18 — Multi-Probe Synchronization

> **TraxProbe & Traxcope User Manual** · Version 1.0.0 · Updated 2026-07-28
 · https://embedya.com

When several boards are traced at once, each runs on its own clock — timelines drift apart within seconds. Multi-probe sync aligns them with a **single shared GPIO wire**: one board (the master) periodically pulls the line low, every board timestamps the same falling edge with its own local clock, and Traxcope computes the mapping between clocks from those simultaneous observations. The result is one true timeline across all probes — a context switch on board A and the ISR it triggers on board B line up exactly in the Trace View.

The host is a passive observer: all sync signaling is firmware-driven.

Hardware setup

```
    Board A (MASTER)        Board B (SLAVE)        Board C (SLAVE)
    GPIO open-drain   ──────────┬──────────────────────┬──────────
         + EXTI                 │                      │
                                │   shared sync wire   │
                          ──────┴──────────────────────┴──────  + pull-up (e.g.
10 kΩ) to Vcc
```

“images/sync_wiring_diagram.png” could not be found.

Image: wiring diagram — open-drain GPIOs on one wire with external pull-up

- **All probes**: the pin is an open-drain output (released HIGH) **and** has a falling-edge EXTI interrupt enabled.
- **Exactly one master** per wire pulls the line LOW to generate sync pulses. Everyone — including the master itself — timestamps the falling edge.
- Ground must be common between the boards.

On the device

Configuration (`trax_config.h`)

Master:

```
#define TRAX_CFG_SYNC_ROLE              TRAX_SYNC_ROLE_MASTER
#define TRAX_CFG_SYNC_GPIO_SET_LOW()    my_sync_gpio_low()    /* must be ISR-
safe */
#define TRAX_CFG_SYNC_GPIO_SET_HIGH()   my_sync_gpio_high()
```

Slaves:

```
#define TRAX_CFG_SYNC_ROLE  TRAX_SYNC_ROLE_SLAVE
/* no GPIO macros needed — slaves only listen */
```

If your sync input passes through a hardware glitch filter with a known delay, declare it as a rational number so the host can compensate:

```
#define TRAX_CFG_SYNC_FILTER_DELAY_NUM  BSP_SYNC_FILTER_DELAY_NUM
#define TRAX_CFG_SYNC_FILTER_DELAY_DEN  BSP_SYNC_FILTER_DELAY_DEN
```

The role is embedded in the session-start header, so **Traxcope auto-detects who the master is — no UI assignment step**.

Code (both roles)

In the EXTI handler for the sync pin, as early as possible:

```
void EXTI4_15_IRQHandler(void)
{
    if (exti_pending(SYNC_PIN)) {
        trax_sync_triggered();      /* timestamps the edge, emits a sync frame
*/
        exti_clear(SYNC_PIN);
    }
    /* ... */
}
```

Code (master only) — generating pulses

```
trax_sync_trigger_start();          /* pull LOW + capture own timestamp,
atomically */
vTaskDelay(pdMS_TO_TICKS(1));       /* hold LOW long enough for every slave to
latch  */
trax_sync_trigger_end();            /* release HIGH */
```

Call sites that work well:

- once shortly **after stream start** → the "first" anchor;
- **periodically** (e.g. 1 Hz) or at least once **before stream stop** → the "last" anchor.

Two anchors spaced far apart let the host estimate not just the clock **offset** but also the **rate difference (scale)** between boards, so alignment stays exact across long sessions. Both functions are safe no-ops on slave/none builds.

In Traxcope

Enabling and monitoring

Multi-probe sync is enabled workspace-wide from the probe configuration (sync section). Once enabled:

1. Each probe's sync role (from firmware) is read at session start; the master becomes the **reference probe** automatically.
2. Every probe's `SYNC_TIMESTAMP` frames are collected; for each non-reference probe the host fits the linear mapping `t_ref = offset + scale × t_local`.
3. All views project every probe's data through its mapping — lanes, plots, logs and events from different boards land on one shared time axis.

The **sync status panel** in probe configuration shows: role, mapping state, computed offset and scale, and the pulse count.

Also available there:

- **Sync filter delay override (µs)** — host-side correction if the firmware didn't declare its input-filter delay;
- **Manual time offset/scale** per probe — fallback alignment when no sync hardware is available.

Traxcope warns about misconfiguration: **two masters** on the wire, or sync enabled with **no master** present.

What you get in the views

- **Trace View** — task/ISR lanes from multiple boards interleaved on one timeline; measure cross-board latencies (e.g. board A sends a message → board B's handler runs) with cursors.
- **Log / Event views** — multi-probe filters with rows in true global order.
- **Scope** — plot variables from different boards against the same time axis.
- **Recordings** — sync anchors are persisted in the recording manifest, so synchronized review works offline exactly like live.

Without hardware sync

No free wire between the boards? You can still co-display multiple probes; expect drift. Use the manual **offset/scale** correction (per probe and per recording) to align around a known common event. For real measurements across boards, the hardware wire is strongly recommended — it is one GPIO and ~20 lines of code.

Troubleshooting

Symptom	Likely cause
"No master" warning	No connected probe was built with `TRAX_SYNC_ROLE_MASTER`
"Duplicate master" warning	Two firmwares claim master on the same workspace — rebuild one as slave
Offset computed but timelines still drift	Only one pulse received → no scale estimate; pulse periodically or at least at start *and* end
Constant small skew between boards	Input-filter delay not declared — set `TRAX_CFG_SYNC_FILTER_DELAY_*` or the UI override
No sync frames from one board	EXTI not firing (pin config, missing pull-up) or `trax_sync_triggered()` not called in its ISR