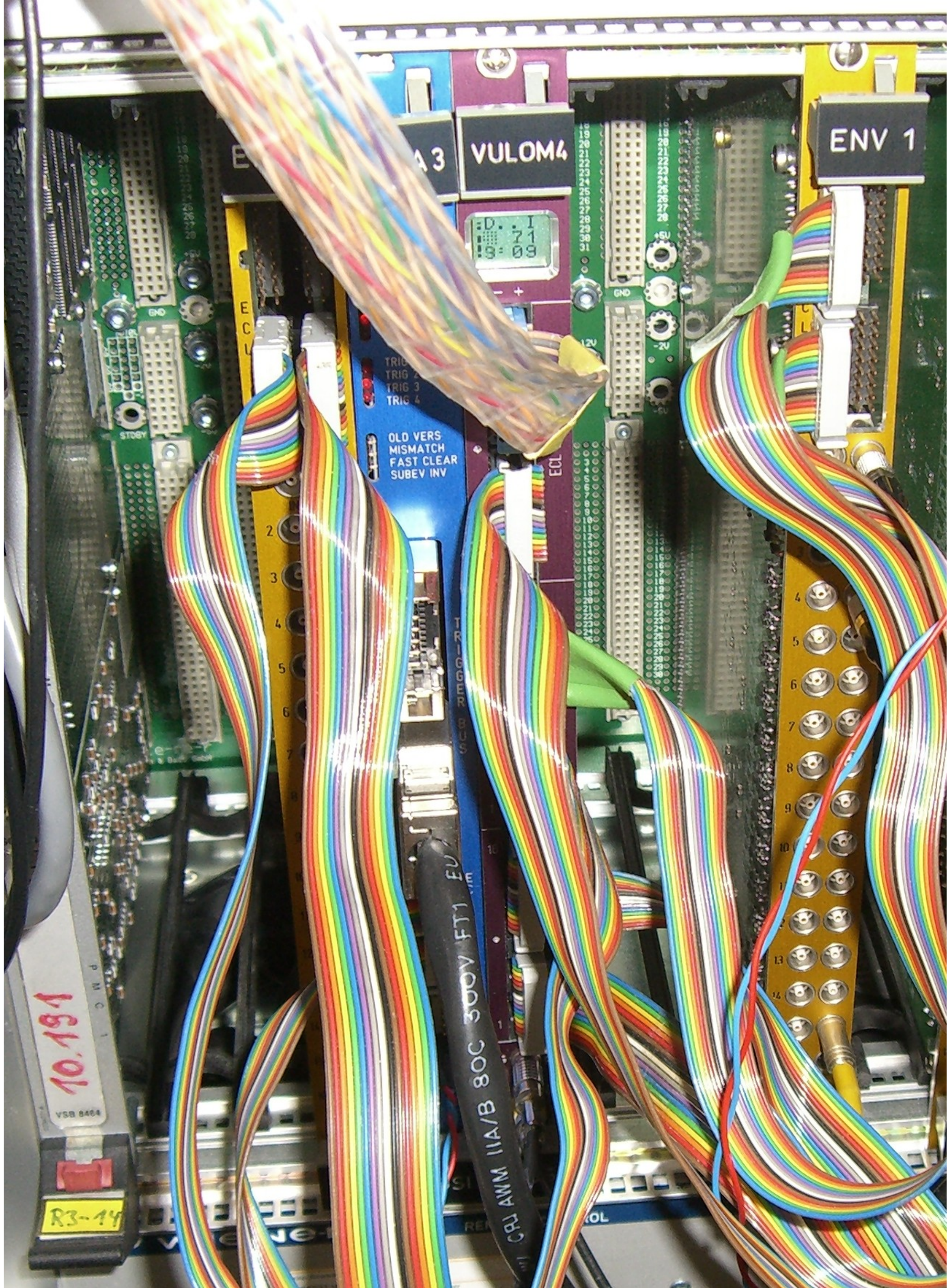
TRLO II with TRIMI – friendly FPGA trigger control

```
COMMAND(wiring)
{
  DEADTIME_IN(1) = ECL_IO_IN(4);

  ECL_IO_OUT(1) = ENCODED_TRIG(1);
  ECL_IO_OUT(2) = ENCODED_TRIG(2);
  ECL_IO_OUT(3) = ENCODED_TRIG(3);
  ECL_IO_OUT(4) = ENCODED_TRIG(4);
```

Håkan T. Johansson,
Chalmers, Göteborg

IT/EE-Palaver, GSI, Darmstadt, Jan 2014



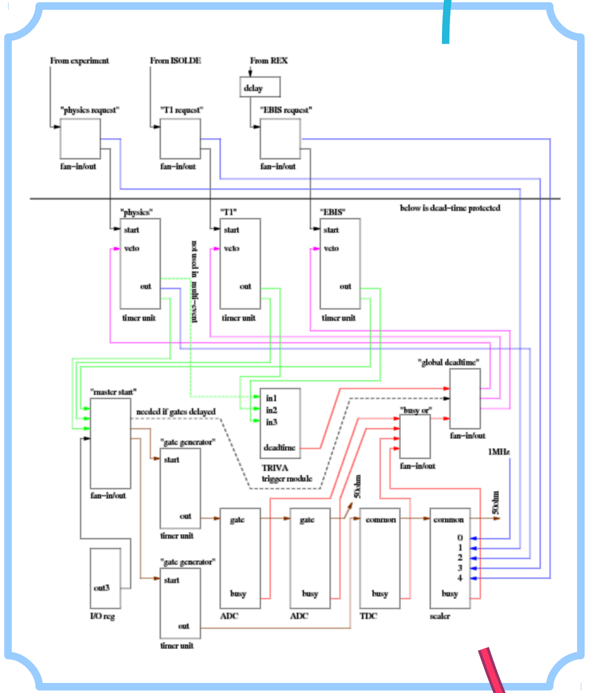
Outline:

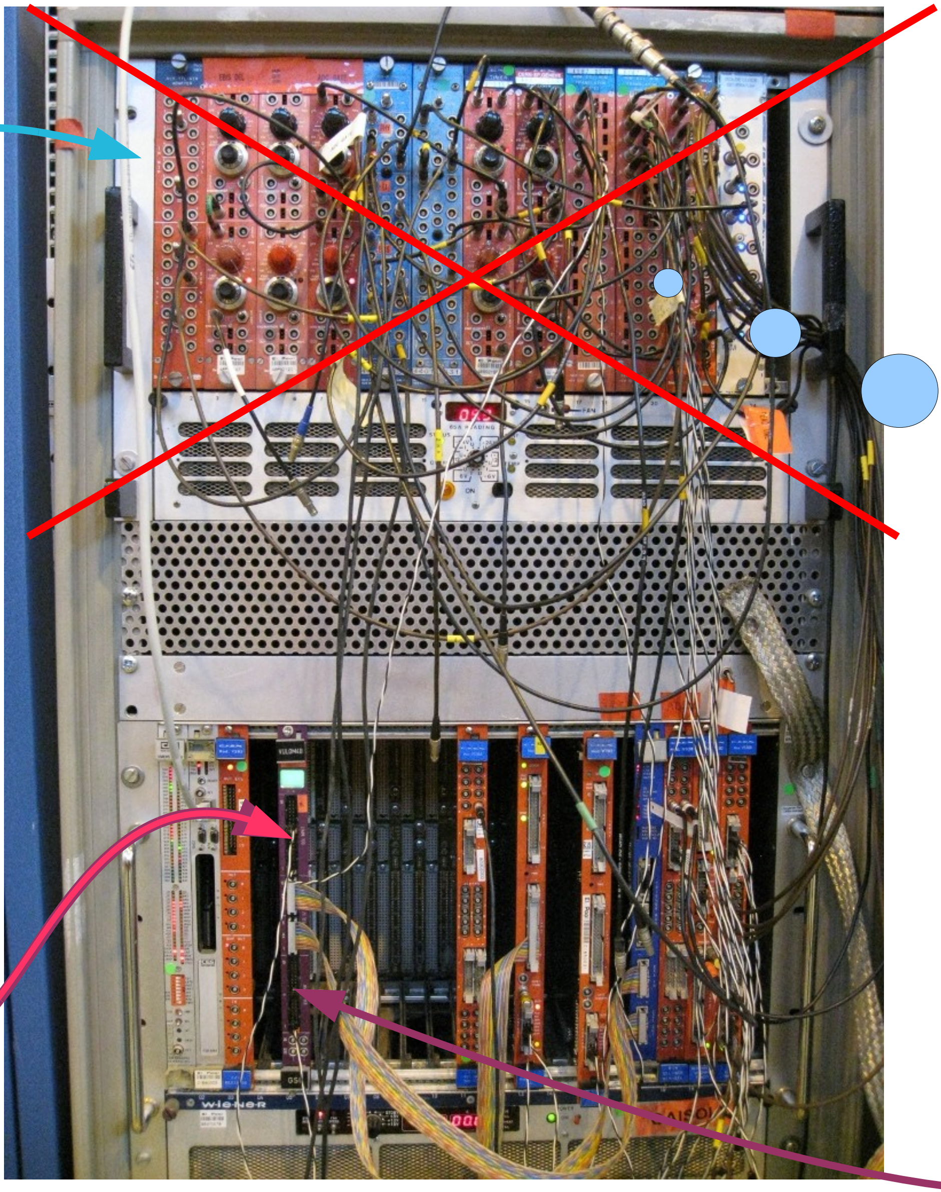
- "In the previous episode..." / Commercials
- `trloctrl` - with parsed setup files
- Serial protocol, v2
- TRIMI — TRIVA mimic
- Serial timestamps

Example: TRLO II for ISOLDE

ISOLDE “std” VME DAQ gets TRLO II

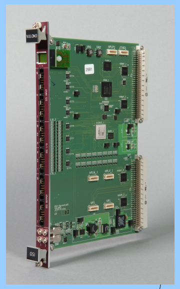
Trigger logic filled
1 NIM crate





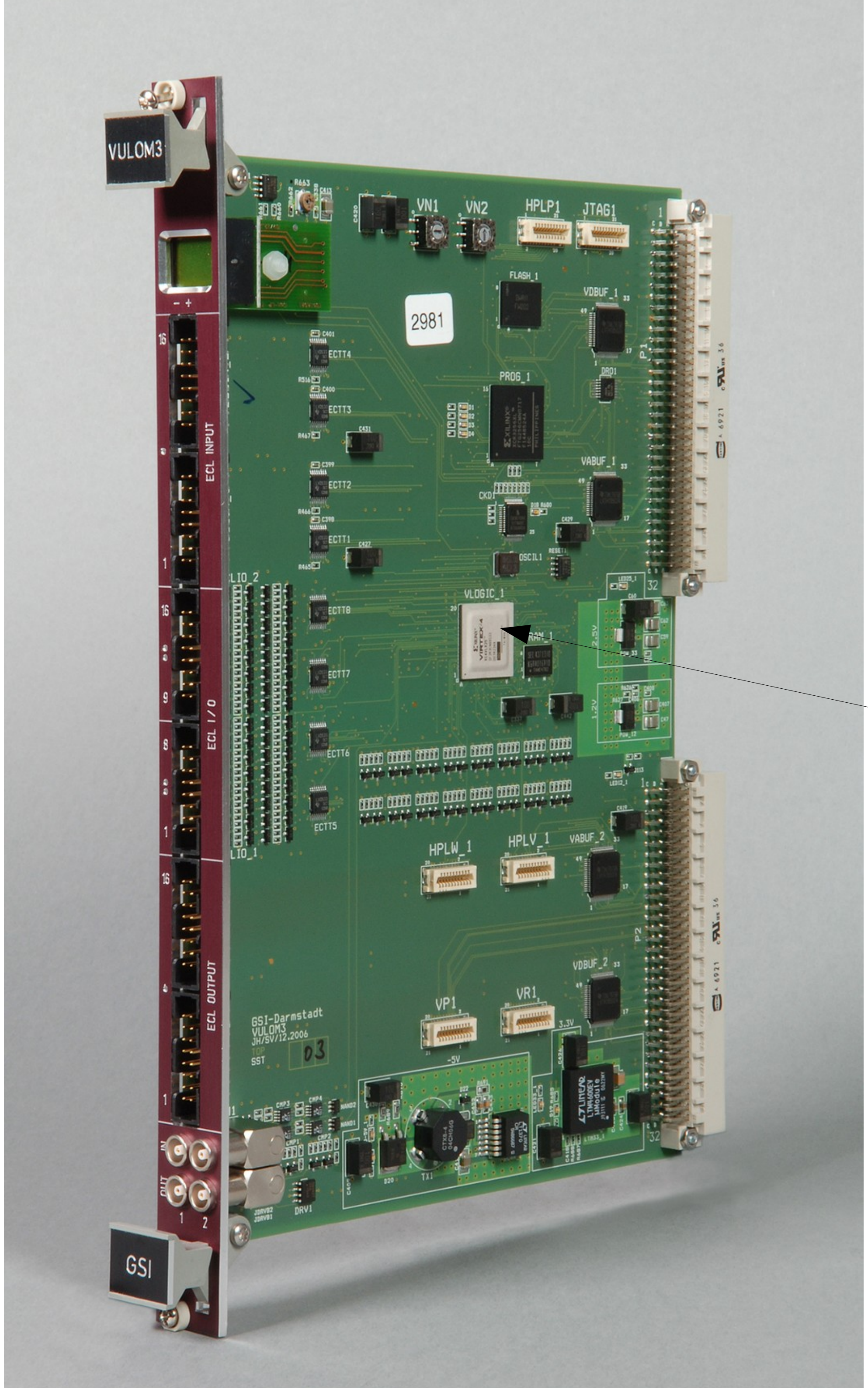
Now by TRLO II
in VULOM4B

TRLO II is a
firmware for
VULOM and TRIDI
modules



VULOM
(VME universal logic
module)

by J. Hoffmann, GSI



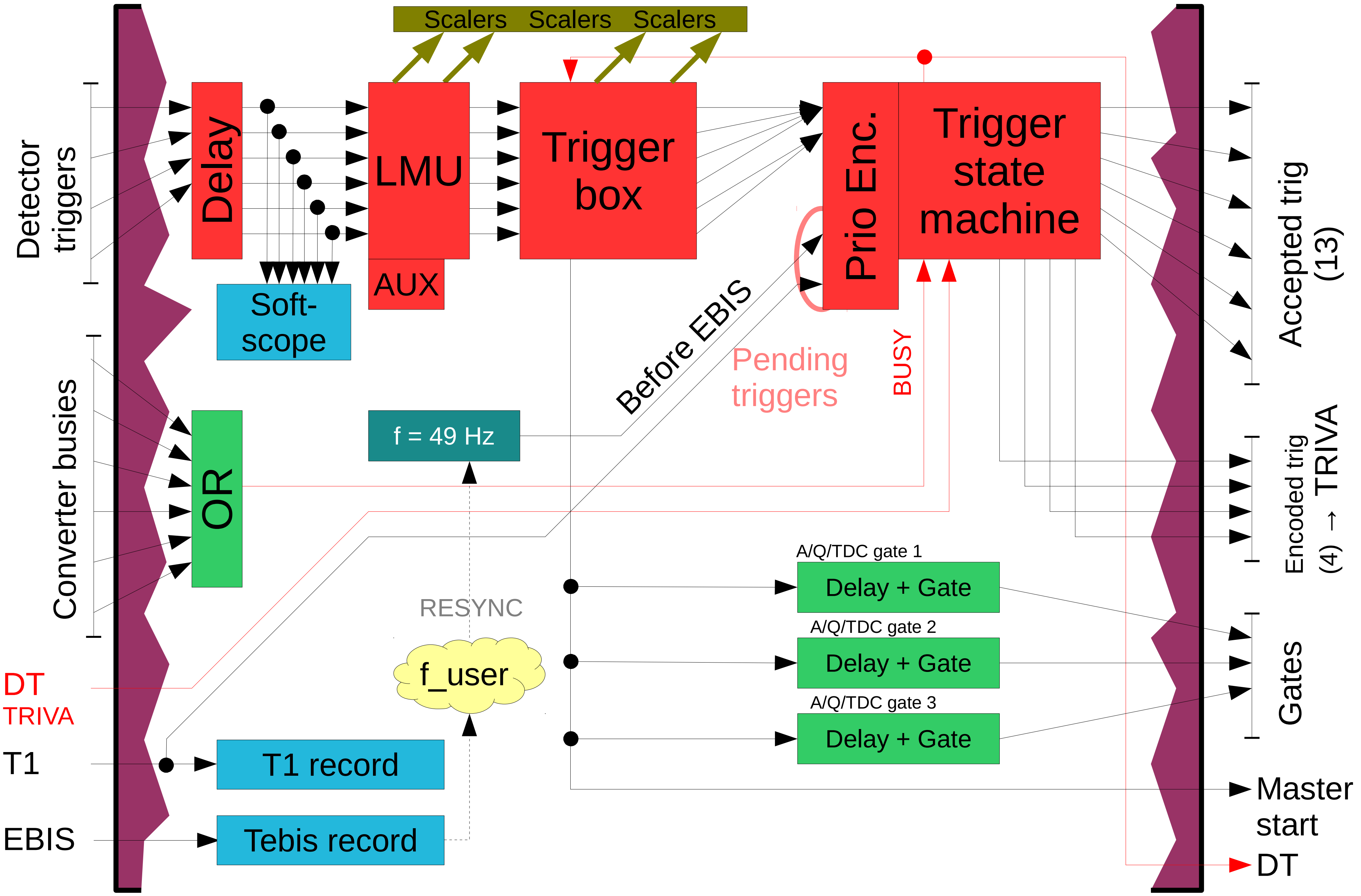
Inputs

Outputs

VME

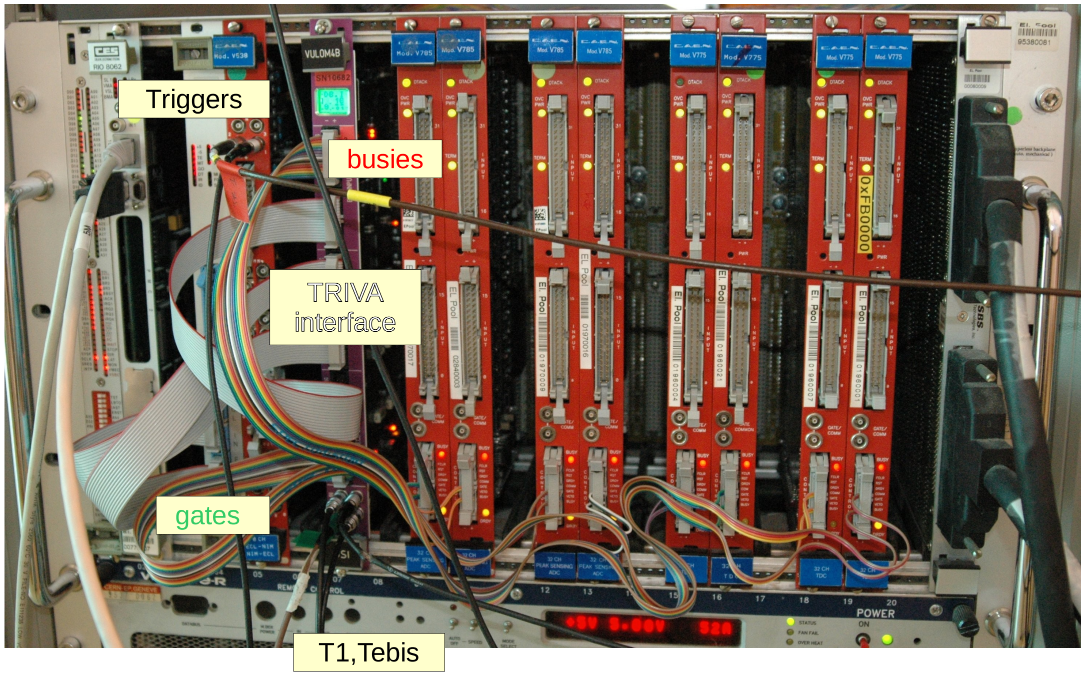
FPGA

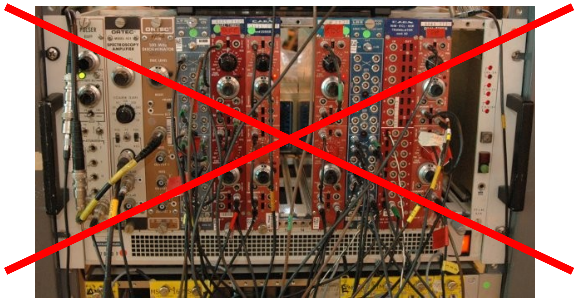
TRLO II @ ISOLDE "std" VME



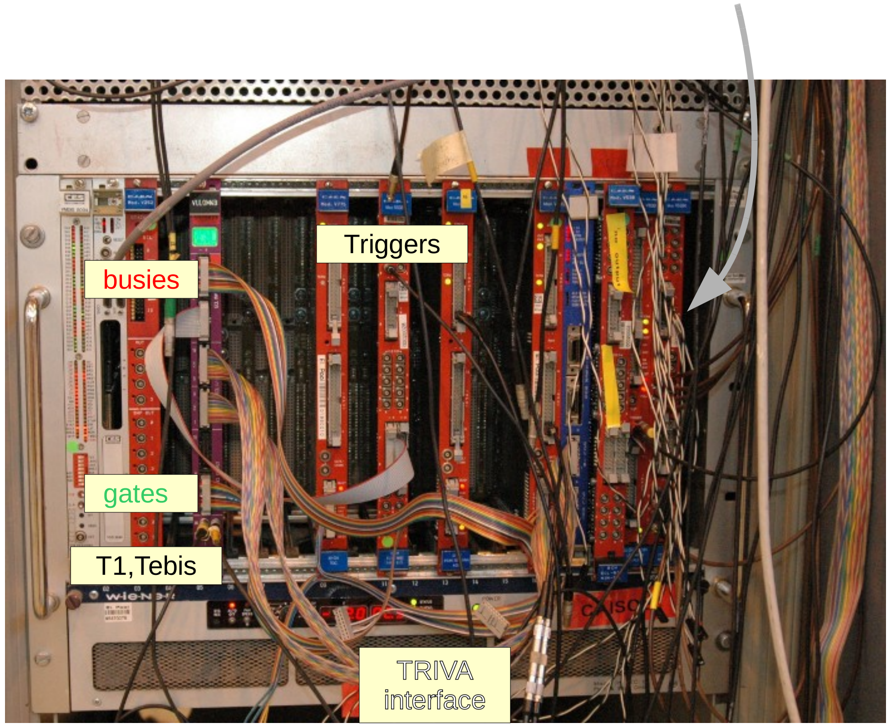
Only a few cables left (practice)

Triggers, accelerator markers,
TRIVA interface, gates & busies





No more leaking deadtime!



Some signals too?

TRLO II in operation:

S393 S306b S389 (2010)

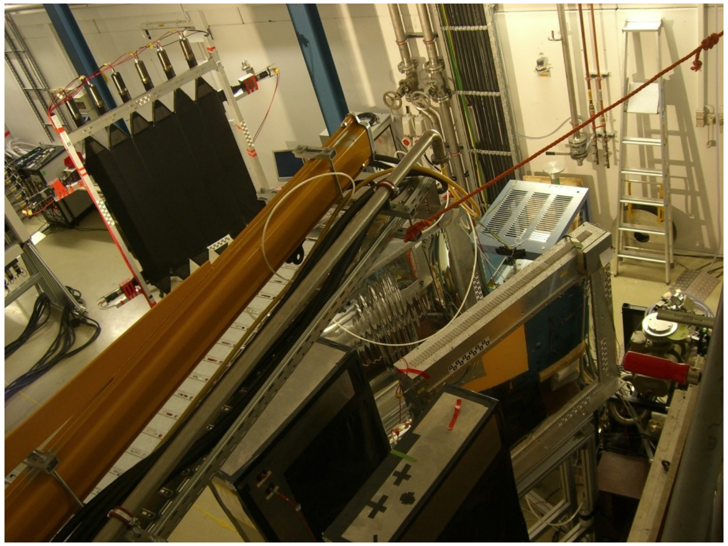
S408, S405 (Oct 2011)

S412, S406 (Apr – Jun, Nov 2012)

(LAND/R^{3}B)

S371 (FIRST)
(V. Monaco, R. Sacchi) (Jul – Aug 2011)

S424 (AGATA-PRESPEC)
(D. Ralet, S. Pietri) (Apr – Jun 2012)



And more...

I83
IS512
…
TU-Da
Duke

Preceded by intense *code* inspection (~300 kB) →

0 critical bugs found

4 minor bugs found in the wild
(one by users :-))

Next: `trloctrl`

Stable VME interface definition

VME registers as C structure,
with named entries,
generated by compilation

Version number is MD5
of full VHDL code

Named constants

Setup registers in block
RAM for readback.

Aggressive checksumming
of output data.

```
#define TRLO_MD5SUM_STAMP           0xccb60dee

// Constants for 'direct_mode':

#define TRLO_DIRECT_MODE_LOGIC             0x0
#define TRLO_DIRECT_MODE_DIRECT            0x1
#define TRLO_DIRECT_MODE_LOGIC_OR_DIRECT   0x2

typedef struct trlo_output_map_t
{
  /*    0 0x0000 */ uint32_t version_md5sum;
  /*    1 0x0004 */ uint32_t compile_time;
  /*    2 0x0008 */ uint32_t timing_tick;
  /*    3 0x000c */ uint32_t deadtime_tick;

typedef struct trlo_setup_map_t
{
  /*    0 0x2000 */ uint32_t mux[122];
  /*  122 0x21e8 */ uint32_t direct_mux[26];
  /*  148 0x2250 */ uint32_t direct_mode[26];
  /*  174 0x22b8 */ uint32_t direct_or[3];
  /*  177 0x22c4 */ uint32_t scaler_mode[8];

// MUX src indices:

#define TRLO_MUX_SRC_ECL_IN(i)        ( 0+(i))
#define TRLO_MUX_SRC_ECL_IO_IN(i)     (16+(i))
#define TRLO_MUX_SRC_LEMO_IN(i)       (24+(i))
#define TRLO_MUX_SRC_WIRED_ZERO       (32)
```

trloctrl – TRLO II command-line control

VME interface by named C structure elements is nice, but still cumbersome...

`trloctrl` already for LAND/R^{3}B setup, but setup specific

New tool with easy generic command line access:

```
trlo_ctrl “LEMO_OUT(2)=GATE_DELAY(1)”
```

Recurring operations (e.g. setup) in parsed configuration file:

```
COMMAND(wiring)
{
  DEADTIME_IN(1) = ECL_IO_IN(4);

  ECL_IO_OUT(1) = ENCODED_TRIG(1);
  ECL_IO_OUT(2) = ENCODED_TRIG(2);
  ECL_IO_OUT(3) = ENCODED_TRIG(3);
  ECL_IO_OUT(4) = ENCODED_TRIG(4);
```

```
trlo_ctrl --addr=1 --config=trlo_setup.trlo wiring
```

trloctrl – Command line

File with commands.

Execute cmd.

```
trlo_ctrl --addr=1 --config=trlo_setup.trlo wiring
```

Use: `alias trloctrl="trlo_ctrl --addr=1 --config=trlo_setup.trlo"`

```
trloctrl wiring
```

```
trloctrl --print-config
```
Dump TRLO II status.

```
trloctrl --clear-setup
```
Clear TRLO II.

```
trloctrl --mux-src-scalers
```
Debug scalers.

```
trloctrl "ECL_OUT(1)=ACCEPT_TRIG[1]"
```
Direct (without file)

```
trloctrl --show
```
Show all setup names.

```
trloctrl --ramtest
```
Test VME interface.

Example .trlo

(abridged)

```
/* Declare some variables. */

user_gate1 = ECL_OUT(1), ECL_OUT(2), ECL_OUT(4);
user_delay1   = 100 ns;
user_stretch1 = 1000 ns;

force_trig_no = 9;
                                /* Main setup command. */
COMMAND(wiring)
{
  DEADTIME_IN(1) = ECL_IO_IN(4);        /* TRIVA connections. */

  ECL_IO_OUT(1) = ENCODED_TRIG(1);
  ECL_IO_OUT(2) = ENCODED_TRIG(2);
  ECL_IO_OUT(3) = ENCODED_TRIG(3);
  ECL_IO_OUT(4) = ENCODED_TRIG(4);

  fast_busy_len = 1 us;
                            /* Busy before TRIVA deadtime. */
  GATE_DELAY(1) = MASTER_START;
  delay(1)   = user_delay1;             /* Prepared gates (delayed & stretched).
  stretch(1) = user_stretch1;              Using variable values. */
  user_gate1 = GATE_DELAY(1);
}
                                        /* Utility command. */
COMMAND(mapgate1) { LEMO_OUT(2) = GATE_DELAY(1); }
```

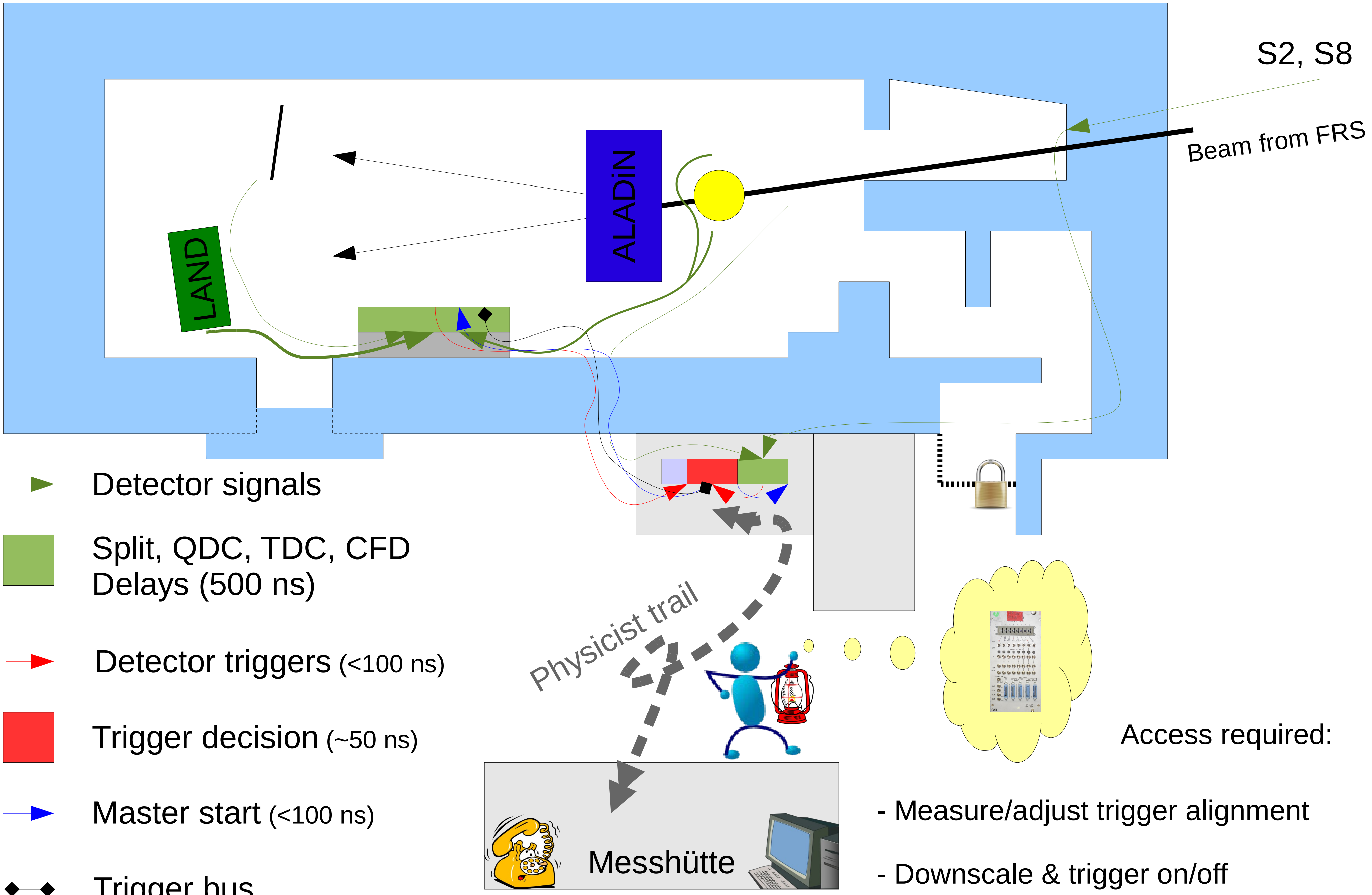
MUX_DEST = MUX_SRC

setup = value | time (with unit)

Next: Cave C

Cave C trigger (2004-)

(Analog to LAND@ Cave B -2004)

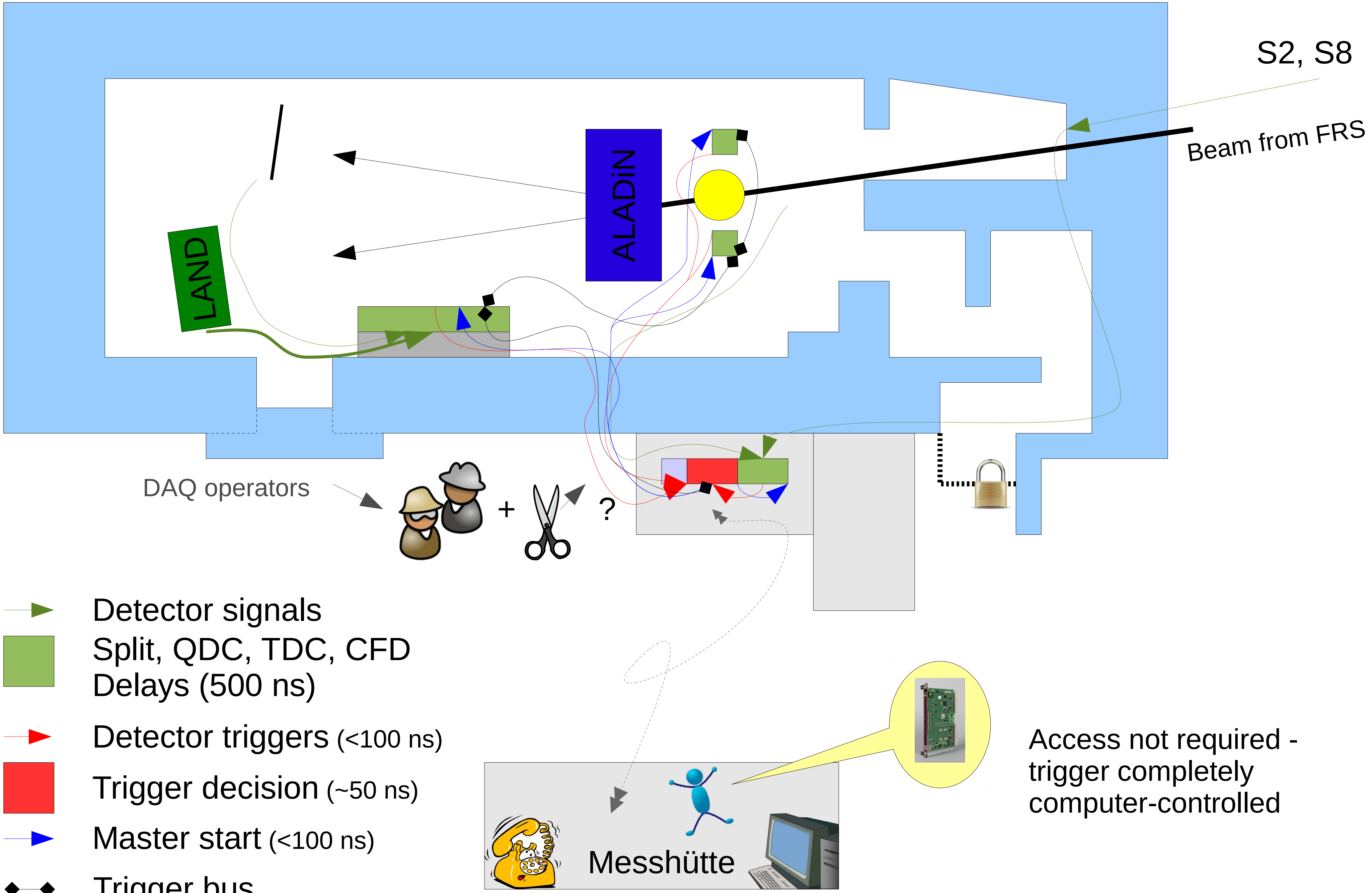


→	Detector signals
🟩	Split, QDC, TDC, CFD Delays (500 ns)
→	Detector triggers (<100 ns)
🟥	Trigger decision (~50 ns)
→	Master start (<100 ns)
◆—◆	Trigger bus

Access required:

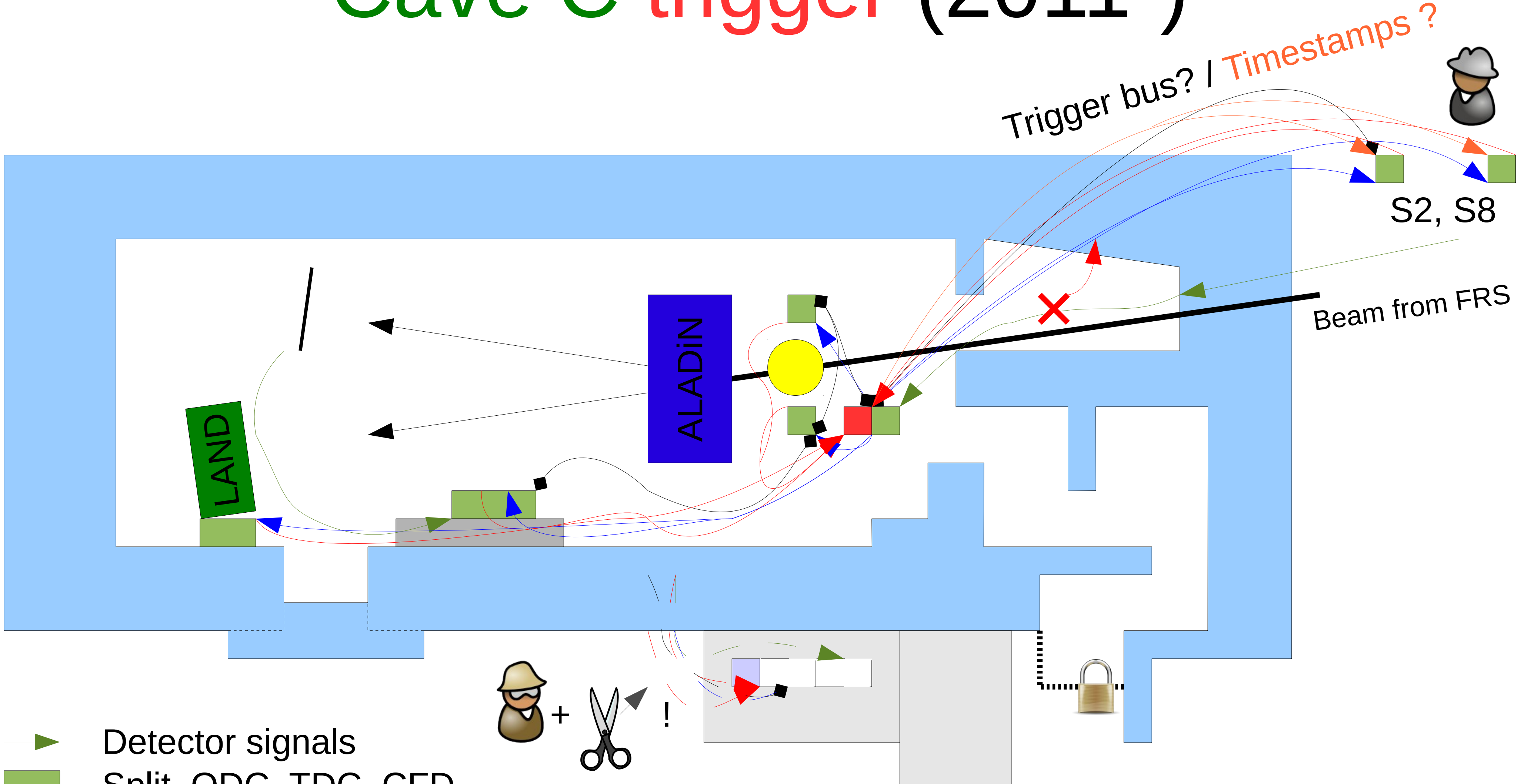
- Measure/adjust trigger alignment
- Downscale & trigger on/off

Cave C trigger (2010-)



──►	Detector signals
🟩	Split, QDC, TDC, CFD Delays (500 ns)
──►	Detector triggers (<100 ns)
🟥	Trigger decision (~50 ns)
──►	Master start (<100 ns)
◆─◆	Trigger bus

Cave C trigger (2011-)



→ Detector signals

🟩 Split, QDC, TDC, CFD
Delays (500 ns)

→ Detector triggers (<100 ns)

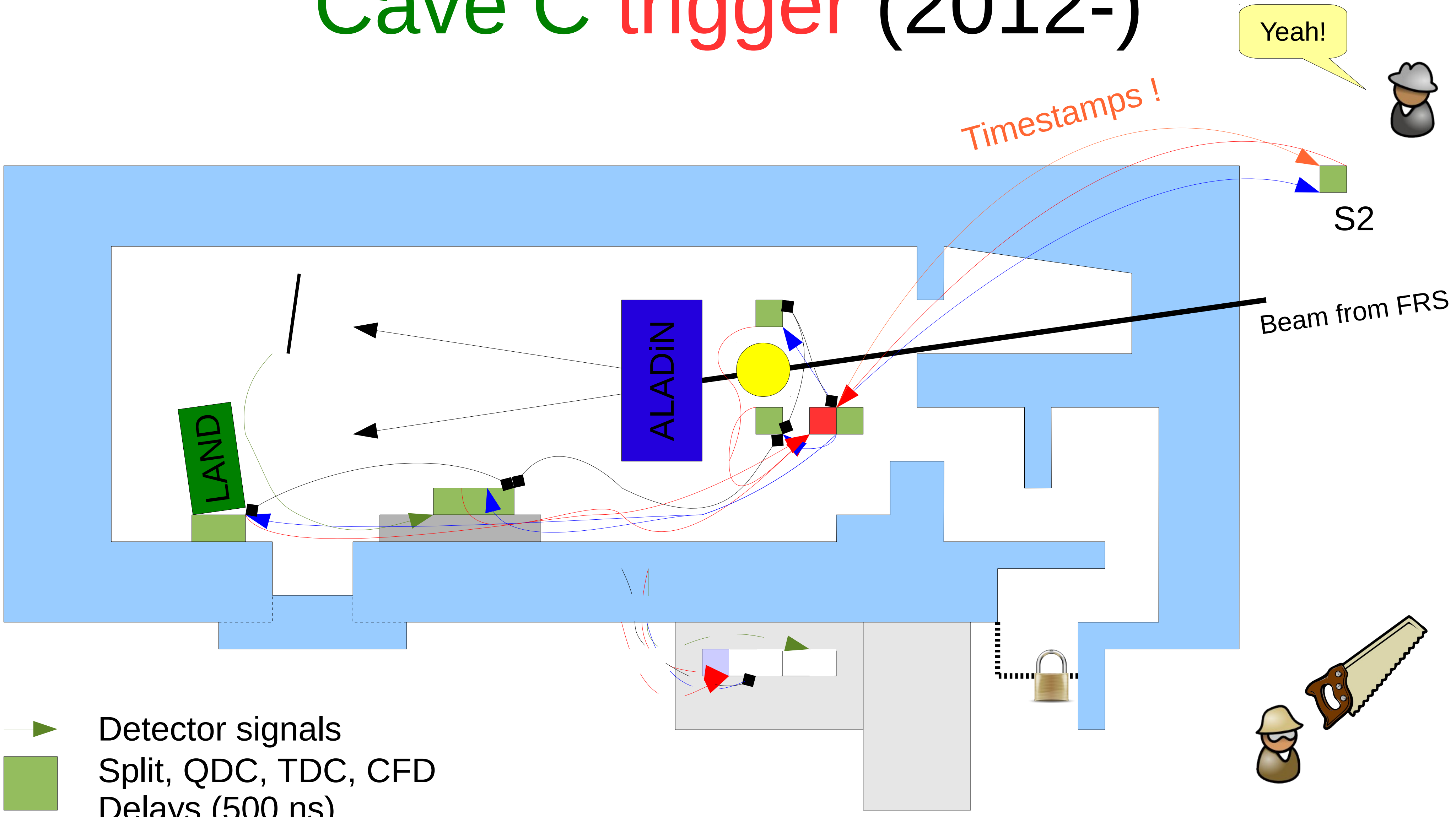
🟥 Trigger decision (~50 ns)

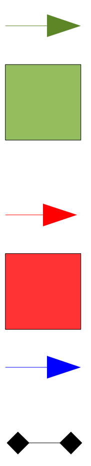
→ Master start (<100 ns)

◆–◆ Trigger bus

Trigger fully computer-controlled
→ moved into Cave!

Cave C trigger (2012-)





Detector signals

Split, QDC, TDC, CFD
Delays (500 ns)

Detector triggers (<100 ns)

Trigger decision (~50 ns)

Master start (<100 ns)

Trigger bus

Separate DAQ @ S2, merge using time-stamps.

2012: Trigger bus Cave C → S2 ?

Unexpected problem:

Trigger bus lost signal integrity Cave C → S2 (ground levels)

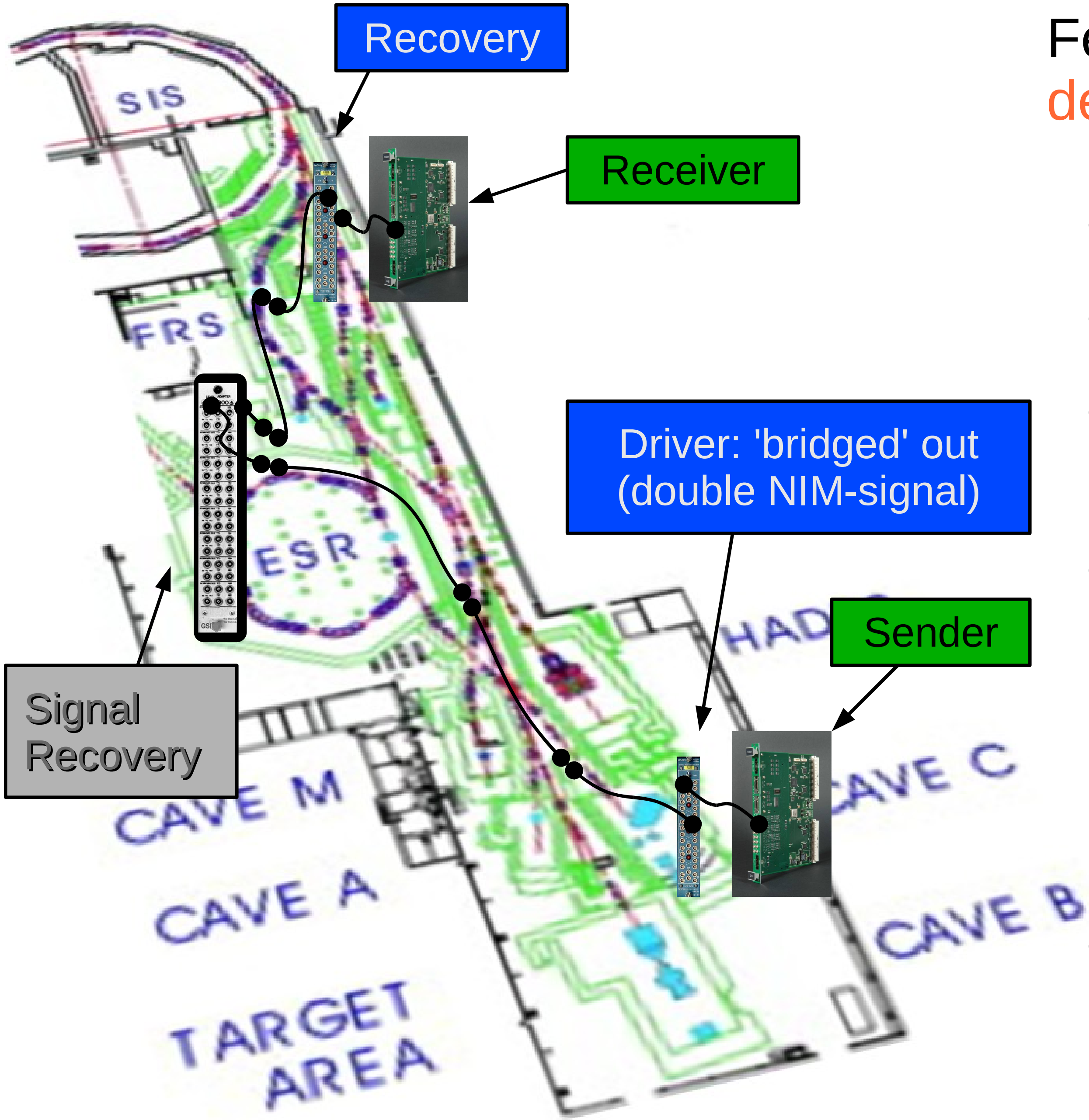
=> Common dead-time domain
with TRIVA @ S2 not possible.

=> Time-stamp distribution with TITRIS modules not possible.
(uses same type bus cable)

2012: Time-stamp distribution with two spare TRIDI1
modules and TRLO II. Using 'any' cable.

Timestamps Cave C → S2 !

2012: Serial timestamps Cave C - (FRS) - S2



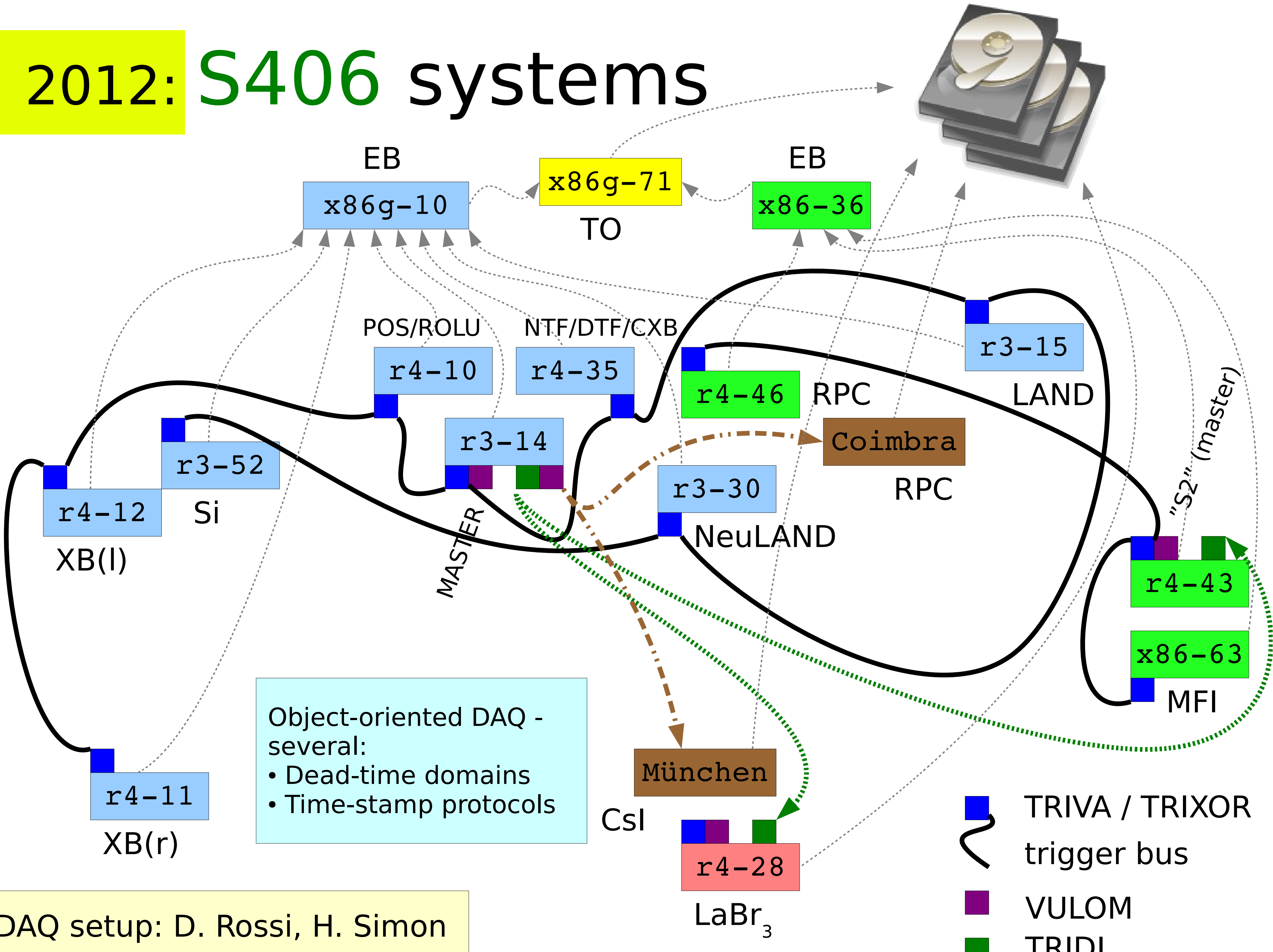
Features for easy deployment:

- Uni-directional protocol
- 1 cable of 'any' kind
- (a trigger also needed) (2nd cable)
- No 'initialisation' - easy setup:

  1. Start sender,
  2. Follow signal (scope),
  3. Receiver auto-sync
- Loss of ≤ 3 protocol cycles → still in sync

Next: TRIMI – TRIVA mimic

2012: S406 systems

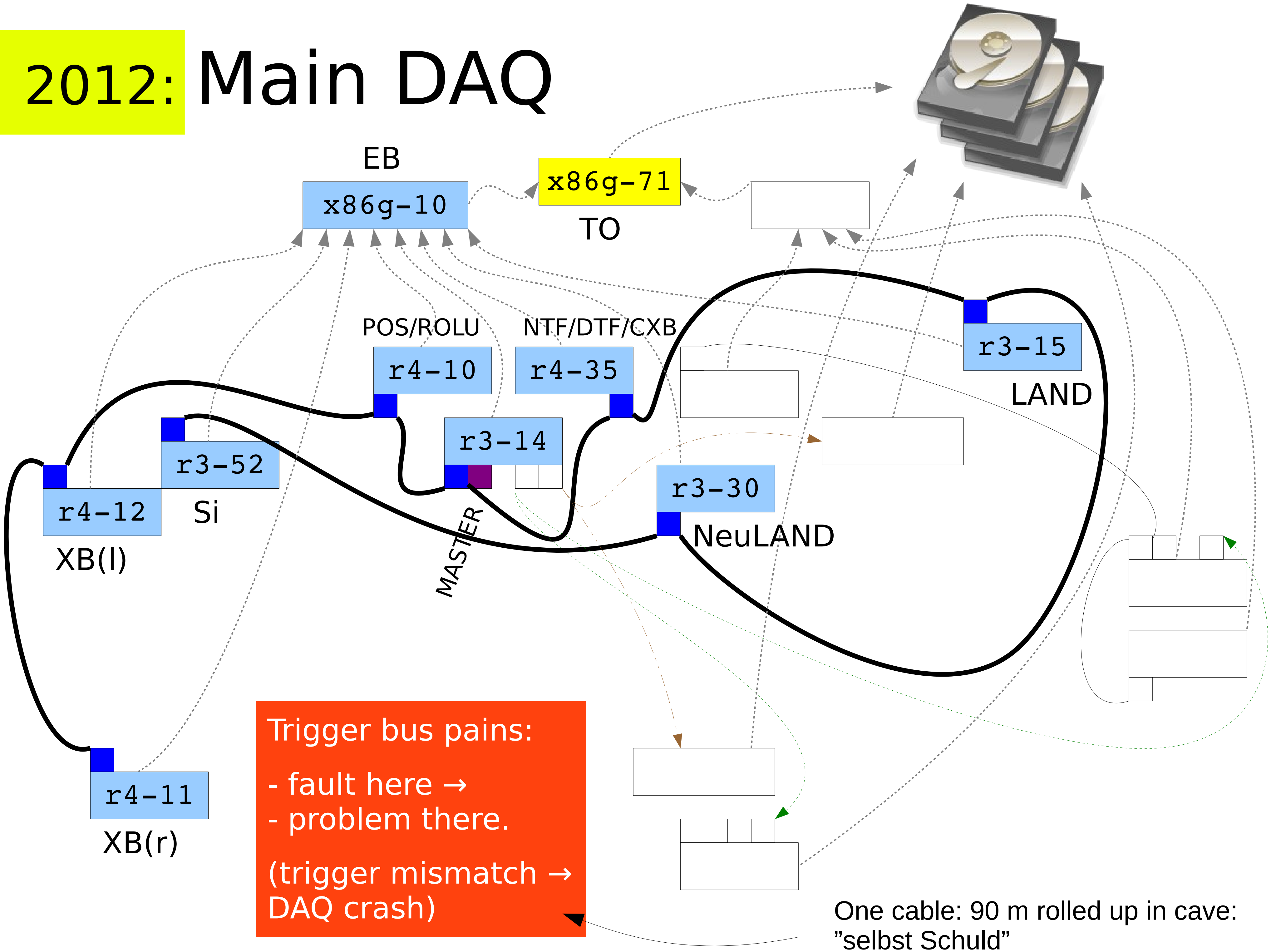


Object-oriented DAQ - several:
- Dead-time domains
- Time-stamp protocols

🟦	TRIVA / TRIXOR
∫	trigger bus
🟪	VULOM
🟩	TRIDI

DAQ setup: D. Rossi, H. Simon

2012: Main DAQ



Trigger bus pains:

- fault here →
- problem there.

(trigger mismatch → DAQ crash)

One cable: 90 m rolled up in cave: "selbst Schuld"

“Don't get me wrong:”

The TRIVA is great!

[VME Trigger Synchronisations Modul]

TRIVAx robustness

MBS:
• Uncompromising event synchronisation
• Stability

Lately, some scalability issues:

• Trigger bus cable at long distances?
• Rapid reconfiguration of multi-branch system
(slave addition and removal during experiment setup)

Any alternative needs to be as stable...

TRLO II TRIVA mimic 'link': Plan

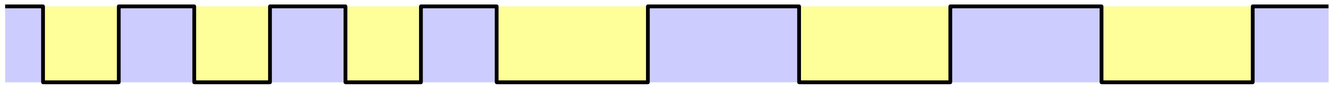
- Serial protocol → Minimise cable needs.
  → 'Any' cable type. Including optical.
- Unidirectional → Easy deployment – autosync.
  → Support tree-like cable arrangement.

(Deadtime return also needed.)
(Simple 'OR' enough; with VULOM as fan-in → individual monitoring.)

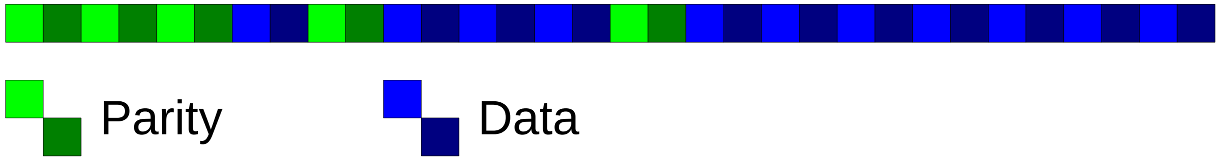
- Backwards compatible. (VME register space) → Use MBS unchanged.
- Clock frequency independent. → Usable on other boards with FPGA/CPLD control.
- Aux. software messages. → Insert subsystems on-the-fly.

Serial protocol

- Pulses 0 or 1.
  50 % 1s, 50 % 0s.

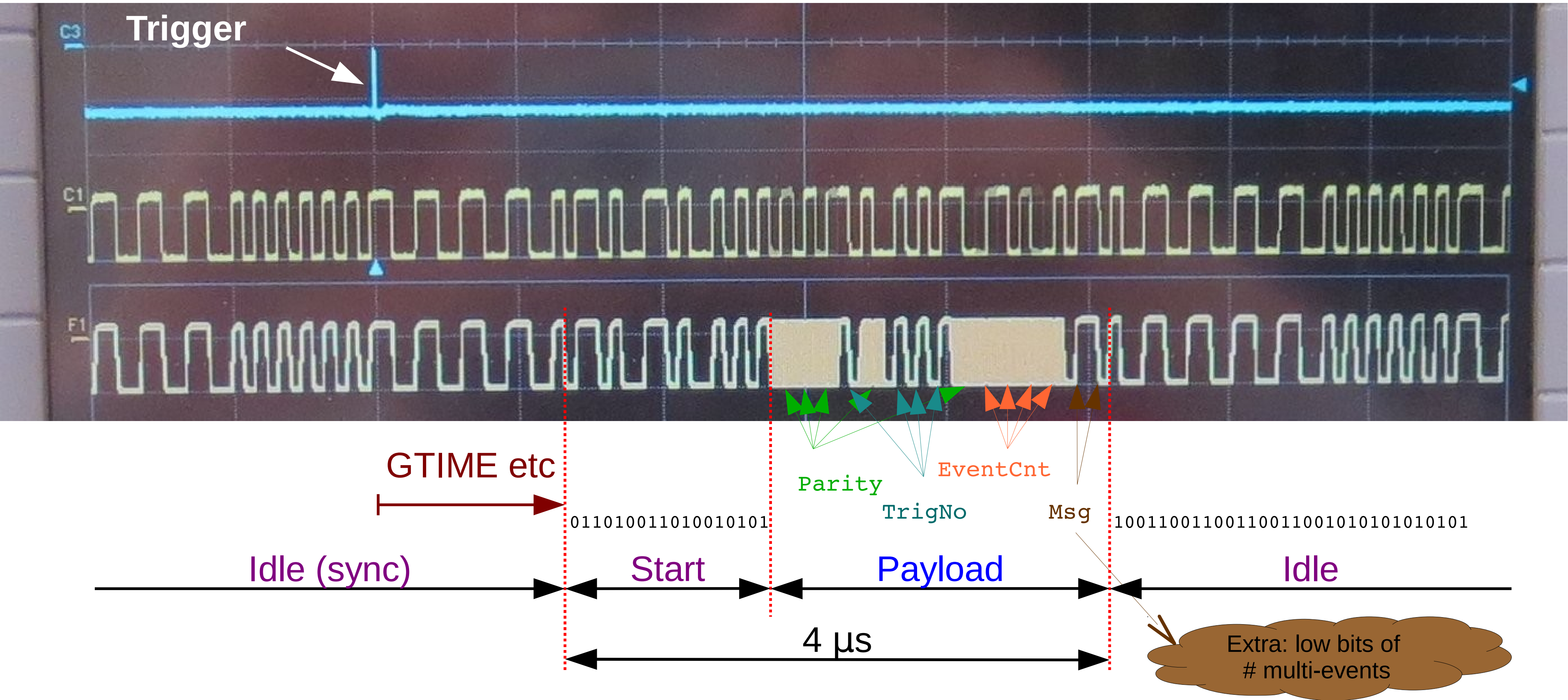


- Each payload bit sent as two pulses; second inverted.
- Never more than two same (1 or 0) in a row.
  → never more than 4 slots between same edge.
  (Used for first rough frequency lock.)
- Receiver: phase and frequency tracking.
- Special (idle) patterns for synchronisation.
  → Automatic inversion handling.
- Payload forward error correction: [16,11] Hamming Code
  → together with bit duplication: 3-bit errors correctable.

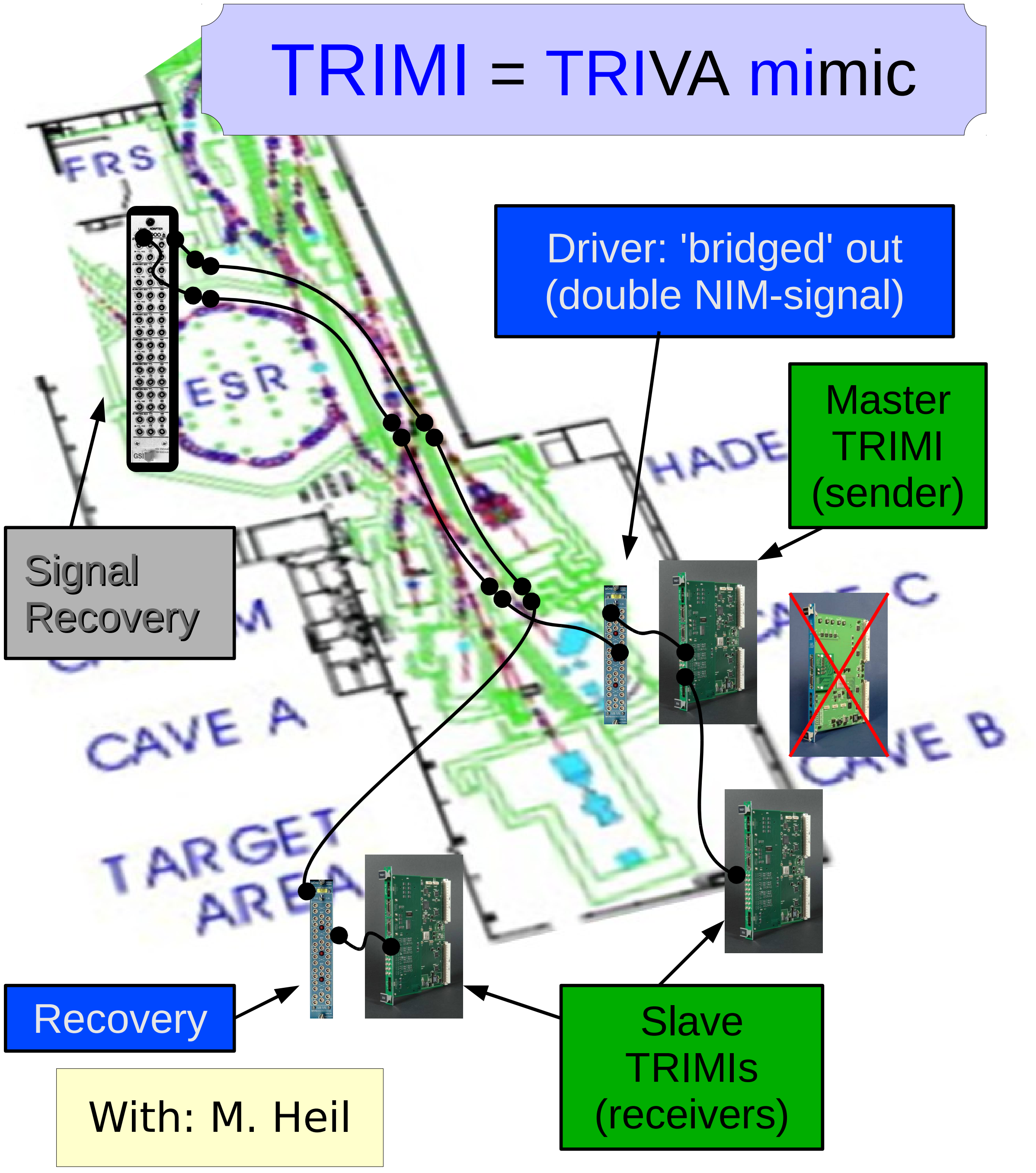


TRIMI protocol

- Continous idle pattern (→ recv. freq./phase sync)
- Trigger payload after start pattern
- Other header pattern for reset and software message (8 bits)



Serial trigger test Cave C - (FRS) - Cave C



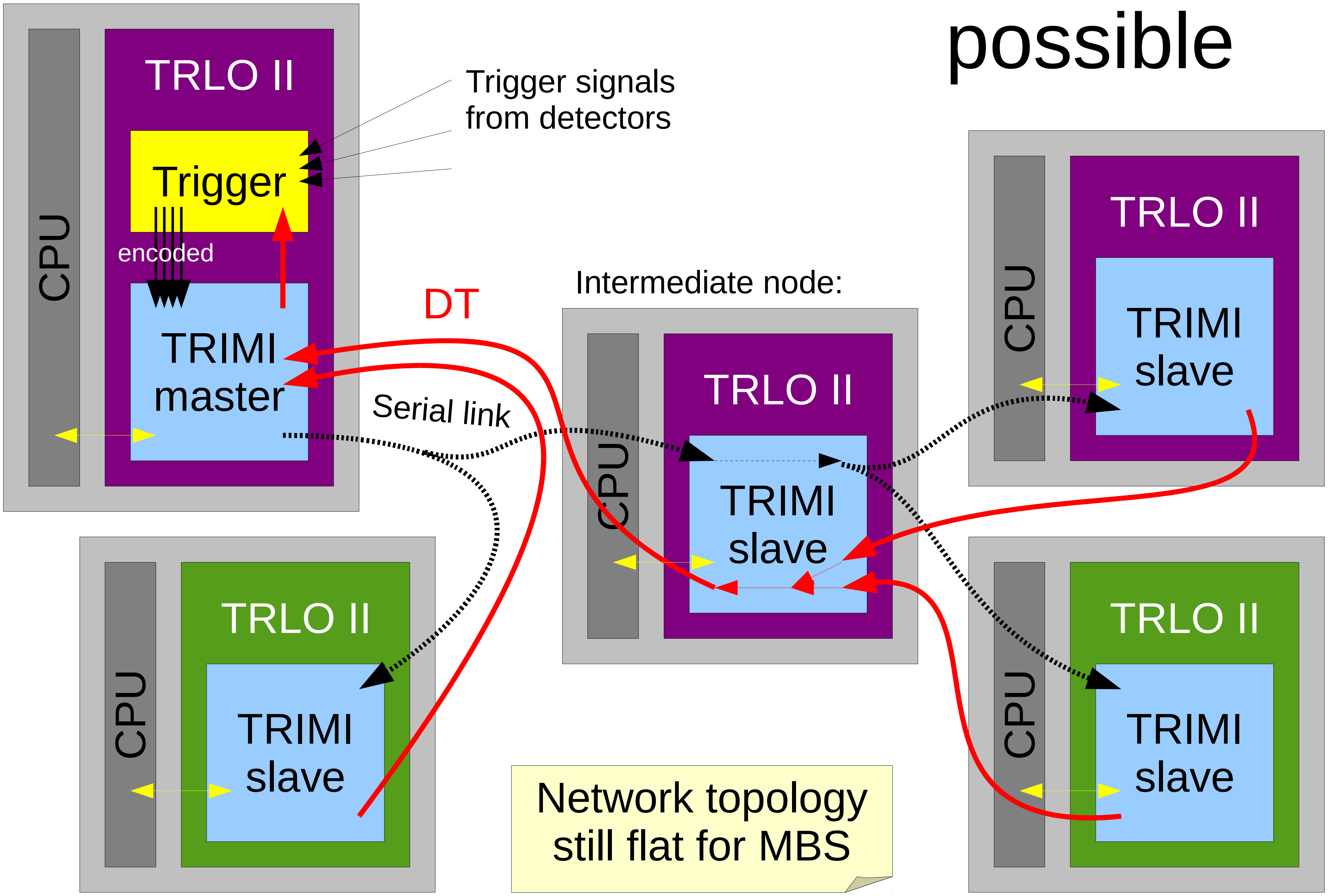
Easy deployment:

- Uni-directional protocol
- 1 cable of 'any' kind
- (DT return also needed) (2nd cable)
- No 'handshake' startup - easy setup:

  1. Start sender,
  2. Follow signal (scope),
  3. Receiver auto-sync ('any' frequency)

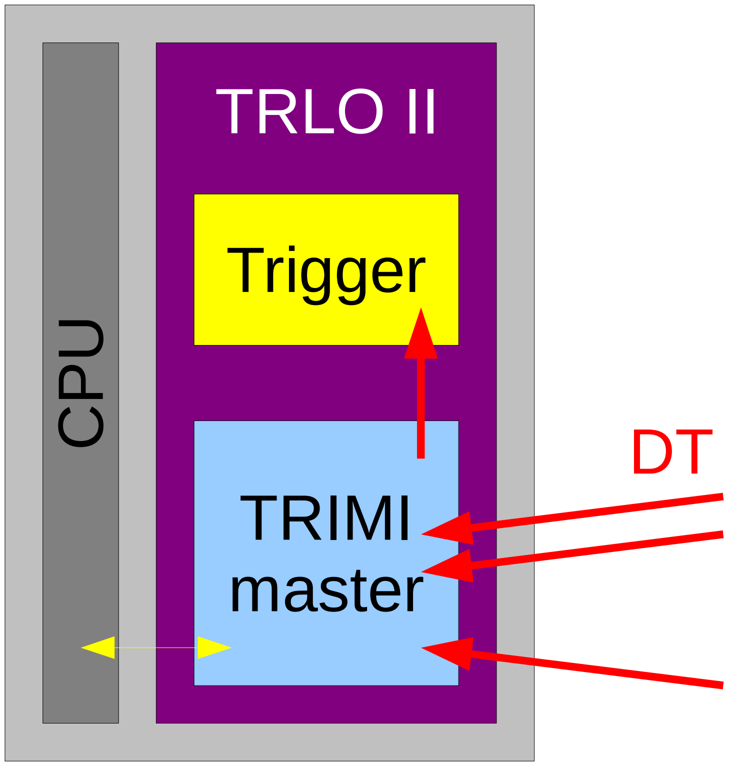
- Loss of ≤ 3 bits/msg → error-correction

TRIMI - tree topology possible



Network topology still flat for MBS

DT monitor (per in-bound slave)



Record time for:

- TRIMI total deadtime
  - begin
  - release
- TRIMI local deadtime release
- Each slave deadtime release

Not out of order, just measurement precision

```
TOT rel  0000d5
 #2 rel  0000d5
TOT on   0000d6
LCL rel  0000eb
 #2 rel  0000fd
TOT rel  0000fe
TOT on   000117
LCL rel  00012e
TOT rel  00013f
 #2 rel  00013f
```

unit: 32 x 10 ns

Q: what can one do with ~200 LUTs/FFs + 1 RAM block?

A: A deadtime monitor!

DT analysis

Slave with IRQ, master polled:
`r4-cth1` `mvme5500-1`

```
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  8.3  TOT: 13.7
#2: 12.8  LCL: 15.0  TOT: 19.6
#2: 12.8  LCL: 15.5  TOT: 20.0
#2: 12.8  LCL: 15.9  TOT: 20.4
#2: 12.8  LCL: 16.1  TOT: 20.6
#2: 12.8  LCL: 10.4  TOT: 15.6
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
#2: 12.8  LCL:  7.3  TOT: 12.8
```

Slave polled:

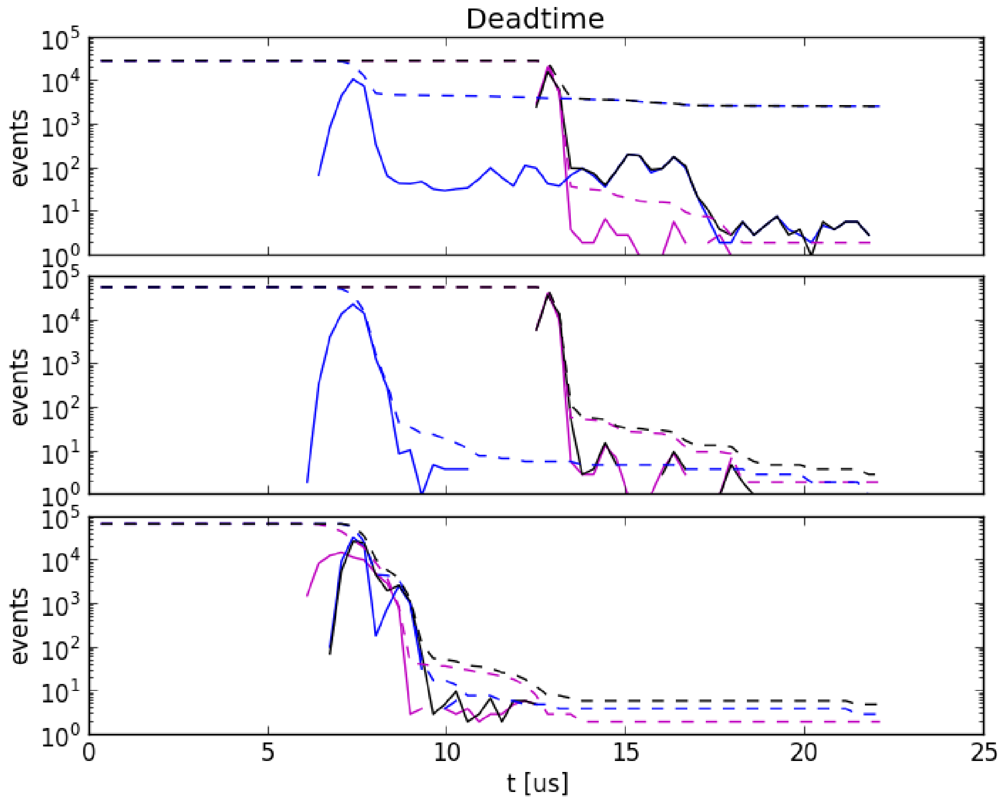
```
#2:  7.2  LCL:  7.5  TOT:  7.6
#2:  7.2  LCL:  7.5  TOT:  7.6
#2:  7.2  LCL:  7.5  TOT:  7.6
#2:  7.2  LCL:  7.5  TOT:  7.6
#2:  7.2  LCL:  7.5  TOT:  7.6
#2:  7.2  LCL:  7.5  TOT:  7.6
#2:  7.2  LCL:  7.5  TOT:  7.6
```

One line per second.

Average DT per part [μs].

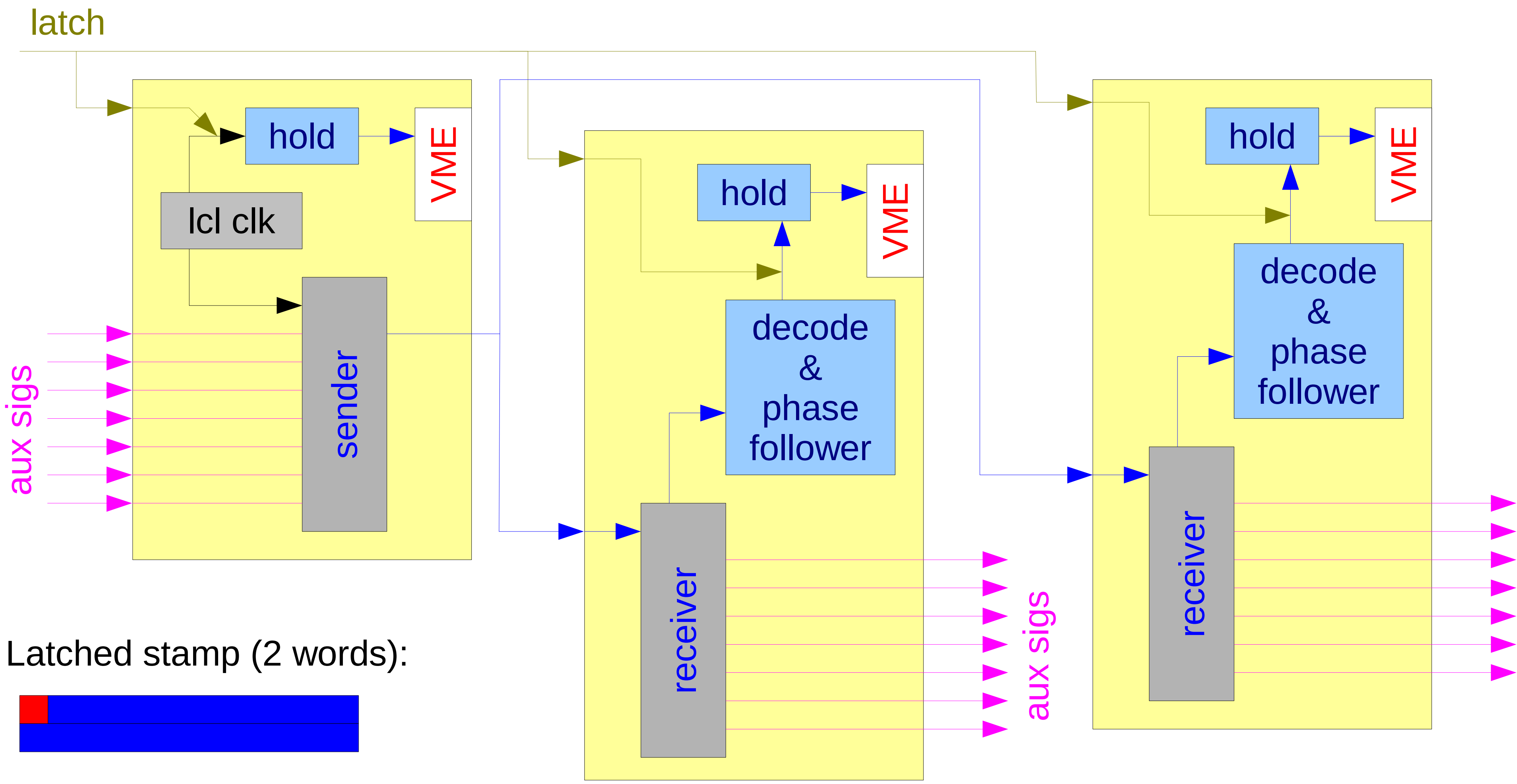
Collected with test program (`trlo_trimi_test`).

Master running extra CPU-consuming task.

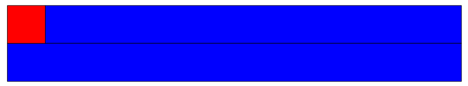


Serial timestamps revisited...

Serial timestamps with mux



Latched stamp (2 words):

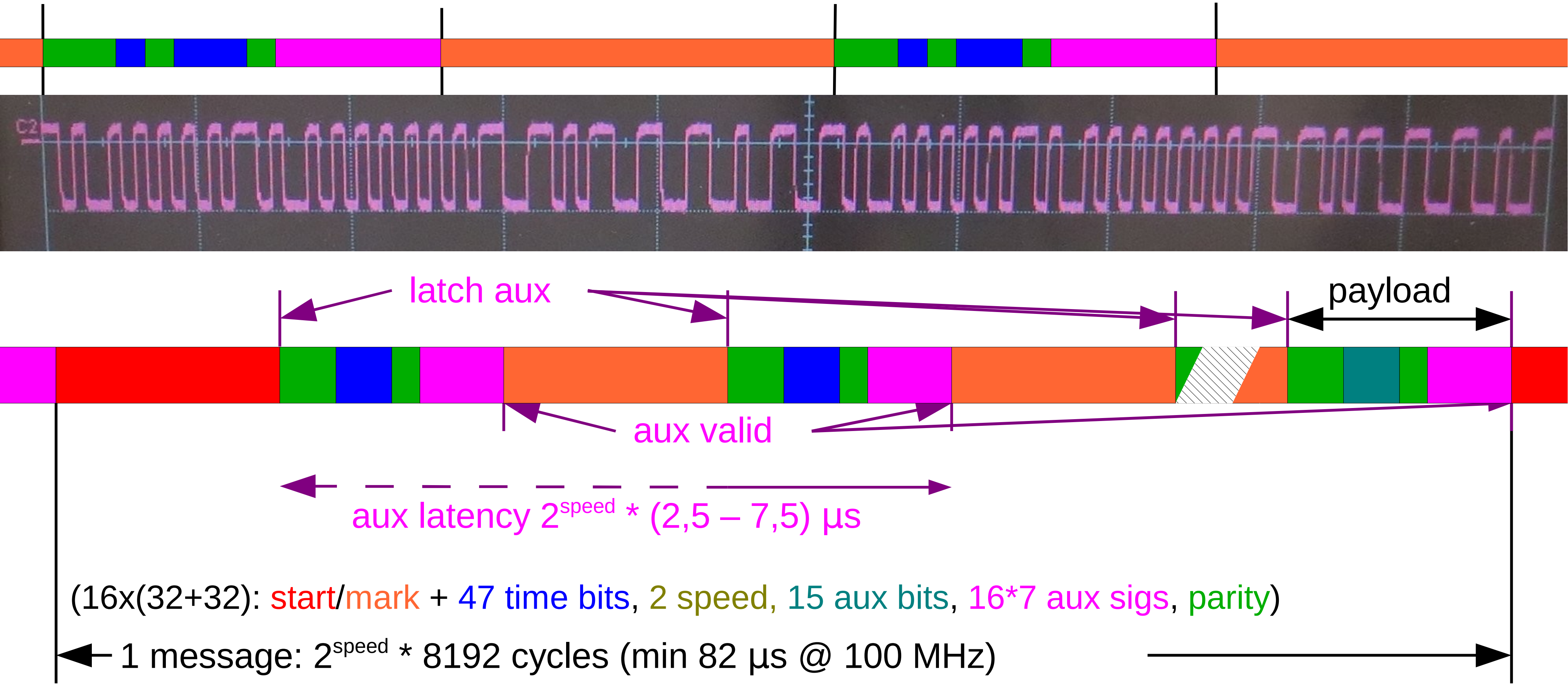


- synchronised?
- time of latch

Simplified from previous version. No CPU-side post-processing.

Serial timestamp protocol

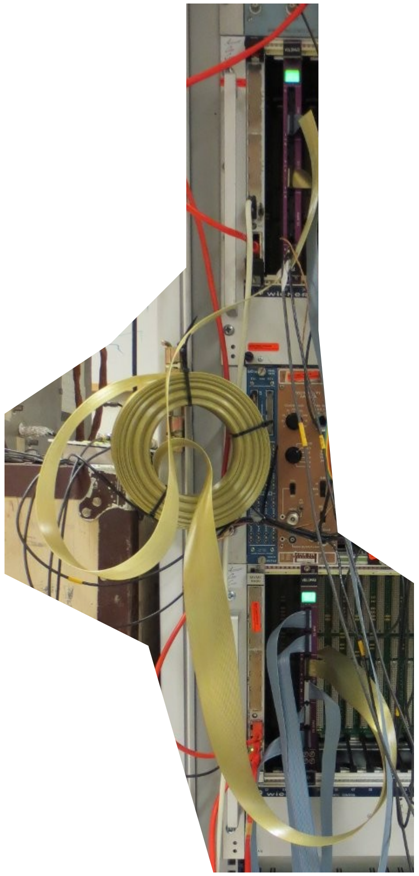
- Continous bitstream (→ recv. freq./phase sync)
- Each payload has 7 aux signals and 4 bits of time, 5 parity
- Header pattern for sync, different before first payload



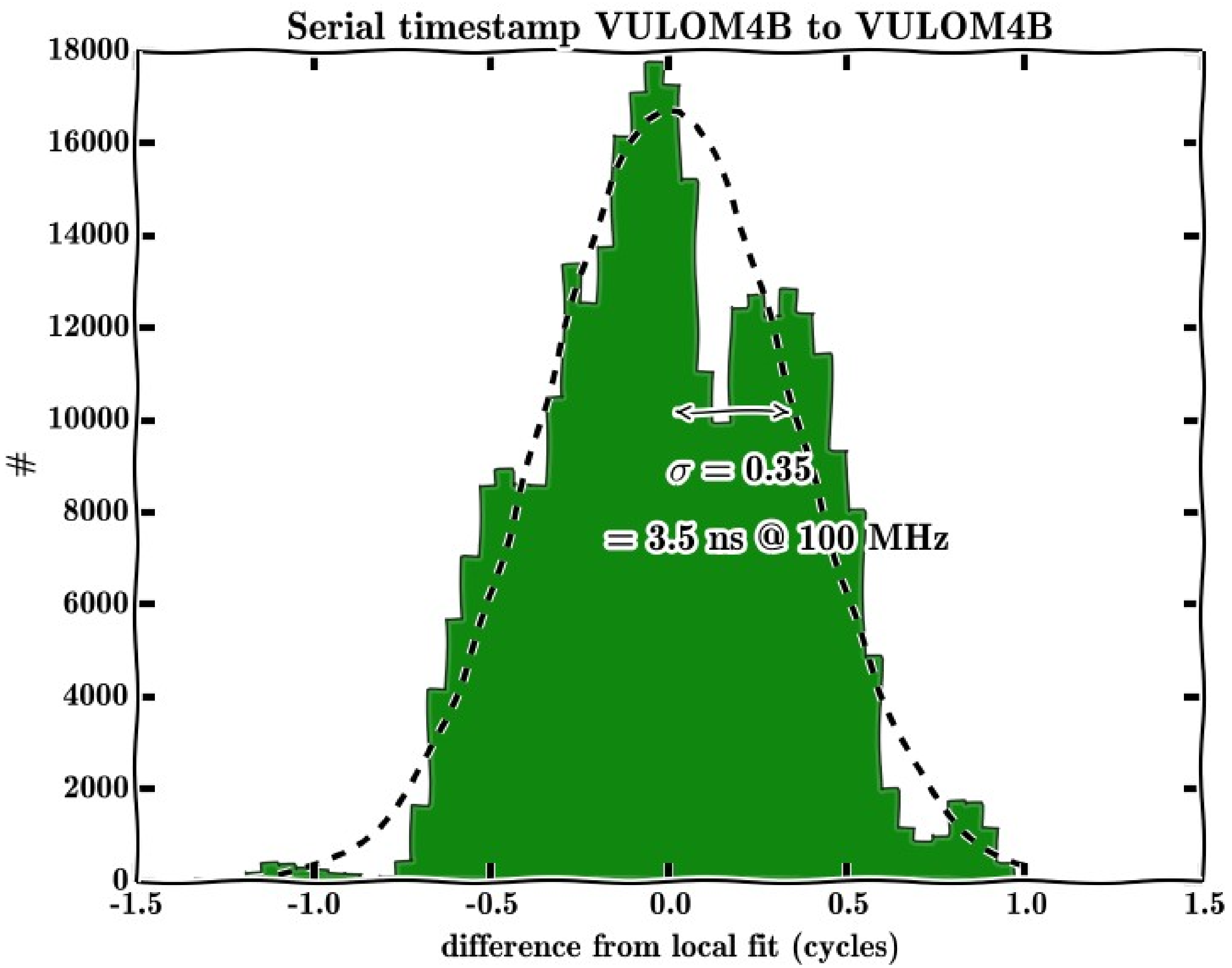
(16x(32+32): start/mark + 47 time bits, 2 speed, 15 aux bits, 16*7 aux sigs, parity)

1 message: 2^{speed} * 8192 cycles (min 82 µs @ 100 MHz)

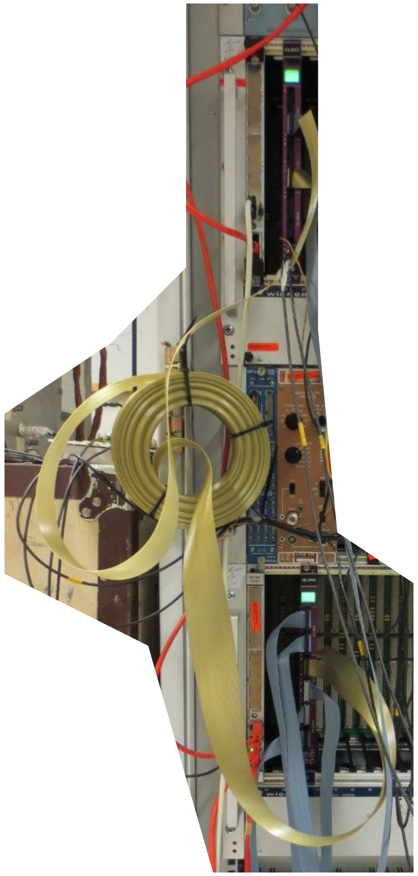
Serial timestamp precision

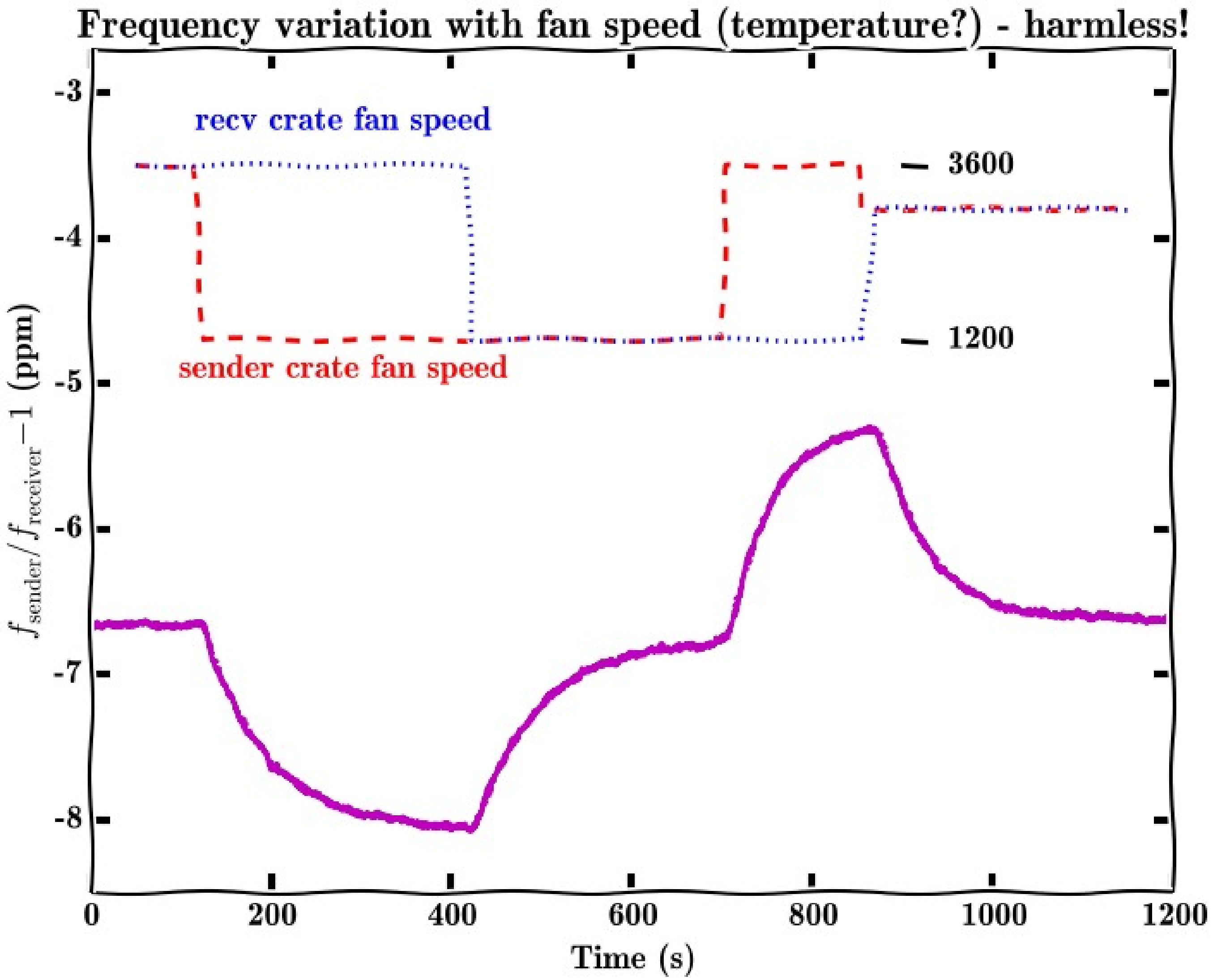


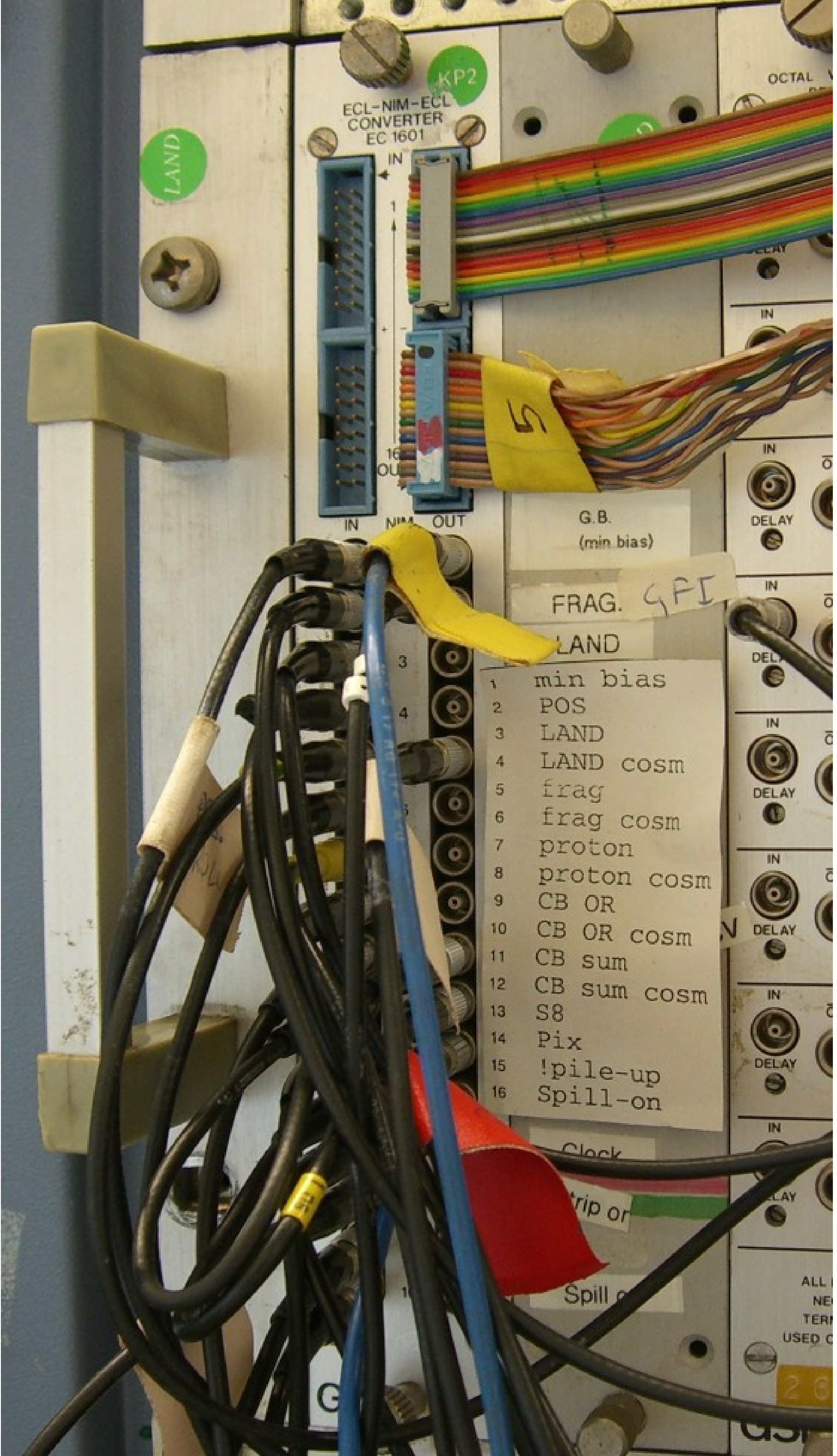
Humps due to
receiver following
signal by
± 0.5 local clocks?



For FUN – oscillator vs. oscillator







Finale!

Thank you!

With thanks to: J. Hoffmann, J. Frühauf, W. Ott, N. Kurz, H. Simon, A. Henriques, M. Heil, B. Löher, A. Charpy, C. Forssén...

FPGAs are FUN!

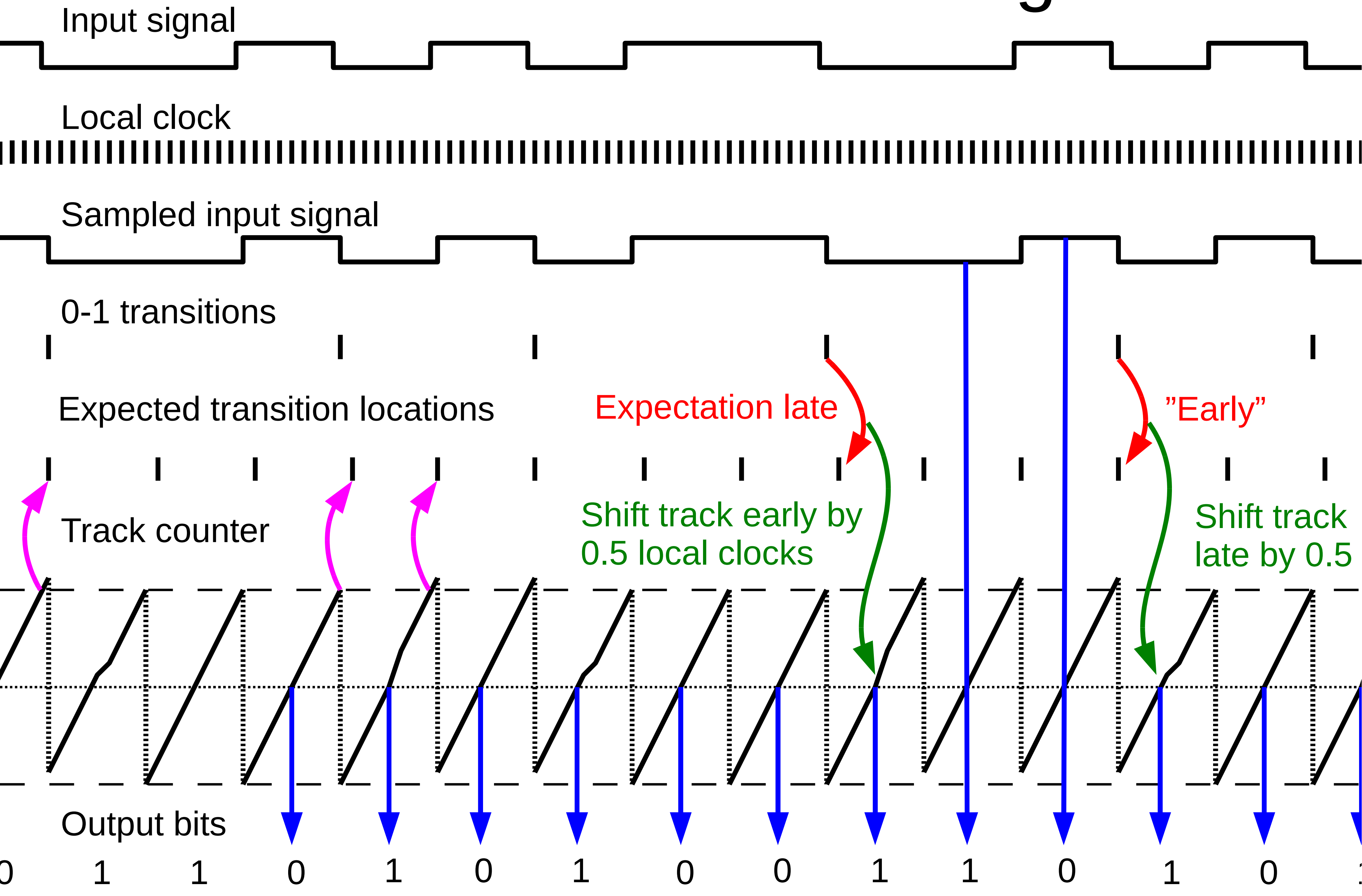
- although getting the v2 serial messages to work has been a loooong...

http://fy.chalmers.se/~f96hajo/trloii/

Live by the compiler timing messages!

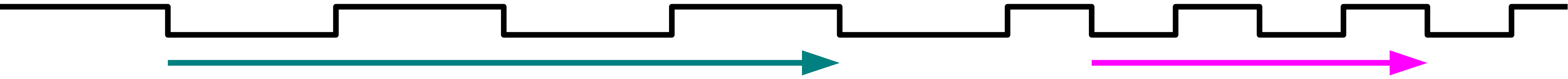
Spare: Serial receiver internals

Phase tracking



First find the frequency

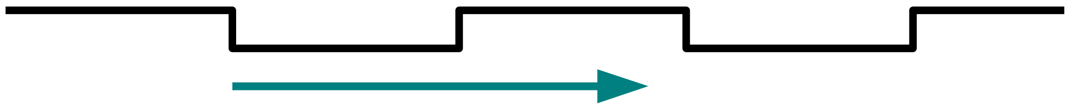
Input signal



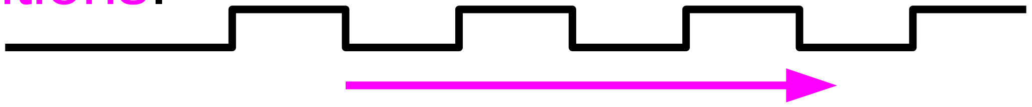
Idle pattern ensures both 2x maximum (2x4 periods) and 2x minimum (2x2 periods), within at most 32 0-1 transitions

Restart track counter every measurement

1. Adjust frequency ↑, to have stretch of **no** transition during 32 estimated 4-periods.

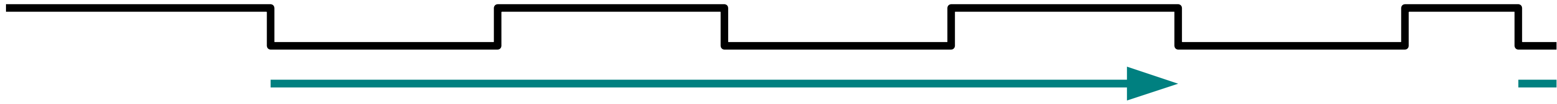


2. Adjust frequency ↓, to have stretch with 2 min-transitions.

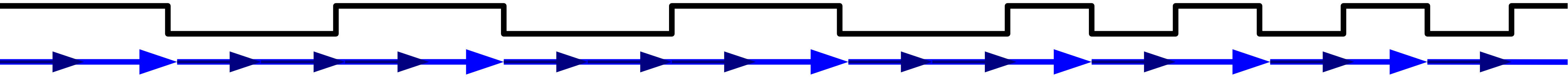


From arb. wrong

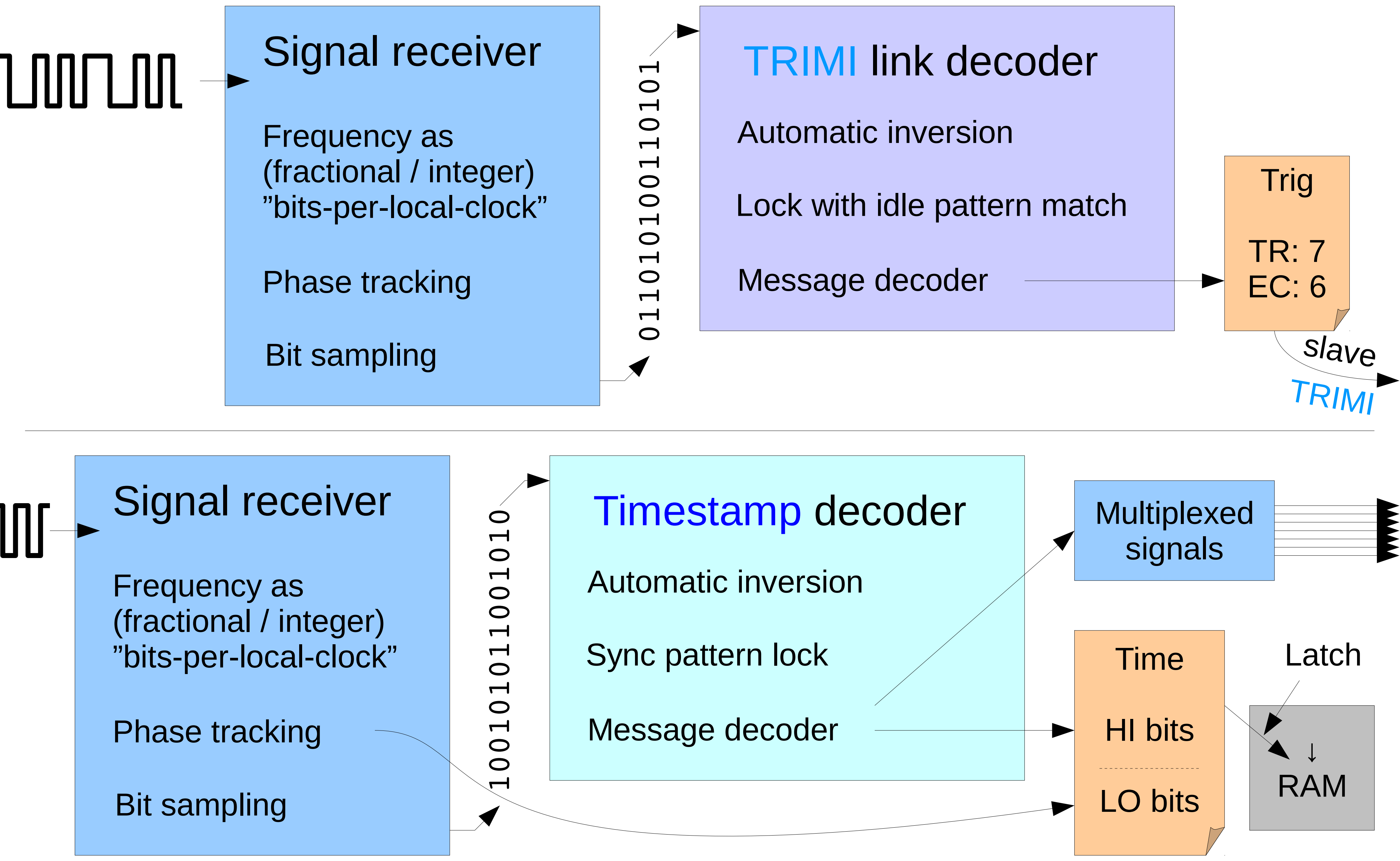
3. Adjust frequency ↕, to match 2x4-periods.



4. Phase tracking. (Abort to 1. only on repeatedly bad bit pattern.)



Serial receivers


Signal receiver
Frequency as (fractional / integer) ”bits-per-local-clock”
Phase tracking
Bit sampling
011010100110101
TRIMI link decoder
Automatic inversion
Lock with idle pattern match
Message decoder
Trig
TR: 7
EC: 6
slave
TRIMI
Signal receiver
Frequency as (fractional / integer) ”bits-per-local-clock”
Phase tracking
Bit sampling
100101011001010
Timestamp decoder
Automatic inversion
Sync pattern lock
Message decoder
Multiplexed signals
Time
HI bits
LO bits
Latch
↓
RAM