
2.4 boxcar: a pump–probe boxcar averager . . . 5
2.5 timetagger: picosecond time tags . . . 5
2.6 feedback: the lock box . . . 6
2.7 lockin: lock-in detection . . . 6
2.8 synth: tones and frequency sequences . . . 6
2.9 atomic_clock: a 10 MHz and PPS source . . . 6
2.10 ajku, mult_add, analog_echo . . . 7
2.11 References and their uses . . . 7

3 Connecting boards **7**
3.1 Pins: outputs to inputs . . . 7
3.2 Levels, cables and crosstalk . . . 7
3.3 Crosstalk in a bundle of jumpers . . . 8
3.4 Analog connections . . . 8
3.5 States at power-up, loading and SAFE . . . 9

4 Time and frequency across boards **9**
4.1 Every board has its own clock . . . 9
4.2 A common reference . . . 9
4.3 Latencies between boards . . . 10

5 Several boards from one program **10**
5.1 One client per board . . . 10
5.2 Software is slow, wires are fast . . . 10
5.3 Recording from several boards . . . 11

6 Two boards **11**
6.1 awg + counter: a sequencer and a photon counter . . . 11
6.2 awg + timetagger: sequences with picosecond tags . . . 12
6.3 awg + boxcar: an emulated laser for a pump–probe instrument . . . 13
6.4 counter + timetagger: counting and tagging the same photons . . . 14
6.5 counter + feedback: holding an experiment at its maximum count rate . . . 14
6.6 synth + counter: frequency scans counted per point . . . 14
6.7 A reference and a timekeeper: atomic_clock or a GNSS oscillator . . . 15
6.8 awg + feedback: a sequence and a lock . . . 15
6.9 awg or synth + lockin: excitation and detection . . . 15
6.10 Analog chains: mult_add and analog_echo . . . 16

7 Three boards **16**
7.1 awg + counter + timetagger: an NV experiment with every photon kept . . . 16
7.2 awg + counter + feedback: an ion or atom experiment with its laser lock . . . 16
7.3 A common reference for all boards . . . 17

8 Testing a combination **17**
8.1 1. Static levels . . . 17
8.2 2. Exact counts . . . 17
8.3 2a. Crosstalk . . . 17
8.4 2b. Random patterns with exact expectations . . . 17
8.5 3. Timing from the instruments themselves . . . 18
8.6 4. An oscilloscope . . . 18
8.7 5. Receipts . . . 18

A The E1 pins of every design **18**

1 Why several boards

Each rpfpga design turns the Red Pitaya into one instrument, and the board's FPGA is full with it: there is no room for a second design beside the first. An experiment that needs a pulse sequencer and a photon counter, or a lock box and a time tagger, needs two boards. Red Pitayas are cheap enough for that, and the designs share their framework (the service, the Python interface, the board calibration, the SD card image), so a second or third board costs little effort: load the design, connect the wires, and control all of them from one program.

1.1 The designs and their roles

Role	Design	In a combination
sequencing, sources	`awg`	the experiment's pulse sequencer: two analog outputs, eight digital outputs (patterns), markers, triggers and event inputs that let a sequence branch
	`synth`	tones and timed frequency sequences: radio-frequency drives, sweeps with markers
	`atomic_clock`	a 10 MHz and 1 PPS source with controlled phase noise: a reference, or a clock under test
	`spin`	a pulsed RF transceiver: phase-cycled pulses and their echoes averaged on the board, single-shot I/Q decisions on a pin and in its own pulse program (NMR, NQR, ESR and qubit readout at an IF)
detection, analysis	`counter`	photon counting in gates, per-point sums, decisions on a pin, coincidences
	`boxcar`	a boxcar averager for pump–probe: gated analog shots sorted by category, $\Delta T/T$ per delay point, electronic delay scans on its pulse outputs, averaged traces and raw segments
	`timetagger`	picosecond time tags of every edge, streamed to a PC; markers; a coincidence engine
	`lockin`	lock-in detection of up to four frequencies, PIDs, a frequency counter
	`ajku`	statistics, histograms and spectra of optical I/Q (homodyne) signals
control	`feedback`	a lock box: fast and slow PIDs, lock acquisition, a TTL reference input, status pins
analog processing	`mult_add`, `analog_echo`	sums, products and functions of the inputs; gain and offset

Table 1: The rpfpga designs by their role in a combination.

Typical combinations pair a source with a detector (awg with counter or timetagger), a detector with a controller (counter's rate with feedback's lock), or a reference with anything that keeps time (atomic_clock or a GNSS-disciplined oscillator with timetagger, counter or lockin). Section 6 goes through them; section 7 through setups of three boards.

1.2 Ground rules

Before connecting two boards

1. **Load both designs first, then connect.** Which pins are outputs depends on the design (appendix A); a pin that one design reads may be driven by another.
2. **Connect outputs only to inputs.** Never wire two outputs together; check every wire against the pin map.
3. **Join the grounds** (E1 pins 25, 26, or the SMA shells) before any signal.
4. **Check each wire with static levels**: drive one end high and low and read the other (section 8). Two of seven jumpers were misplaced in the first try of the session of section 6.1; the check found both

before anything was measured.

5. **Use hardware triggers for timing.** Commands over the network take milliseconds and vary; everything that must happen within a microsecond goes through wires (section 5).

1.3 What is tested

Each combination in this guide is marked Tested (date) if it was wired and measured on the boards used here, or Not tested if it follows from the designs' documented interfaces but has not been tried together. A Not tested combination is a plan, not a result: test it as in section 8 before relying on it.

2 The instruments' interfaces

This section lists, design by design, what each one offers another board: its pins on the extension connector E1, its SMA inputs and outputs, its triggers, markers and references, the latencies measured so far, and the client methods a combined program calls. Each design's README and manual have the details; appendix A has every E1 pin of every design in one table.

2.1 awg: the sequencer

- **E1 inputs** (pull-downs): TRIG A on DIO0_P and TRIG B on DIO3_P, timed to 1 ns; EVENT 0–3 on DIO4_P–DIO7_P, sampled on the 8 ns grid, which start, gate, key or make the sequencer jump (rising, falling or both edges). A jump event takes part from 8 samples after it arrives and the jump comes at least 16 samples after it; an event before the waiting step's first point is not taken.
- **E1 outputs**: markers 1, 2 on DIO1_P, DIO2_P (driven always, low by default) and the digital outputs D0–D7 on DIO0_N–DIO7_N (high impedance until enabled, and after `SAFE`): one 8-bit word per sample from up to eight programs (patterns, counts, PRBS), started at an apply, by an output's start, or by sequence steps; their delay against the analog outputs is set in 8 ns steps (D0 and the markers land 3.3 ns after the analog edge on the calibrated board).
- **SMA**: OUT1, OUT2 (waveforms, sequences); IN1, IN2 as threshold triggers or modulation sources.
- **Measured**: TRIG A to OUT1 377.9 ns (delay 0); EVENT0 to a digital output 363 ns (aligned with the analog outputs) or 251 ns (the shortest digital delay); with timetagger: a digital output's period jitter about 21 ps rms, D1 20 ps after D0, an 8 ns word 8.79 ns wide (2026-11-05).
- **Client** `AwgClient`: `set_output`, `play`, `save_sequence`, `save_pattern`, `set_digital`, `trigger`, `hardware` (the last start times).

2.2 counter: gated photon counting and decisions

- **E1 inputs** (pull-downs, 1 ns): C1–C8 on DIO0_P–DIO7_P; TRIG, GATE, NEXT, SEQ on DIO0_N–DIO3_N.
- **E1 outputs**: O1–O4 on DIO4_N–DIO7_N (high impedance until enabled, and after `SAFE`; 2 ns resolution): decisions, gate windows, TRIG echoes, a channel's echo, heralds, the rate alarm, patterns.
- **SMA**: IN1, IN2 are the analog count channels C9, C10 (discriminators in volts); OUT1, OUT2 rate monitors (a voltage proportional to a count rate) or test pulses.
- **Measured**: a decision 101–109 ns after its gate's end (105 ns on average); a channel's echo 92.5 ns after its edge, a herald 121.5 ns + W after the cluster's first event; 2 million shots per second counted exactly from awg's patterns; SHOT records to a PC without loss up to 1.5 million per second (2026-11-05).

- **Client** `CounterClient`: `run`, `stop`, `results`, `decisions`, `coincidences`, `rates`, `measure_latency`; `RecordStream` (port 5026).

2.3 spin: a pulsed RF transceiver

- **E1 inputs** (pull-downs, 1 ns): TRIG on DIO0_P (a WAIT continues 192 ns after the edge, the RF phase referred to it within 1 ns), COND A and COND B on DIO1_P, DIO2_P (levels for branches, read 23 clocks after the branch's start).
- **E1 outputs**: D1–D8 on DIO0_N–DIO7_N (high impedance until enabled, and after `SAFE`; 2 ns delays): TTL bits of the pulse program, power-amplifier gates with lead and lag, the capture window, SHOT and run markers, a window's single-shot state.
- **SMA**: OUT1, OUT2 the transmitters (two real channels or an I/Q pair); IN1, IN2 the receivers (or an I/Q pair). **E2**: AO0–AO3 set by the program (field sweeps, attenuators).
- **Measured**: the RF phase against an asynchronous TRIG 0.28 ns rms; a pulse at the connectors 377–384 ns after a TRIG edge; branch decisions to the clock (2026-11-07). From a window's last sample to a chosen pulse D + 28 clocks, to a state output D + 5.5 clocks plus the board's pin delay (512 ns and 332 ns nominal; on the test board 488 ns as programmed and 281 ns, measured with an oscilloscope, 2026-11-08).
- **Client** `SpinClient`: `run`, `stop`, `trigger`, `window_average`, `results` (RESULT records in volts), `capture`; the templates in `spin.experiments`.

2.4 boxcar: a pump–probe boxcar averager

- **E1 inputs** (pull-downs, 1 ns): TRIG on DIO0_P (a laser's sync: the shot's time to 1 ns), CHOP on DIO1_P (a chopper's reference: the shot's category, with a guard band), ARM on DIO2_P (gates the shots, or NEXT: the next point, a delay stage's "in position"), AUX on DIO3_P (spare).
- **E1 outputs** (high impedance until enabled, and after `SAFE`; 2 ns steps): P1–P8 on DIO0_N–DIO7_N (a pulse per shot at a delay, per category, with a delay table per point: pump and probe triggers; or levels: markers, the next shot's category for a modulator), M1–M4 on DIO4_P–DIO7_P (markers: shot, point, sweep, category).
- **SMA**: IN1, IN2 the detectors (signal, reference); OUT1, OUT2 the boxcar outputs (an average per category, within lockable limits) or the test source (an emulated laser's detector pulses).
- **Measured**: through a cable loopback ($\Delta T/T$ within 0.5 % of the emulated sample's, every shot's gates at 2.016 MHz) and with awg as an asynchronous laser (section 6.3: every shot taken to 1.95 MHz, $\Delta T/T$ within 4×10^{-4} of awg's, awg's pulse 392 ns after P1), 2026-11-09. With an oscilloscope (2026-11-10): a pulse output's edge 21.7 ns plus its delay after the TTL edge at TRIG (±0.6 ns rms; the shortest on-time delay 88 ns), level outputs 74 ns; OUT1 changes 0.90 µs and OUT2 1.08 µs after the end of the shot's last gate at IN.
- **Client** `BoxcarClient`: `run`, `stop`, `next_point`, `points` (the service's POINT totals: $\Delta T/T$ per point), `trace_block`, `segments`, `records`.

2.5 timetagger: picosecond time tags

- **E1 inputs**: channels 1–8 on DIO0_P–DIO7_P (pull-downs, about 10 ps rms); eight markers on DIO0_N–DIO7_N, timed to 1 ns (pulses under 4 ns count as glitches). It drives no E1 pin.
- **SATA**: S2 two more channels (1.8 V differential); S1 its test signal, driven only when enabled.
- **SMA**: IN2 a 10 MHz reference for its phase meter; OUT1, OUT2 never driven.

- **Measured with awg** (2026-11-05): awg's sweeps sorted exactly by its markers (2.4 million points); markers 16.008 ns after a channel for a 16 ns offset, 290 ps rms; crosstalk from awg's simultaneous edges fires markers whose lines idle low (section 3.3); the coincidence engine, its marker gate and the conditional filter exact on awg's patterns.
- **Client** `TimeTaggerClient`: `start` (now, at a sample, or armed on a channel), `stop`, `acquisition`, `now`, `counters`; `RecordStream` (port 5026).

2.6 feedback: the lock box

- **E1 inputs**: a TTL reference on DIO0_P (edges timed to 1 ns; its PLL locks an oscillator to an external clock, also at N/M of its frequency) and a capture trigger on DIO3_P.
- **E1 outputs**: a sync square wave on DIO1_P (2 ns edges, low while disabled), and four **status pins on DIO0_N–DIO3_N, driven low after loading and `SAFE`** (locked, scanning, a PID's sign, the frame, clipping, ...).
- **SMA**: the fast PIDs' inputs IN1, IN2 and outputs OUT1, OUT2; **E2**: slow outputs AO0–AO3 (PWM, 0–1.8 V, to be low-pass filtered) and inputs AI0–AI3 (0–7 V).
- **Measured**: its sync into its own TTL input 107.5 ns; a PLL on a 5 MHz TTL clock within 3×10^{-5} Hz (2026-10-26); locked to awg's D0 from another board, 290 ps rms per reading (the 1 ns quantization), holdover within 3 Hz with a short loss time (2026-11-05).
- **Client** `FeedbackClient`: `set_ttl`, `lock_ttl`, `set_sync`, `set_dio` (the status pins' sources), `capture`, `count_frequency`.

2.7 lockin: lock-in detection

- **E1 input**: a capture trigger on DIO3_P (pull-down; not tested on the board). It drives no E1 pin.
- **SMA**: four oscillators on OUT1, OUT2; demodulators on IN1, IN2; PLLs lock to a signal on an input; a frequency counter (within 0.1 ppm of the board's clock).
- **Client** `LockinClient`: `set_frequency`, `set_output`, `set_demodulator`, `lock_pll`, `capture`, `set_stream`, `record`; an SR86x-compatible port (TCP 23).

2.8 synth: tones and frequency sequences

- **E1**: a trigger input on DIO0_P (pull-down; 0–8 ns of jitter against the sample clock) and a marker output on DIO1_P (on the first sample of a sequence step).
- **SMA**: 16 tones per output; IN1, IN2 as amplitude-modulation sources, captures, tone measurements.
- **Measured**: its marker back into its trigger 248 ns after the step's sample (2026-10-22).
- **Client** `SynthClient`: `apply`, `play` (a sequence that can wait for the trigger), `trigger`, `zero_time`, `counter`.

2.9 atomic_clock: a 10 MHz and PPS source

- **E1 outputs**: the simulated clock's 1 PPS on DIO1_P and the board's reference 1 PPS on DIO2_P (3.3 V, 20 µs by default), optionally aligned to UTC (to milliseconds, from NTP).
- **SMA**: OUT1 a 10 MHz with programmable phase, frequency offset and noise; OUT2 a clean 10 MHz (exactly 2/25 of the board's clock); IN1, IN2 a 10 MHz phase meter.
- **Client** `AtomicClockClient`: `set_phase`, `set_frequency_offset`, `set_pps`, `align_utc`, `measure`.

2.10 ajku, mult_add, analog_echo

- **ajku** reads a PPS on DIO0_P (the sample index of each pulse) and starts captures at an exact sample index; IN1, IN2 are its I and Q inputs; it drives no E1 pin. Client: `sample_index`, `capture(start_index=...)`, `statistics`.
- **mult_add** computes weighted sums of 13 functions of IN1 and IN2 on each output (456 ns, or 608 ns with X/Y); **analog_echo** applies a gain and an offset (32 ns in the FPGA). Neither uses E1.

2.11 References and their uses

Design	External time or frequency reference
atomic_clock	*makes* a 10 MHz (OUT1, OUT2) and PPS (DIO1_P, DIO2_P); measures an external 10 MHz on IN1, IN2
timetagger	10 MHz on IN2 (phase meter); a tagged clock or PPS on any channel; an acquisition armed on a channel
counter	counts a 10 MHz on IN2 (C10): the board's crystal against it
feedback	locks an oscillator to a TTL clock on DIO0_P (N/M); a sync output on DIO1_P
lockin	a PLL to a signal on IN1 or IN2
ajku	a PPS on DIO0_P: the board's sample rate from the indices of its pulses
awg, synth, analog_echo, mult_add	none: frequencies relative to the board's own crystal

Table 2: References. No design takes an external sample clock: every board runs on its own crystal.

3 Connecting boards

3.1 Pins: outputs to inputs

All E1 pins of all designs are 3.3 V LVCMOS: an output of one board drives an input of another directly, with a short wire or a coax cable, and a common ground. The one rule is the pin map (appendix A): connect an **output** only to **inputs**. Two outputs wired together fight: one drives high while the other drives low, and the pins' drivers carry the short circuit.

No straight ribbon cables between boards

A flat cable from one E1 header to another connects equal pins: DIO4_N to DIO4_N, and so on. Many of them are outputs on both sides: awg's D4–D7 face counter's O1–O4, awg's markers face synth's marker, feedback's sync or atomic_clock's PPS on DIO1_P, feedback's status pins face awg's D0–D3. Wire each signal individually, from an output to the input it is meant for.

Pins that a design leaves high impedance until enabled (awg's D0–D7, counter's O1–O4) are safe to wire before they are enabled; pins that a design drives from the moment it loads (awg's markers, feedback's sync and status pins, synth's marker, atomic_clock's PPS) are outputs as soon as the design is loaded. Load the designs first, then wire, then enable outputs, then check the wires with static levels (section 8).

3.2 Levels, cables and crosstalk

The E1 outputs drive high-impedance inputs: another board's pins (with their pull-downs), an oscilloscope probe, a logic gate. A 50 Ω load pulls their high level far below a logic 1; feedback's sync output is the exception, documented to drive 50 Ω. Wire lengths:

- **up to about 20 cm** of jumper wire: measured to work for awg's digital outputs into counter's inputs and counter's outputs into awg's event input, at 8 ns pulses and 125 MHz patterns, with no lost edge in millions (section 6.1);
- **longer**: coax, with a series resistor of about 33–47 Ω at the output (source termination) and nothing else on the line;
- **crosstalk**: see below.

3.3 Crosstalk in a bundle of jumpers

Jumper wires side by side couple: an edge on one wire induces a short spike on its neighbours, and an input that samples every nanosecond sees the spike as a pulse. Two measurements here:

- 8 ns pulses on the wire beside counter's TRIG made 2 false TRIG edges per million (awg + counter); TRIG's hold-off at 450 ns, longer than the coupled spikes' delay from the real TRIG, removed them;
- two of awg's outputs rising *in the same 8 ns word* fired timetagger's markers on two other wires of the bundle: one of them at every edge, the other at 75–85 % of them, with hundreds of thousands of glitches per second (awg + timetagger). One output alone, or the two a word apart, fired nothing. A second ground wire between the boards changed nothing; with the quiet lines not driven at all the spikes remained; with the quiet lines driven high they vanished. So it is coupling between the wires, adding up when several edges coincide, and it creates false *rising* edges on lines that idle low;
- lines that idle high are not immune either. With counter's TRIG and SEQ idling high and triggering on falling edges, random patterns on the two pulse wires made a false falling edge about 1 ns after both rose in the same word, at one or two places of each 32 768-word pattern, in 1–4 % of its passes (awg + counter, evening). A false SEQ showed as bad sweeps; a false TRIG 463 ns before a real one was hidden by TRIG's hold-off and found only by the TRIG times in counter's records. With no two outputs changing in the same word nothing was found in 44 million shots.

Remedies: keep trigger and marker lines away from fast pulse lines (separate bundles, or a ground wire between them), twist each signal with its own ground wire, use coax for anything fast; never let several outputs change in the same word next to a trigger line; use a trigger's hold-off against ringing after a real edge (it cannot help with a false edge before one). Idling a line high protects an input that reacts only to rising edges. Test for it (section 8): switch the outputs in pairs and watch every quiet input, then run random patterns with exact expectations. Join the boards' grounds with a short wire (E1 pins 25 or 26 on both) before any signal; power the boards from the same outlet strip.

3.4 Analog connections

The SMA outputs have a 50 Ω source impedance, the inputs 1 MΩ in parallel with 10 pF. An output's amplitude is calibrated for a 50 Ω load: into a 1 MΩ input it is up to about twice as large at high frequency (2.1× at 10 MHz, measured). Between boards:

- put a 50 Ω terminator at the receiving input (a T with a 50 Ω cap) when the amplitude matters or the cable is long; then the received amplitude is the one set;
- mind the input's range: the LV jumper takes ±1 V, the HV jumper ±20 V (set the board calibration's range to match the jumper);
- one output can feed several inputs through T's, with one termination at the far end.

3.5 States at power-up, loading and `SAFE`

For about two minutes after power-on a board's outputs are undefined; then the service loads the last design and restores its settings. What each design drives right after loading:

- atomic_clock: both 10 MHz sines (uncalibrated, about 0.57 V) and its PPS;
- feedback: its status pins low, its sync low;
- awg: its markers low; the analog and digital outputs off;
- synth, lockin: the outputs off;
- mult_add, analog_echo: an uncalibrated passthrough of the inputs;
- counter, timetagger, ajku: the outputs at 0 V; counter's O1–O4 in high impedance;
- spin: both outputs at 0 V, D1–D8 in high impedance, AO0–AO3 at their idle levels (0 V).
- boxcar: both outputs at 0 V (the test source off), P1–P8 and M1–M4 in high impedance.

`SAFE` returns each design to its documented safe state (see each design's README). Before power-cycling a board that feeds another instrument's input, consider what that input will see.

4 Time and frequency across boards

4.1 Every board has its own clock

Each Red Pitaya runs on its own 125 MHz crystal oscillator; the designs have no input for an external sample clock, so two boards are never phase-locked to each other. Their clocks differ by a few parts per million and drift with temperature: one board here runs 5.66 ppm slow against a GNSS-disciplined 10 MHz (counter's count of the reference on IN2), and a board's crystal moves by about −0.13 ppm per kelvin of the FPGA die's temperature (timetagger's measurement against the same reference); timetagger measured the awg board's crystal 5.02 ppm fast against its own (2026-11-05; 10 ppm in a test of 2026-10-28), and over 2 s awg's edges wandered by 0.46 ns rms around that constant offset: the two crystals' relative phase noise. Two consequences:

- **long intervals disagree**: 1 ppm is 1 µs per second. A rate measured on one board and a duration on another, or two boards' records of the same hour, differ by that much; relate them through a common event (section 5.3) or a common reference (below);
- **short intervals jitter by a clock period**: a signal from one board arrives at a random phase of the other board's clock. A design that acts once per 8 ns clock period then reacts after a delay that varies by 8 ns from event to event (figure 1): an even spread, not a drift. Designs that sample their inputs every nanosecond (counter, the triggers of awg and feedback, timetagger's markers) time the arrival to 1 ns, and timetagger's channels to picoseconds; their *reaction* still waits for their own clock.

Within one shot of an experiment, a hardware trigger from one board makes the other start at a known time to within its time resolution; the crystals' difference only matters over the length of the shot (1 ppm of a millisecond shot is a nanosecond).

4.2 A common reference

Several designs measure or use an external 10 MHz or 1 PPS reference (table 2, section 2). With one reference distributed to all boards (a GNSS-disciplined oscillator, or `atomic_clock`'s outputs), each board's records can be corrected to the reference's time scale.

4.3 Latencies between boards

What matters for feedback between boards is the time from a cause at one board's connector to the effect at another's. Table 3 collects the measured ones; section 2 lists each design's documented values.

Path	Latency
counter: a gate's end → its decision pin	101–109 ns, 105 ns on average (oscilloscope 103.5 ns)
awg: its event input → a digital output aligned with its analog outputs	363 ns
awg: its event input → a digital output at the shortest delay	251 ns
counter's gate end → awg's answer at counter (the preselection loop)	468 ns, or 356 ns
awg: TRIG A → OUT1 (delay 0)	377.9 ns
synth: a step's sample → its marker back at its own trigger input	248 ns
feedback: its sync output → its own TTL input (a jumper)	107.5 ns

Table 3: Measured latencies between connectors (counter and awg, 2026-11-05; awg's trigger 2026-10-24; synth 2026-10-22; feedback 2026-10-26). Not measured: synth's and lockin's trigger to output, awg's event jitter beyond its 8 ns grid, counter's echo and herald latencies.

5 Several boards from one program

5.1 One client per board

Every board runs the same service on TCP port 5025; a program holds one client per board. On a PC with the repository's packages on the path (`common/python` and each design's `control`):

```
from rpfpga.client import Client, connect
Client("192.168.0.104").load("awg")            # make sure each board runs the right design
Client("192.168.0.102").load("counter")
awg = connect("192.168.0.104")                 # a client of the active design's class (AwgClient)
cnt = connect("192.168.0.102")                 # CounterClient
print(awg.idn(), cnt.idn())
```

`connect(host)` returns the client class of whatever design the board runs; a design's own class (`AwgClient`, `CounterClient`, `TimeTaggerClient`, ...) can also be used directly. Clients are thread-safe with one command at a time per connection: two threads, or two processes, can drive two boards in parallel. Each board's settings, presets and calibration are its own.

5.2 Software is slow, wires are fast

A command travels over the network, waits for the board's service, and returns: about a millisecond to tens of milliseconds, with an uneven spread. A loop closed in Python runs at 10–100 Hz at best. So:

- **configure** the boards from the program, before the experiment;
- **start** them with a hardware event: arm every board (a counter run that starts at the next SEQ, a time tagger armed on a channel, an awg sequence that waits for its trigger), then let one board's output start the others;
- **react** through wires (a decision into an event input: section 4.3);
- **read** the results from the program afterwards, or as they stream.

```
cnt.apply({"shots": {"start_at_seq": True}})
cnt.run()                                       # armed: waits for awg's first SEQ
awg.apply({"digital": {"run": "continuous"}})   # awg's pattern starts; its D3 is counter's SEQ
```

5.3 Recording from several boards

Each board records in its own timebase: counter's records count 8 ns words of its crystal, timetagger's tags count its own ticks, a PC's timestamps its own clock. To put records of two boards on one time axis, give both a common event: a marker pulse that one board makes and the other records (a counter SEQ that timetagger also tags, a PPS that both see), and compare the two boards' times of the same pulses; their slope is the boards' relative frequency (parts per million, section 4.1), their offset the start. The record streams of the designs that have them (counter, timetagger) are served by each board's ring pump on port 5026; one program can read several streams in threads.

6 Two boards

Each combination below gives its purpose, the wires (A and B are the two boards; "A pin 4" is E1 pin 4 of board A), the settings that matter, and its test status. Every wire runs from an output to an input; join the grounds (E1 pin 25 or 26) first.

6.1 awg + counter: a sequencer and a photon counter

Tested (2026-11-05, awg 3.0 and counter 1.0, 20 cm jumpers, an oscilloscope)

The pulsed experiments of counter's manual (NV centres, ions, atoms): awg plays the sequence (laser, microwave, TRIG, SEQ) and branches on counter's decision; counter counts, sums per point, and decides. Results: awg's patterns counted exactly (39 million shots in 20 s, every per-point sum

From (output)	To (input)	Signal
A pin 4: awg D0 (DIO0_N)	B pin 3: counter C1 (DIO0_P)	a detector's stand-in (test pulses)
A pin 6: awg D1 (DIO1_N)	B pin 5: counter C2 (DIO1_P)	a second channel; awg's answer
A pin 8: awg D2 (DIO2_N)	B pin 4: counter TRIG (DIO0_N)	shot start
A pin 10: awg D3 (DIO3_N)	B pin 10: counter SEQ (DIO3_N)	sweep start
B pin 12: counter O1 (DIO4_N)	A pin 11: awg EVENT0 (DIO4_P)	the decision
A pin 26	B pin 26	ground

Table 4: awg (A) and counter (B), as tested. In an experiment D0 and D1 drive the laser's AOM and the microwave switch, and the detector goes to C1.

exact, every sweep with a missing TRIG flagged); crosstalk between the jumpers made 2 false TRIGs per million until TRIG's hold-off was set to 450 ns; the decision 101–109 ns after the gate's end; awg's answer through EVENT0 363 ns after the decision (251 ns at its shortest digital delay); the whole loop 468 ns (356 ns) (figure 1).

```
cnt.load_preset("preselection")                     # D1 = G1 >= 2 as a 200 ns pulse on O1
awg.save_segment("z64", [0.0] * 64)
awg.save_pattern("detect", [0x05] * 2 + [0x01] * 12498)   # laser (D0) for 100 us, TRIG (D2) first
awg.save_pattern("go", [0x02] * 25)                       # the experiment's first pulse on D1
awg.save_sequence("presel", [
    {"segment": "z64", "repeat": 200, "digital": "detect", "on_event": "now", "event_target": 1, "
    ↪ next": 0},
    {"segment": "z64", "repeat": 4, "digital": "go", "next": 0}])
prog = lambda p: {"mode": "memory", "pattern": p, "rate": 125e6, "repeats": 1, "end": "idle"}
awg.apply({"arb": {"sequence": "presel", "rate": 125e6, "run": "continuous",
                   "jump": {"source": "event0", "edge": "rising"}},
           "outputs": {"1": {"arb": {"on": True, "amplitude": 0.0}}},
           "digital": {"run": "sequencer", "enable": 0x07, "delay": -112e-9,
```

```
                        "programs": {"detect": prog("detect"), "go": prog("go")}}})
cnt.run()
```

awg's sequence jumps on an edge: use counter's `pulse-true` decision output (a level gives an edge only when the decision changes), and let the waiting step begin before counter's gate ends, since awg ignores an event that arrives before its waiting step's first point.

The same evening, with the same wires (counter's manual, section "The tests on awg's edges"):

- **random patterns with exact expectations**: 896 000 shots of seven random patterns (four gates, two decisions, two coincidence units): five exact in every sum, record and cluster; in two a false TRIG or SEQ edge from crosstalk (section 3.3), and every one of their SHOT records exact at its own TRIG time;
- **repeat until success**: awg attempts every 4.096 µs and waits for EVENT0; counter's test emitter is the photon source, D1 = G1 ≥ 1 on O1 makes awg jump to its "use" step: 2 047 844 attempts, 204 167 successes, $p = 0.0997$ as expected, every success answered once and no failure, the attempts per success geometric (χ^2 71.6 for 78 degrees of freedom);
- **the next day** (two more jumpers: awg's D5 to NEXT, D7 to GATE; awg's OUT1 cabled to IN1): NEXT and GATE exact against their definitions; SEQ sent in the same word as a TRIG lands one slot after it in 1–1.5 % of the sweeps (send it a word earlier); awg's analog pulses as a detector on C9 (a plateau curve, time walk, NIM-like pulses), 33.5 ns after the digital marks awg emits with them on its nominally calibrated board: measure that offset before relying on an analog pulse and its digital mark together;
- **the boards' crystals as a dither**: the two clocks are unlocked (awg 5.02 ppm fast), so awg's edges drift through counter's 1 ns sampling instants and the fraction of shots at a boundary measures an edge's offset to tens of picoseconds: C1's edge 0.52 ns before TRIG + 64 ns, an edge after a short pulse ±95 ps away from an isolated one.

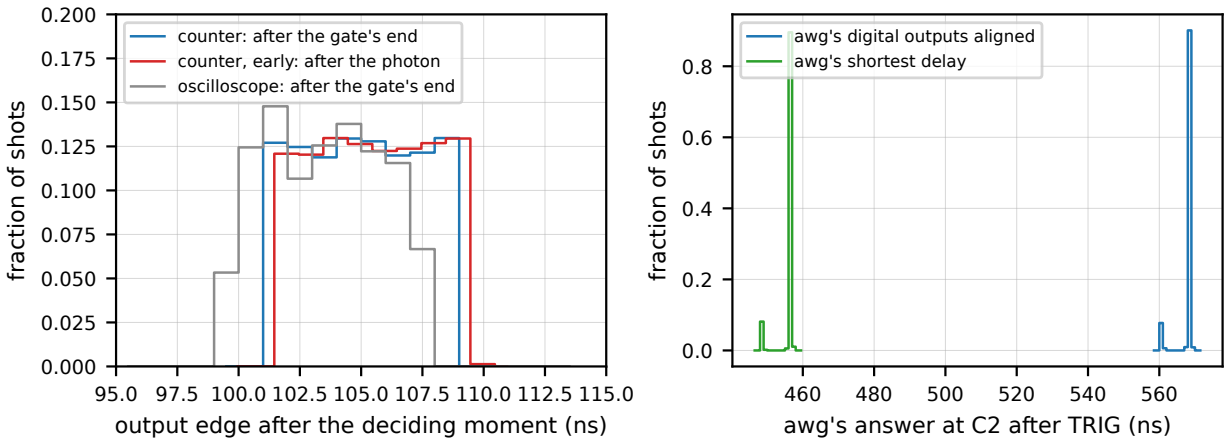


Figure 1: Tested (2026-11-05) Left: counter's decision edge after its deciding moment, measured by counter itself and by an oscilloscope; the 8 ns spread is TRIG's position in counter's clock period. Right: awg's answer at counter's input after TRIG, on awg's own 8 ns grid, with its digital outputs aligned to its analog outputs and at their shortest delay.

6.2 awg + timetagger: sequences with picosecond tags

Tested (2026-11-05, awg 3.0 and timetagger 2.0, 20 cm jumpers)

Time-resolved measurements in a sequence: awg drives a pulsed laser or a modulator and marks the sequence's structure; timetagger tags the photons to picoseconds and the marks to a nanosecond, and the PC sorts the photons by the sequence's step. Results (receipt `guides/combining/data/awg-timetagger-2026-11-05.json`):

From (output)	To (input)	Signal
A pin 4: awg D0 (DIO0_N)	B pin 3: timetagger channel 1 (DIO0_P)	pulses, tagged to ps
A pin 6: awg D1 (DIO1_N)	B pin 5: channel 2 (DIO1_P)	pulses, tagged to ps
A pin 8: awg D2 (DIO2_N)	B pin 4: marker 1 (DIO0_N)	a point's start (1 ns)
A pin 10: awg D3 (DIO3_N)	B pin 10: marker 4 (DIO3_N)	a sweep's start (1 ns)
A pin 26	B pin 26	ground

Table 5: awg (A) and timetagger (B), as tested: the same wires as awg + counter, with timetagger loaded on the counter's board (all its E1 pins are inputs).

- **sorting by markers**: awg's sweeps (SEQ on marker 4, a TRIG per point on marker 1, $k + 1$ pulses on channel 1 at point k) for 5 s: 305 177 sweeps, 2 441 414 points sorted on the PC from the tags, every one with its exact count;
- **awg's outputs at picoseconds**: a 1 µs period 5.02 ppm short in timetagger's time (the two crystals), 30.0 ps rms from period to period (about 21 ps of it timetagger's two tags); D1 20 ps after D0 for edges in the same word (25.8 ps rms, with the two jumpers' lengths); a one-word pulse 8.79 ns wide at the threshold;
- **markers against channels**: awg's D2 16 ns after D0 gave 16.008 ns on average with 290 ps rms, the markers' 1 ns quantization: markers and channels refer to the same connectors;
- **crosstalk**: D0 and D1 rising in the same word fired the markers on the D2 and D3 wires (section 3.3). For marker lines, keep them apart from fast pulse lines, idle them high (the markers count rising edges), or let the marked events never coincide with other outputs' edges;
- **the coincidence engine, its gate and the conditional filter** (evening, timetagger on the same board, no two outputs changing in one word): a pattern of syncs on channel 1, clicks on channel 2 8–120 ns later and a gate on marker 1, 2000–4000 passes per run: the engine's patterns, its four masks and the virtual channel's events in the records exact against `analysis.coincidences` on the pattern, ungated and gated; the filter (60 ns: syncs followed by a click within 8–56 ns kept, 64 ns and later dropped) kept exactly 170 of 705 syncs per pass, and the engine sees the filtered events. Once, one pair of 98 000 was split into two clusters with the filter on; it did not recur in 784 000 more, and no timestamp of 5.7 million deviated from awg's grid by more than 0.6 ns.

timetagger can also start an acquisition armed on a channel: an awg pulse at the sequence's start then defines time zero.

6.3 awg + boxcar: an emulated laser for a pump–probe instrument

Tested (2026-11-09, awg 3.0 and boxcar 1.0, 20 cm jumpers, two coax cables)

Without a laser, awg on a second board plays one: its digital outputs give the laser's sync and a chopper's reference, its analog outputs the detectors' pulses with a known $\Delta T/T$ in its own codes, and boxcar measures them as it would a real experiment, asynchronously (the two crystals differ by a few ppm). Results (boxcar's receipt `board-1.0-awg-session-2026-11-09.json`): every shot taken from 1 kHz to 1.95 MHz (4 million shots), the categories exactly awg's chopper, none invalid; awg's crystal 4.3 ppm fast; $\Delta T/T$ within 4×10^{-4} of awg's exact value for pulses up to 0.68 V at IN1 (the converters' linearity); awg's shared noise of 5.8 % removed by the reference to 0.26 %; a gate holding the whole pulse independent of the laser's phase to ±0.05 % (a 2-sample gate ±35 %); the chopper guard to the clock; the hold released by awg's D2 exactly; with boxcar as the master, awg's pulse 392 ns after P1 and a delay scan with the gates following within ±0.05 %. A chopper edge 60 ns before the gate on one category's shots added 4×10^{-4} to $\Delta T/T$ through the wires: keep the edges half a period from the shots. The code is in boxcar's `RECIPES.md`.

From (output)	To (input)	Signal
A pin 4: awg D0 (DIO0_N)	B pin 3: boxcar TRIG (DIO0_P)	the laser's sync
A pin 6: awg D1 (DIO1_N)	B pin 5: boxcar CHOP (DIO1_P)	the chopper's reference
A pin 8: awg D2 (DIO2_N)	B pin 7: boxcar ARM (DIO2_P)	NEXT (a stage's "in position")
B pin 4: boxcar P1 (DIO0_N)	A pin 3: awg TRIG A (DIO0_P)	boxcar as the master
A pin 26	B pin 26	ground
A OUT1, OUT2 (coax)	B IN1, IN2	the signal and the reference detector

Table 6: awg (A) and boxcar (B), as tested. awg loops two shots (pump on, pump off) of its sequencer, a detector-like pulse on each output, with a digital pattern per shot: the sync on D0, the chopper on D1 with its edges half a period from the shots.

6.4 counter + timetagger: counting and tagging the same photons

Not tested

counter decides and sums on the board; timetagger keeps every photon's time. Both read the same detectors:

- **split each detector's signal**: a T to both boards' inputs (both high impedance), with the 50 Ω termination at the far end only; channels 1–8 have the same pins on both designs;
- **counter's outputs as timetagger's markers**: counter's O1–O4 (DIO4_N–DIO7_N) into timetagger's markers 5–8 on the *same-named* pins of the other board (output to input): a decision, a gate's window or a herald then marks the time tags, so that the PC selects the photons of heralded or of "bright" shots;
- **NIM detectors**: counter's IN1, IN2 discriminate NIM pulses and its echo outputs give one 4 ns logic pulse per counted pulse; into timetagger they are timed to counter's 2 ns output grid, about 0.1 µs late: enough for counting and coincidences, not for picosecond timing (use a NIM-to-TTL converter for that);
- both use the same definition of a coincidence: their coincidence counts of the same events agree to counter's 1 ns resolution.

6.5 counter + feedback: holding an experiment at its maximum count rate

Not tested

A fibre coupler, a laser's frequency on a fluorescence line, or a sample's position drifts; the count rate tells how far. counter's rate monitor turns the rate into a voltage on its OUT1; feedback dithers a piezo (or the laser's current) with one of its oscillators, demodulates the rate monitor at the dither frequency (the slope of the rate), and its PID holds the slope at zero: the peak. The monitor's

From	To	Signal
A OUT1 (counter's rate monitor)	B IN1 (feedback, 50 Ω at the input)	the rate as a voltage
B OUT1 (feedback: dither + PID)	a piezo driver's input	the actuator
B pin 4: feedback status (DIO0_N, "locked")	A pin 6: counter GATE (DIO1_N)	count only while locked

window must be shorter than the dither period (it averages 2^k words, 128 ns to 134 ms); the lock's noise is the shot noise of the counts in each window. counter's gates in `gate` mode, with GATE from feedback's "locked" status pin, count only while the lock holds.

6.6 synth + counter: frequency scans counted per point

Not tested

A radio-frequency scan whose effect is counted: the secular-frequency "tickle" of a trapped ion (a weak drive on an electrode heats the ion at its motional frequency and its fluorescence drops), an

RF transition in a vapour cell. synth's sequence steps the frequency and marks each step on DIO1_P; counter counts per step:

- A pin 5 (synth's marker, DIO1_P) to B pin 4 (counter TRIG, DIO0_N): one shot per step, G1 as long as the step's dwell, per-shot points (a SEQ from the sequence's start, or `timer_seq` off and the points cycling);
- or to B pin 8 (NEXT): several shots per step from counter's own timer, the points following synth's steps.

The marker comes on the first sample of each step (248 ns after the step in synth's own loop test).

6.7 A reference and a timekeeper: atomic_clock or a GNSS oscillator

Not tested (each design's reference input was tested with a GNSS-disciplined oscillator)

- **comparing two boards' crystals**: atomic_clock's OUT2 is exactly 2/25 of its board's clock; into counter's IN2 (C10, threshold 0 V) it is counted against counter's clock: the two boards' relative frequency to a fraction of a ppm in seconds;
- **a common reference**: a GNSS-disciplined 10 MHz split to timetagger's IN2, counter's IN2, lockin's or atomic_clock's inputs, and its PPS to timetagger's channels and ajku's PPS input (DIO0_P): every board's records can then be put on the reference's time scale (section 4.2);
- **a clock under test**: atomic_clock's OUT1 with programmed phase noise and drift, tagged by timetagger against a reference: the clock comparisons of timetagger's manual, with a known answer.

6.8 awg + feedback: a sequence and a lock

Tested (2026-11-05 (the TTL reference only), awg 3.0 and feedback 2.0, the awg + counter wires)

feedback locks a laser (to a cavity, a line, a reference); awg runs the experiment's sequence. The lock's state belongs in the sequence: feedback's status pin "locked" (DIO0_N–DIO3_N, driven) into awg's TRIG B (DIO3_P) or an EVENT input (DIO4_P–DIO7_P) lets the sequence wait or branch while the lock is lost; awg's marker into feedback's capture trigger (DIO3_P) records the lock's error signal at a chosen step of the sequence. Never connect awg's D0–D3 to feedback's status pins: both are outputs on the same-named pins.

Tested: awg's D0 (DIO0_N) as a 5 MHz TTL clock into feedback's TTL reference (DIO0_P), with feedback loaded on the board that had run counter. The bench's other wires from awg's D1–D3 led to feedback's sync and status pins: awg left them in high impedance (`enable` = D0 only), which is how one wiring can serve several designs safely.

- feedback's counter read the clock at 5 000 025.24 Hz (1 s gate): the crystals' difference, +5.05 ppm, as counter and timetagger measured it;
- `lock_ttl` (1 kHz) locked oscillator 2 at 1, 2 and 3/2 times the clock: 200 of 200 readings locked, the mean frequency within its error of the counter's; the phase readings 0.5° rms (290 ps): the 1 ns timestamps' quantization, a slow sawtooth because the 5 ppm drift moves each edge by only 1 ps, so that averaging 8 or 64 edges per reading did not reduce it and 512 halved it;
- **holdover**: with the clock switched off the PID held, but at 250–570 Hz from the locked frequency with `lock_ttl`'s loss time of 20/bandwidth (20 ms): until the loss is declared, frames without an edge repeat the last reading and the integrator keeps integrating it. A loss time of 1 ms gave 6–15 Hz, 4 µs 0.4–2.7 Hz. For a fast clock set `ttl.loss_time` to a few of its periods.

6.9 awg or synth + lockin: excitation and detection

Not tested

lockin has its own oscillators, so it needs another board only for an excitation it cannot make: an arbitrary waveform, a pulse train, a timed sequence of tones. Feed the excitation's signal (or a copy) into one of lockin's inputs and lock a PLL to it: lockin then demodulates at the excitation's frequency as the other board's crystal sees it, with no clock shared. awg's marker into lockin's capture trigger (DIO3_P) captures the response at the excitation's start.

6.10 Analog chains: mult_add and analog_echo

Not tested

mult_add and analog_echo process analog signals between other instruments: mult_add forms, for example, a normalized difference $(X - Y)/(X + Y)$ of two photodiodes as an error signal for feedback, or a product for a simple demodulation; analog_echo scales and offsets a signal, for example to bring awg's output into a sensitive input's range. Each adds its processing delay (mult_add 456 or 608 ns, analog_echo 32 ns inside the FPGA) and the converters' delays and bandwidth (about 60 MHz) to the chain.

7 Three boards

Three boards combine the pairs of section 6; the rules do not change. Draw the wiring as a table of outputs and inputs first, and check against the pin map that no pin is an output on two boards.

7.1 awg + counter + timetagger: an NV experiment with every photon kept

Not tested

awg sequences, counter counts, sums and preselects, timetagger keeps every photon's time for the fluorescence's time-resolved shape and for correlations.

From (output)	To (inputs)	Signal
awg D0, D1	the laser's AOM, the microwave switch	the sequence
awg D2, D3	counter TRIG, SEQ	shots and sweeps
awg marker 1 (DIO1_P)	timetagger marker 1 (DIO0_N)	the sweep's start, for sorting the tags
the SPAD (through a T)	counter C1 and timetagger channel 1	the photons (50 Ω at the far end)
counter O1	awg EVENT0	preselection
counter O2 (a gate's window)	timetagger marker 6 (DIO5_N)	which photons fell in the readout window

7.2 awg + counter + feedback: an ion or atom experiment with its laser lock

Not tested

feedback holds the cooling or detection laser locked; awg runs the sequence; counter detects the state and watches for the ion's loss.

- feedback's status "locked" into awg's TRIG B: the sequence waits while the lock is lost;
- the same status into counter's GATE: no counts are taken while the laser is unlocked;
- counter's rate alarm on an output into an awg event input: the sequence branches to reloading when the ion is lost;
- awg's D-outputs into counter's TRIG and SEQ, and counter's decision into another awg event input, as in section 6.1.

Count the outputs and inputs: awg's four event inputs (DIO4_P–DIO7_P) and two triggers serve three sources here; counter's four outputs serve a decision and the alarm; feedback's four status pins its lock states.

7.3 A common reference for all boards

One reference, distributed: a GNSS-disciplined (or atomic_clock's) 10 MHz through a splitter to each board that measures it (timetagger IN2, counter IN2, atomic_clock's or lockin's inputs), its PPS through a buffer to timetagger's channels and ajku's PPS input. Each board still runs on its own crystal; the reference lets each one's records be corrected to one time scale afterwards, and shows each crystal's drift as it happens.

8 Testing a combination

A combination is two instruments and the wires between them; each instrument has been tested on its own, the wires and the assumptions about timing have not. This section is the procedure used for awg and counter on 2026-11-05 (section 6.1), with what it found at each step.

8.1 1. Static levels

With both designs loaded and all outputs in high impedance, wire, then drive each output high and low from the program and read the input at the other end:

```
for word in (0x00, 0x01, 0x02, 0x04, 0x08):          # awg's D0..D3 one at a time
    awg.set_digital(run="off", idle=word, enable=0x0F)
    print(hex(word), sorted(k for k, v in cnt.pins().items() if v["level"]))
awg.set_digital(run="off", idle=0, enable=0)          # back to high impedance
for mode in ("low", "high", "low"):                   # counter's O1, O4 the other way
    cnt.apply({"outputs": {"1": {"mode": mode}, "4": {"mode": mode}}})
    print(mode, awg.hardware()["levels"]["event0"], cnt.pins()["C8"]["level"])
```

Every output should change exactly its intended input and nothing else. In the first try two of seven jumpers had one end on the wrong board: driving the *free* outputs of the other board, and reading every input of both, showed where each had gone. A wire that connects nothing is harmless; one between two outputs is not, which is why the boards' outputs stay in high impedance until this check is done.

8.2 2. Exact counts

Next, a deterministic test: a pattern whose every pulse is known, counted by the receiving instrument. awg played sweeps of 8 points with $k + 1$ pulses on one channel and $8 - k$ on another at point k; counter's per-point sums had to be exact multiples of the shots. The first run was not: 2 TRIG edges per million were spurious. Varying which wires carried pulses located them: 8 ns pulses on the jumper beside the TRIG wire coupled into it. A hold-off of 450 ns on TRIG removed them, and 39 million shots were then counted exactly. Without the exact test this crosstalk would have shown only as a few ppm of misassigned points.

8.3 2a. Crosstalk

Switch the sender's outputs in pairs (both in the same clock period, then a period apart) and record every input of the receiver, also the ones that should stay quiet. In the awg + timetagger test (section 6.2) two outputs rising together fired two quiet marker lines; neither alone did (section 3.3).

8.4 2b. Random patterns with exact expectations

A fixed pattern tests a few cases many times; random patterns test many cases. Generate the sender's pattern at random (pulses, gaps, coincidences of every kind), compute on the PC what the receiver must count from the receiver's own definitions (gates, dead times, clusters), and compare every number and record. When a difference appears, the receiver's records (times, points) usually show whether its input or its logic is at fault: in the awg + counter test every record of two failing

patterns was exact at its own TRIG time, which put the fault on the wires. Keep the boundaries (gate edges, windows, dead times) a few nanoseconds away from the pattern's grid, or predict both outcomes there: the sub-nanosecond offsets of real edges decide them.

8.5 3. Timing from the instruments themselves

The receiving instrument can often time the sender's signals: counter's gate-delay scan measured, with 1 ns windows, when its own decision arrived back at an input and when awg's answer arrived, both against the shot's TRIG, at its connectors. That gave the decision latency and the loop time without any other instrument, over millions of shots, with their spreads (section 4.3).

8.6 4. An oscilloscope

An oscilloscope checks the instruments' own timing independently. Three practical points:

- **a pin with a jumper takes no probe**: let the sender copy each wired signal onto a free pin (awg's pattern on two digital outputs at once; counter's decision on two outputs) and probe the copies; the copies' own skew is small, and can be measured on free pins before wiring;
- **probes and coax inputs have different delays**: a 10× probe's cable added 7.04 ns against a direct coax input, measured by putting both on the same pulse in a BNC T; without that correction an analog-versus-digital comparison would have been wrong by 7 ns;
- **slow probes** shift an edge's crossing: calibrate each against a fast one on the same signal (+1.09 ns for the slow probe used here).

The scope can be read over the network (SCPI) and the edges' crossings computed from the waveforms, hundreds of acquisitions in minutes.

8.7 5. Receipts

Keep the numbers: each design's `docs/data/` holds the JSON receipts of its board tests, and the awg and counter session's is `counter/docs/data/board-1.0-awg-scope-session.json`. A combination's receipt records the wiring, the boards' builds and calibrations, the scope and probes, and every measured number, so that the next test can be compared with it.

A The E1 pins of every design

Plain: an input (with a pull-down where documented). **Bold**: an output driven from the moment the design loads. Dagger†: an output in high impedance until enabled (and after `SAFE`). –: not used. E1 pins 1 (+3.3 V) and 2 (−3.4 V) supply power; 25 and 26 are ground; 19–24 are used by no design. analog_echo and mult_add use no E1 pin.

Other connectors: feedback uses E2 (AO0–AO3 outputs, AI0–AI3 inputs), spin drives AO0–AO3 and ajku reads AI0–AI3; timetagger reads SATA S2 and drives SATA S1 only when its test output is enabled. The state of a pin that a design does not use is not documented: treat it as unknown and leave it unconnected.

Pin	E1	awg	counter	timetagger	feedback	synth	lockin	ajku	atomic_clock	spin	boxcar
DIO0_P	3	TRIG A	C1	ch 1	TTL ref.	trigger	–	PPS	–	TRIG	TRIG
DIO1_P	5	**marker 1**	C2	ch 2	**sync**	**marker**	–	–	**PPS**	COND A	CHOP
DIO2_P	7	**marker 2**	C3	ch 3	–	–	–	–	**ref. PPS**	COND B	ARM
DIO3_P	9	TRIG B	C4	ch 4	capture	–	capture	–	–	–	AUX
DIO4_P	11	EVENT 0	C5	ch 5	–	–	–	–	–	–	M1†
DIO5_P	13	EVENT 1	C6	ch 6	–	–	–	–	–	–	M2†
DIO6_P	15	EVENT 2	C7	ch 7	–	–	–	–	–	–	M3†
DIO7_P	17	EVENT 3	C8	ch 8	–	–	–	–	–	–	M4†
DIO0_N	4	D0†	TRIG	marker 1	**status 0**	–	–	–	–	D1†	P1†
DIO1_N	6	D1†	GATE	marker 2	**status 1**	–	–	–	–	D2†	P2†
DIO2_N	8	D2†	NEXT	marker 3	**status 2**	–	–	–	–	D3†	P3†
DIO3_N	10	D3†	SEQ	marker 4	**status 3**	–	–	–	–	D4†	P4†
DIO4_N	12	D4†	O1†	marker 5	–	–	–	–	–	D5†	P5†
DIO5_N	14	D5†	O2†	marker 6	–	–	–	–	–	D6†	P6†
DIO6_N	16	D6†	O3†	marker 7	–	–	–	–	–	D7†	P7†
DIO7_N	18	D7†	O4†	marker 8	–	–	–	–	–	D8†	P8†

Table 7: The E1 pins of every design. Every pair of boards is checked against this table before wiring: an output (bold or dagger) may be wired only to inputs (plain).