TTD status / updates

Mikihiko Nakao (KEK-IPNS)

mikihiko.nakao@kek.jp

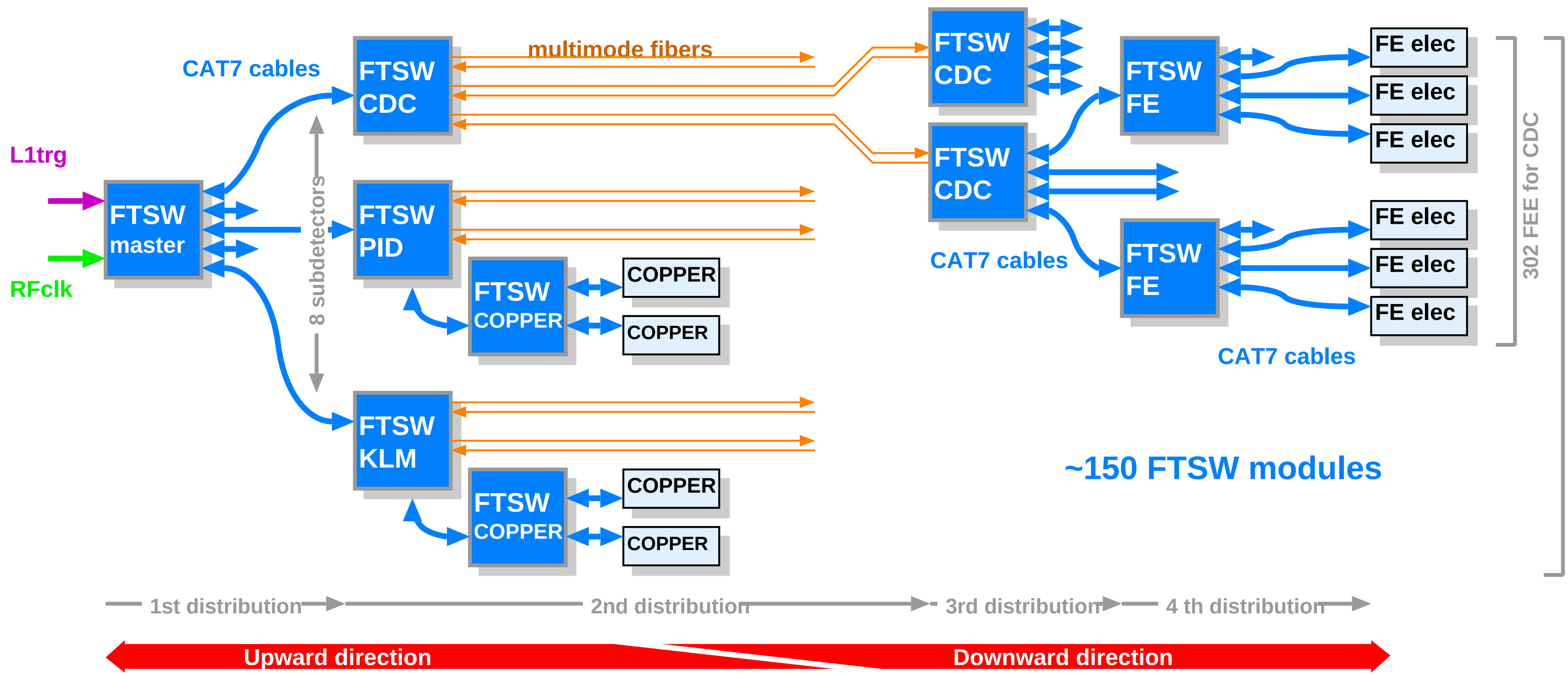
January 17, 2012

Outline

disclaimer: this talk file contains lots of personal memorandum which would be too much technical detail, sorry.

1. **Introduction**
2. **FTSW production/firmware status**
3. **Timing properties update**
4. **What will be distributed by TTD**
5. **What will be collected by TTD**

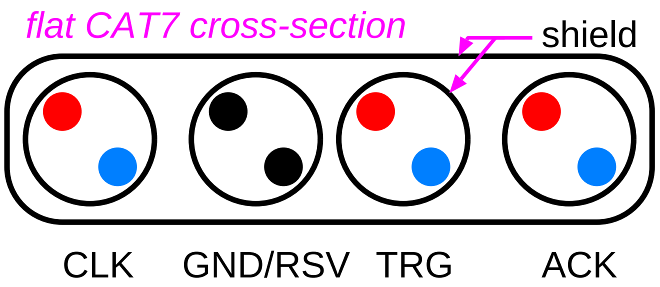
Task of TTD



- **Distribute L1 trigger and system clock to ∼1000 nodes**
- **Fast control** — reset broadcast, partitioning, command to single subsystem, collect status info
- **Single type of module — FTSW** [Frontend Timing SWitch] with a few types of additional daughter cards

Signals and Cables

- **3 LVDS pairs minimum**
  - **CLK —** system clock, 127.216 MHz (SuperKEKB RF div by 4)
  - **TRG —** encoded trigger and other downward signals
  - **ACK —** encoded flow control and other upward signals
- **High-grade (Category 7) LAN cable with RJ-45 connector**
  - **Flat cable (2.2mm thickness), commercially available**
  - **Doubly shielded**
  - **Up to 15 m length due to attenuation / degradation**



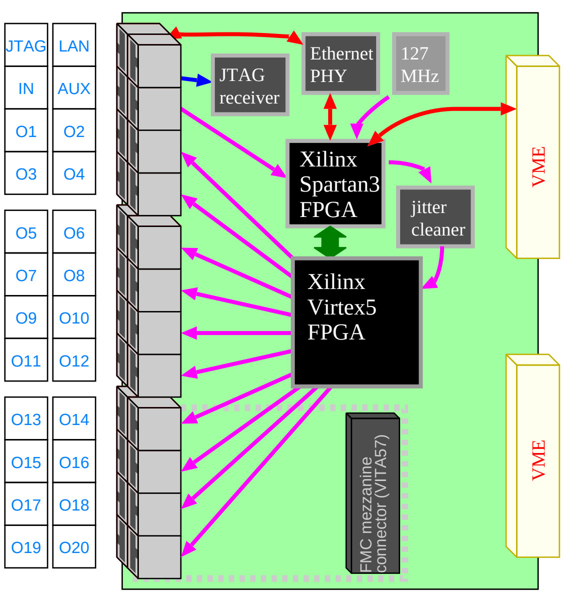
- **Optical fiber (multimode LC-LC pair)**
  - **2 pairs for 3 signals (almost no degradation)**
  - **1 pair with clock recovery circuit (worse jitter)**

Requirements

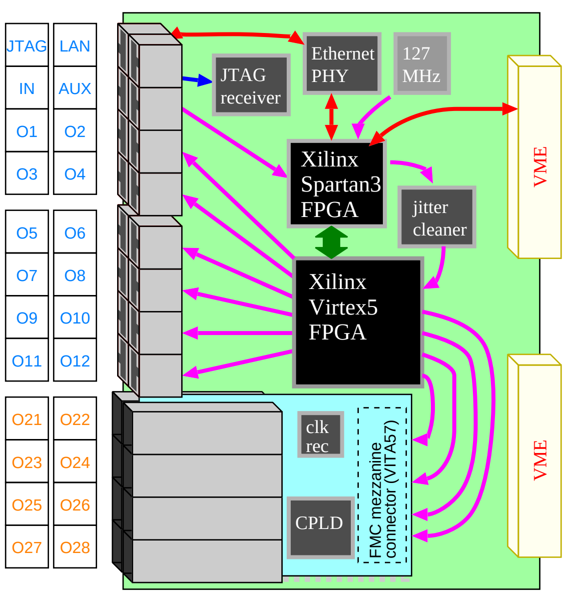
1. **$\sim$1100 destinations**
   — 600 (CDC, BPID, EPID) are inside the detector
   — large multiplication factor, compact cable
2. **5 μs total latency for SVD, $\sim$0.5μs for TTD**
   — encoding / decoding (for compact cable) takes time
   — homemade encoder/decoder to bypass intermediate steps
3. **BPID needs $<$20 ps jitter, Belle2link needs $<$30 ps jitter**
   — degradation by long metal cable
   — degradation by clock recovery
   — jitter cleaner and careful design

FTSW VME module

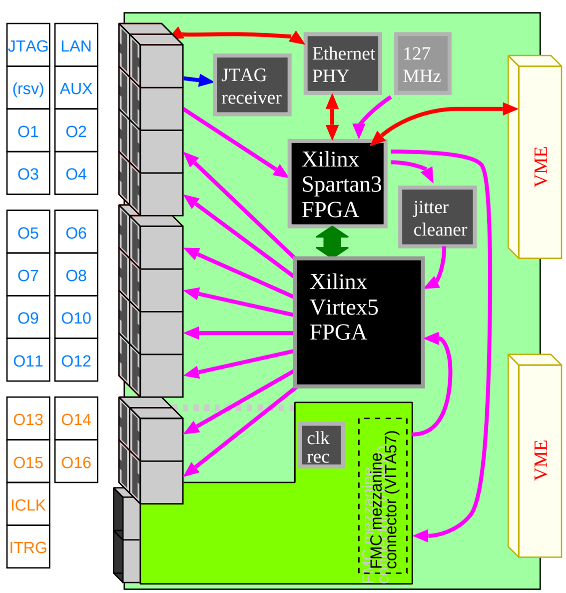
- **Small FPGA (Virtex5 LX30) for 1-to-20 distribution**
  - **6U VME 2 slot width**
  - **Special care to minimize clock jitter**
- **1 PCB for 3 types, with two optical daughter boards**
  - **1 CAT7 in** → **20 CAT7 out** [type-S]
  - **1 CAT7 in** → **8 optical out** **+** **12 CAT7 out** [type-P (FT optical port)]
  - **1 or 2 optical in** → **16 CAT7 out** [type-R (FT optical receiver)]



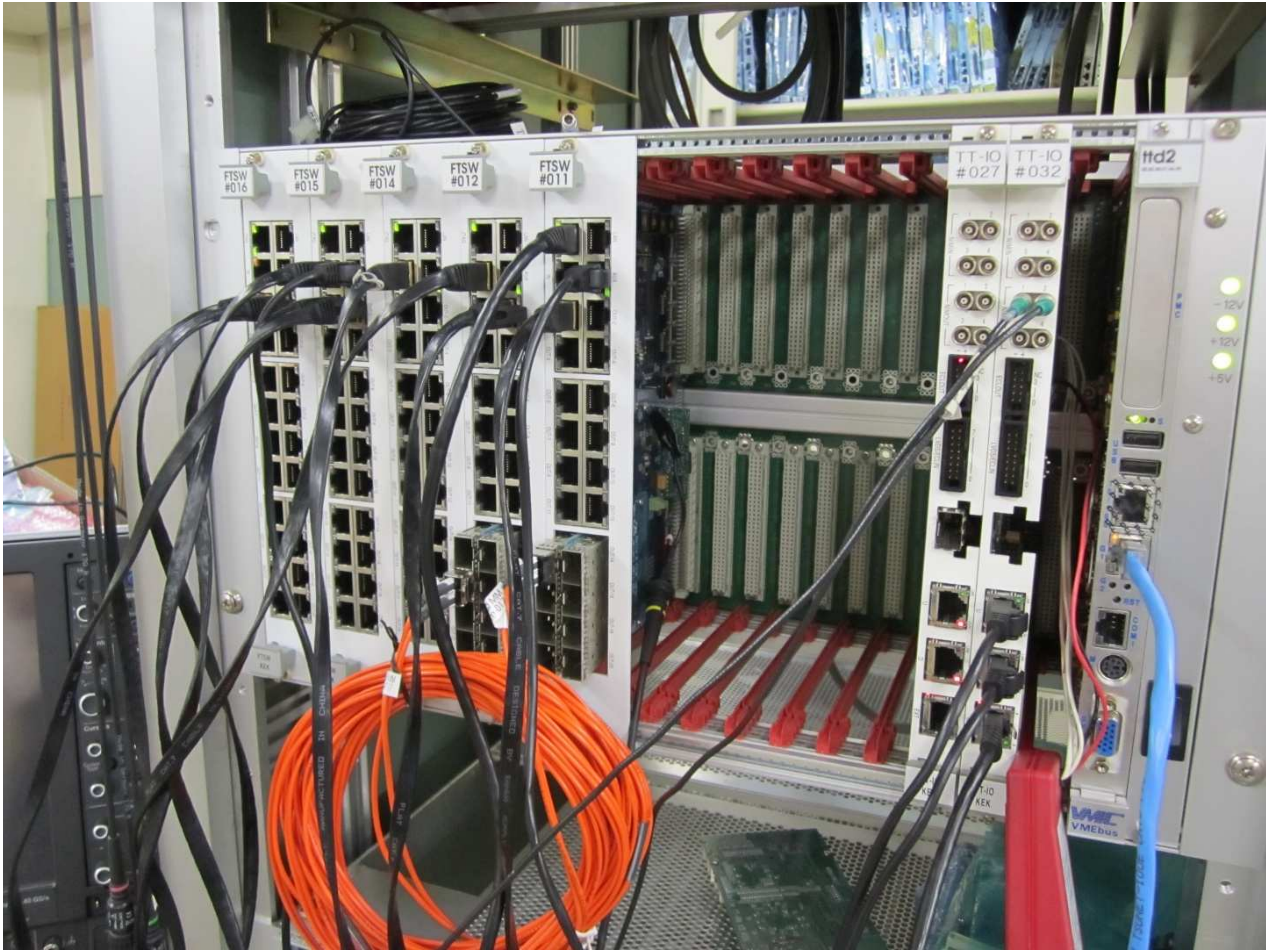
FTSW-S



FTSW-P + FTOP



FTSW-R + FTOR


FTSW #016
FTSW #015
FTSW #014
FTSW #012
FTSW #011
TT-IO #027
TT-IO #032
ttd2
-12V
+12V
+5V
VMEbus

FTSW production

- **1000 pieces of 127.216 MHz oscillators delivered**
  - 68 were used for FTSW
  - 8 were given to trigger group
- **68 FTSW boards (version 2.1) produced, 80+ more needed**
  - 24 FTSW type-S — with 24 port
  - 14 FTSW type-P — with 16 port and FMC slot
  - 20 FTSW type-R — with 20 port for optical receiver
  - 2 FTSW type-P/no-5V — with 16 port, for ECL crate with no 5V
  - 8 FTSW type-R/no-5V — with 20 port, for ECL crate with no 5V
- **14 FTOP boards (version 1.0) produced (completed)**
- **FTOR boards**
  - Need to understand the clock recovery with FTOP
  - Plan was to design in this FY, time is running out. . .

FTSW hardware test status

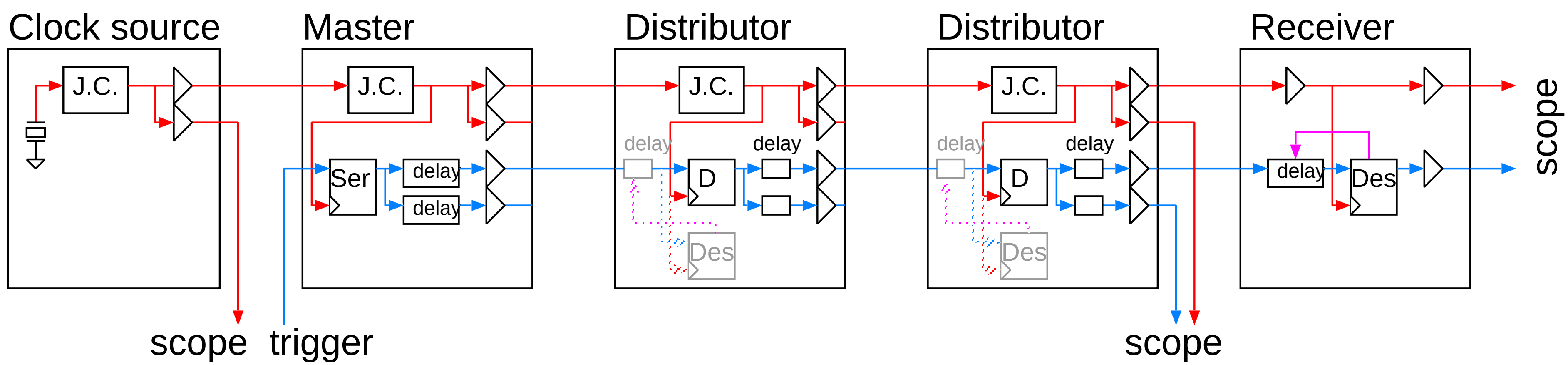
- **5 FTSW boards are regularly used for development**
- **Can be provided for front-end board development:**
  2 type-P taken by Hawaii/Nagoya, 1 type-S by IHEP, 1 type-R by NUU, 1 type-R by TUM
- **2 FTSW boards need to replace jitter cleaner**
  one can permanently damage the chip by a bad data sequence to configure the chip, warning in the datasheet
- **Jitter measurement procedure is established**
  Need to go through ∼60 boards, with help of Yonsei student
- **12 FTOP boards are tested, large jitter variation observed**
  (discuss later)**. . .**

belle2tt firmware status

(belle2tt — receiver side module to be included in frontend)

- **The current version of firmware has very limited functions**
  - **Receives CLK**
  - **Receives TRG and decode trigger timing and revolution signal** (automatically aligned to the CLK by a programmable delay)
  - **Generate a reduced clock** (up to subsystem's requirement) **and a trigger singal synchronous to the reduced clock**
  - **Status to tell if the link is established is returned as ACK signal**
  - **Reset signal is defined, but no usage is defined**
- **And following things are yet to be implemented**
  - **Trigger flow control**
  - **Schemes for fast control and status collection**
  - **Integration of the SEU recovery core**

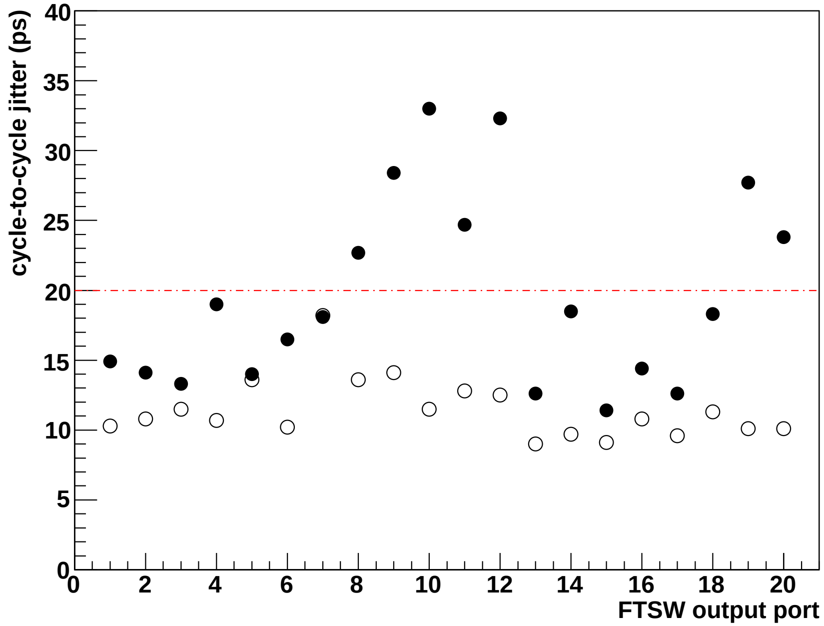
Jitter measurement



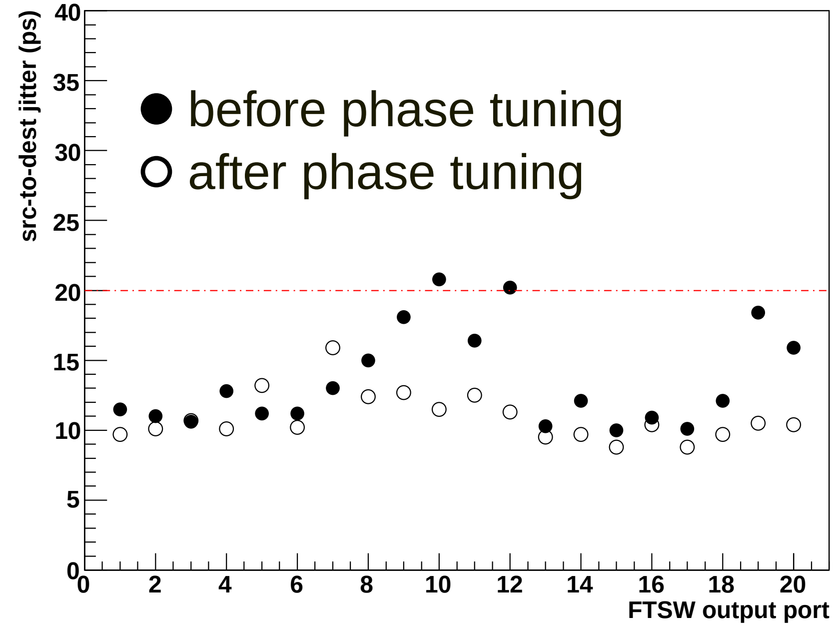
- **5 FTSW modules, 4 different firmware**
- **Jitter cleaner at each stage except for receiver**
- **All distributions with 15 CAT7 cables**
- **Clock source has very good jitter (~ 4ps)**
- **Two types of jitter measurements**
  - **cycle-to-cycle jitter**
  - **source-to-destination jitter**

Jitter result

cycle-to-cycle



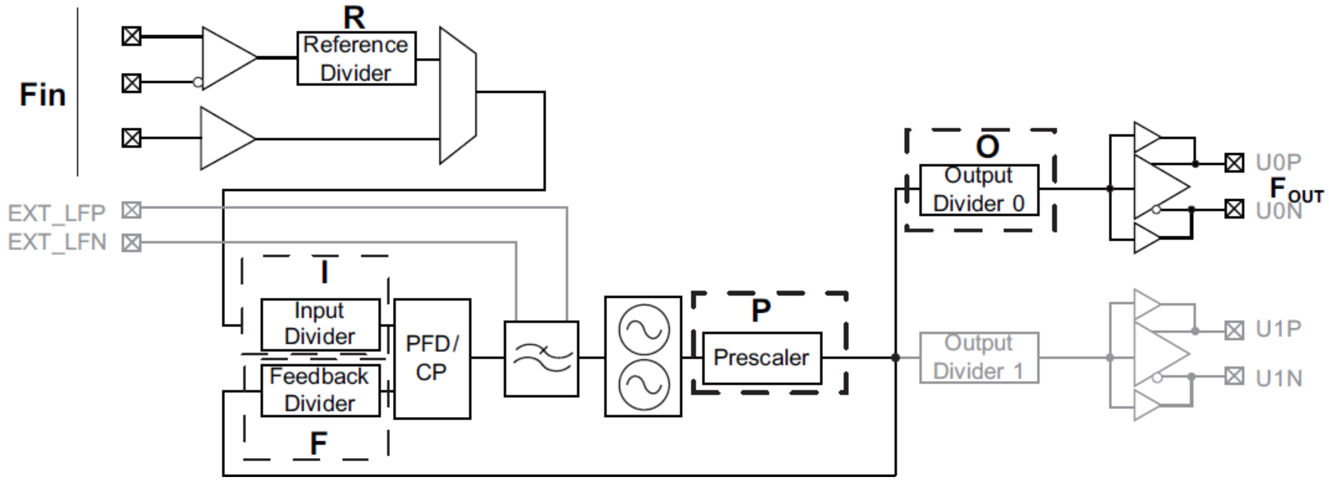
src-to-dest



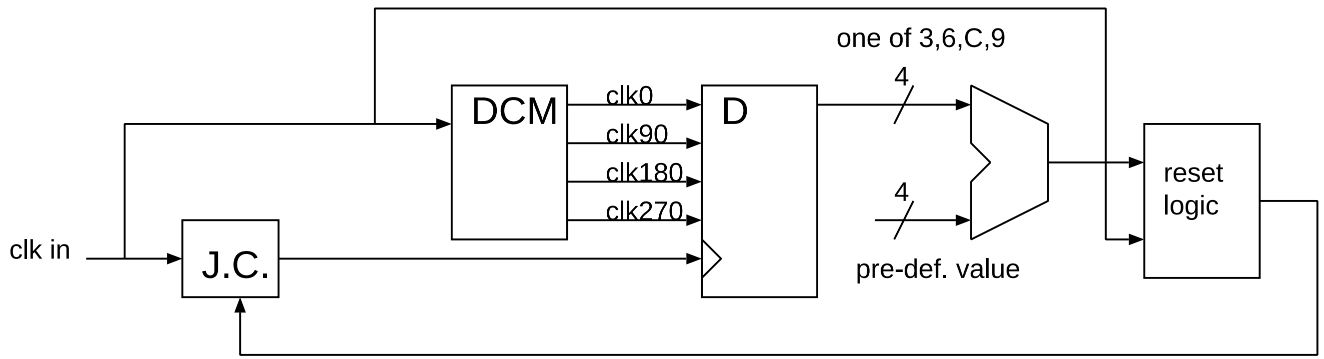
- **FTSW port dependence, due to phase between CLK & TRG**
- **Close circle before phase tuning, open circle after tuning**
- **Delay tuning brings the jitter below 15ps in most of cases**
- **Two distributors with the same delay setting, with 0.3ns diff**

(similar results with other modules, need systematic measurements)

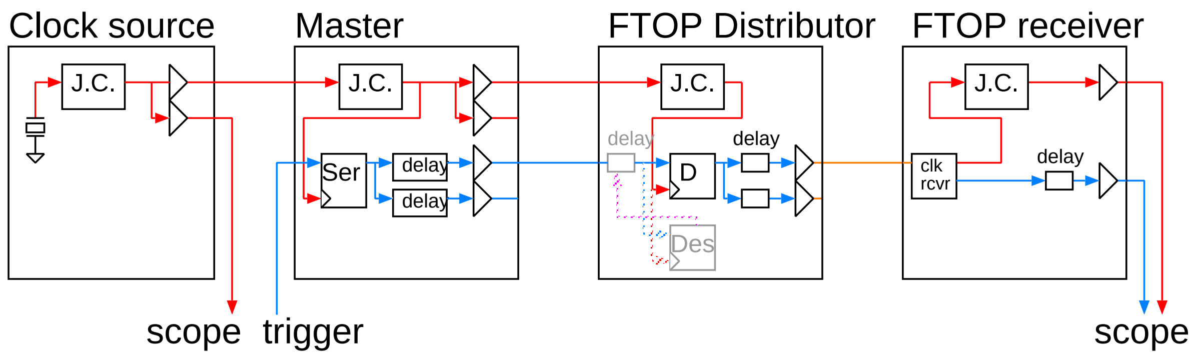
Jitter cleaner phase ambiguity

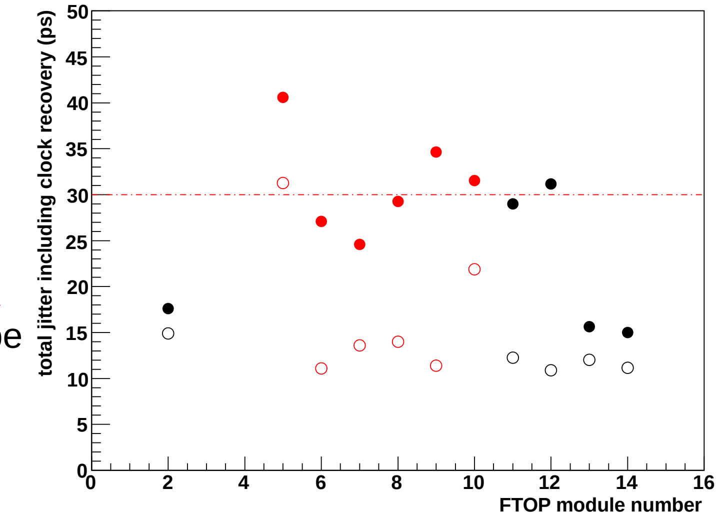


- **Because of the output divider (=4), 4-fold phase ambiguity**
- **Only by resetting the jitter cleaner, the phase is changed**
- **Phase detection code was added** (reset logic is yet to be added)



Clock recovery measurement



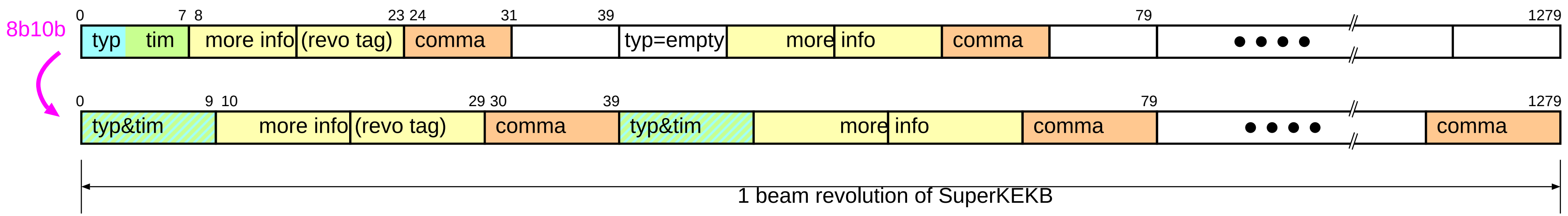


- **Similar measurement, except for the fiber connection**
- **Two Master firmware were tried** (solid and open circles)
- **Clock recovery generates 254 MHz from 254 Mbps, jitter cleaner makes it down to 127 MHz**
- **First module (#002) was OK, although I was using the chip** (SY87700AL, black) **beyond its max frequency** (208 MHz)
- **Second module (#005) was not OK, although I changed the chip to** (SY87701AL, red) **which supports higher frequncy**
- **But it's not due to the chip, still need to be understood**

Reduced clock

- **Need the best jitter for the reduced clock**
- **Simple counter output does not have a jitter as good as the driving clock**
- **Need more study — possible options:**
  - **PLL**
  - **ODDR**
  - **asynchronous logic**

FTSW data frame (prototype/old)

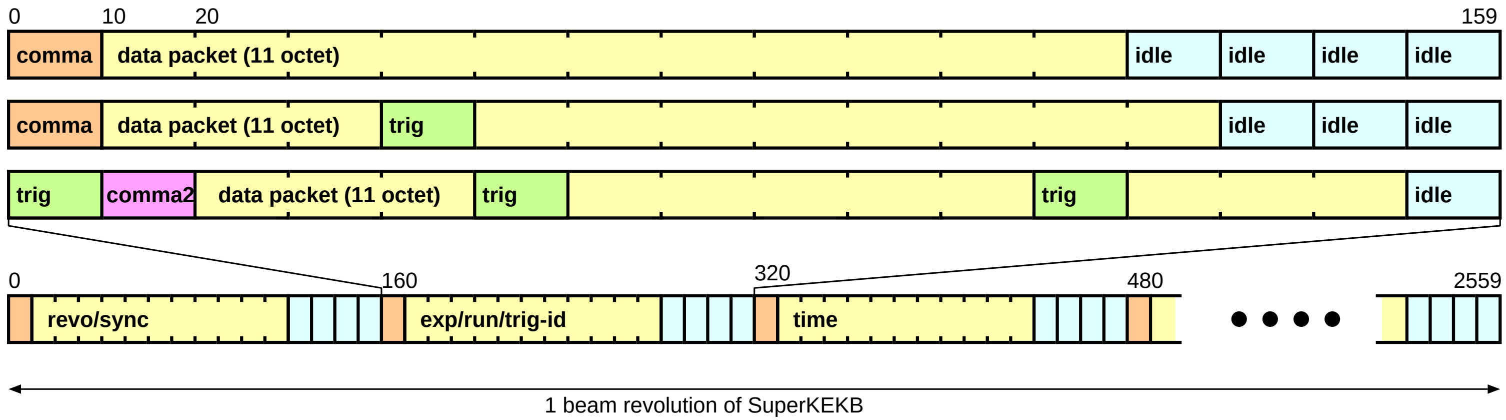


- **32-bit frame (40-bit after encoded, 80 RF buckets) with 8-bit for trigger, 8-bit for synchronization, and 16-bit for anything**
- **64 frames coincide with one beam turn (5120 RF buckets)**
- **Trigger timing is represented by 5-bit**
- **Trigger type by 3-bit**
  - **8 trigger types would be enough to define triggers from: track with or without PID timing, neutral, random, and calibration**
  - **Can be used to read/transfer no PXD data for certain trigger type**
- **Trigger latency $>$ 500 ns including $\sim$ 40 m cable+fiber at least 20 clocks (160ns) needed for 40-bit frame**

New TT protocol

- **Motivation**
  - **A bit of cleanup needed to add more features**
  - **To minimize trigger latency:**
    40-bit frame → 10-bit trigger high-priority octet
  - **To fix bug in the 8b10b code** (thanks Vladimir)

New TT protocol



- **How the trigger octet works**
  - **One frame as 16 octets, first octet is a comma**
  - **If the trigger at the first octet, second octet is a comma**
  - **Trigger octet = trig bit + 5-bit timing + 2-bit type**
    timing: 3-bit (0..4) in 5$\times$ 8ns clock (10b) + 2-bit for 2ns fine time
  - **Data octet = trig bit=0 + 7-bit data, Up to 4 triggers in one frame, 11 octet (77-bit) data in a frame**
  - **Data frame = broadcast bit + 12-bit cmd + 64-bit data**
    or broadcast bit=0 + 12-bit cmd + 20-bit adrs + 44-bit data

TTD address

- **5-bit (1...20) to define output ports**
  - **0: this**
  - **1...20: one of 20 RJ-45 ports**
  - **21...28: one of 8 optical ports**
  - **31: all ports (broadcast)**
- **$5 \times n$ bits to define the node in (downlink)^n configuration**
- **20 bits to define the address of the FTSW or FEE board for the nominal configuration** ([dist1][dist2][dist3][dist4])**, but the master can remove the first 5-bits in the data packet**
- **All the FTSW and FEE boards have to learn its own address during initialization. In this case, the Master FTSW has to talk to each FTSW and FEE one by one.**
- **Once the initialization is made, all the downlink data is simply broadcasted, and the FTSW/FEE will take the signal only for their addresses**

Data frame

- **1 data frame per 630 ns, 16 frames per one beam revolution**
- **Trigger-id is NOT attached to every trigger timing**
- **List of requests**
  - **initialization mode** for address assignment
  - **port disable and port mask** for partitioning
  - **time** unix time in 32-bit
  - **revolution signal** with revolution count
  - **event counter reset / buffer clear signal** with run number
  - **error counter reset** for recovered SEU, etc
  - **exp/run/trigger number** for self-consistency check

Address assignment procedure

- **initialization mode request** is broadcasted, then further requests are no more broadcasted
- **for each output port of master**
  - **port mask request** to select only one port at master
  - **address assignment request** to set the address of downlink
  - **for each output port of downlink**
    - **port mask request** to select only one port at downlink
    - **address assignment request** to set the address of $(\text{downlink})^2$
    - **for each output port of** $(\textbf{downlink})^2$ **. . .**
- **Broadcast mode request** is broadcasted
- **For 1000 nodes, $\sim$2000 requests needed, taking $\sim$1.3 ms**
- **Tree-structure is kept in a database**
  — just a list of 20-bit addresses (and their names)

New firmware status

- **Logic first designed in C and then translated into VHDL**
  - **8b10b code fixed, tested in C, and implemented into VHDL**
  - **Trigger octet and data frame encoder programmed in C**
  - **Data frame decoder is now being programmed, including packet alignment using comma characters**
  - **Addressing scheme is yet to be programmed**
- **Once these are done, ACK signal will be designed**
  - **Similar 16 octet data packet, but flow control should be faster**
  - **Similar high-priority flow-control packet, based on K characters**
- **Target: end of FY, may not be ready for B2GM**

ACK signal

This is still more like random thoughts than implementation

- **ACK is also serial data of 40-bit frames with 8b10b**
- **Has to be merged at every distributor, takes almost 1 μs to get back to the master**
- **Master issues up to 5 triggers with $\geq$190 ns interval before receiving the response** (or more than 5, according to the recent discussion about the SVD readout scheme)
- **FEE issues "response" at every trigger when it has a buffer to receive next n triggers after x μs** (n and x to be defined)
- **Or, FEE asserts "busy" at any time when it no more has a buffer to receive next $n + 1$ triggers** (same thing? This status may come earlier)
- **For PXD-DHH, sequencial triggers with short intervals are more like a bit longer single trigger**

ACK signal latency

- Cable delay $\sim$200 ns
- Deserializing + serializing 10-bit $\sim$100 ns $\times$ 4
- Use K-characters directly without 8b10b decoding to reduce latency
- Shorter trigger latency also helps
- Not implemented yet, to be tested next

Collecting status

- **Trigger flow control as K characters** (top priority)
- **Frame synchronization with comma charactors** (overridden by trigger flow characters)
- **Up-going error message packet** (20-bit address + 4-bit code)
  - **Buffer clear request**
  - **FPGA download request**
  - **Unrecoverable SEU/MBU detected**
  - **Trigger number inconsistent**
  - **Clock PLL unlock occured**
- **Status message packet** (20-bit address + 4-bit code + 96-bit info)

Slow controling Fast control

- **Read**
  - **All information collected every O(ms) from >1000 nodes at master FTSW, fast enough for run/slow control**
- **Write**
  - **Address setting takes time as driven by a CPU but still O(s)**
  - **Broadcasting a reset is a single (or a few) register shot**
  - **and partitioned broadcast (per detector) is a few register shots**
- **All FTSW based subsystems viewed in a structured way**
  - **Good for GUI, probably independent of control framework**
- **JTAG**
  - **JTAG sequence can be embedded into the TRG request** (no trigger nor initialization during JTAG)
  - **Single JTAG tick per request corresponds to 1.6 MHz** (a bit slow)
  - **Two JTAG ticks would be better** (close to Xilinx default)

More on JTAG-link

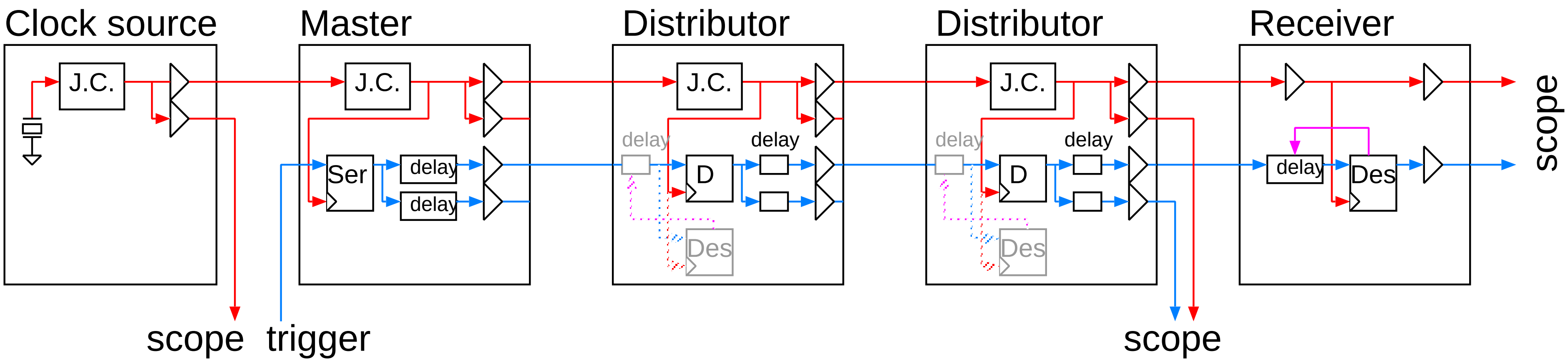
- **Most of the FEE boards now equips the RJ-45 port for JTAG**
- **Programming via impact program, Xilinx USB cable and JTAG-link is established and regularly used**
  (CAT cable is easier to plug in than Xilinx flat cable)
- **Programming multiple FPGA in parallel has been tested**
- **Programming and verifying without Xilinx cable with XSVFplayer program is established by Igarashi-san**
- **A few JTAG functions have been established by a standalone C program even without the XSVFplayer**
- **Plan is to pack all these findings into a standalone C code to program and verify directly from *.bit and *.msk file**

Summary

- **Basic timing performance of FTSW-based TTD system is established, but not completed yet**
- **Many modules have been produced for a massive test**
- **A prototype firmware is available, with limited functions**
- **A more realistic** (but not necessarily final) **version of the firmware is under development**
- **To start implementing and testing these signals**
(as basic worries on performance have now been cleared)

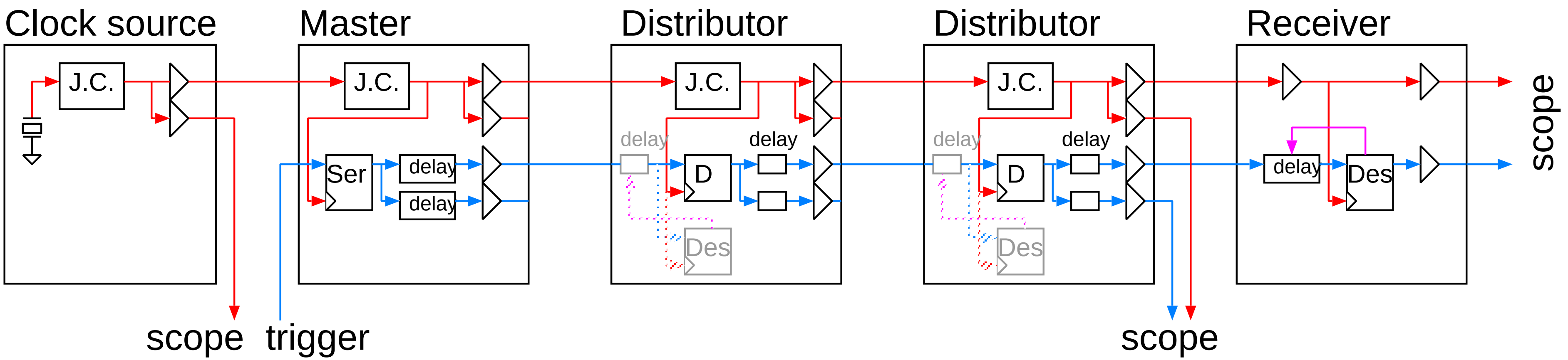
Backup (B2GM slides)

FTSW firmware



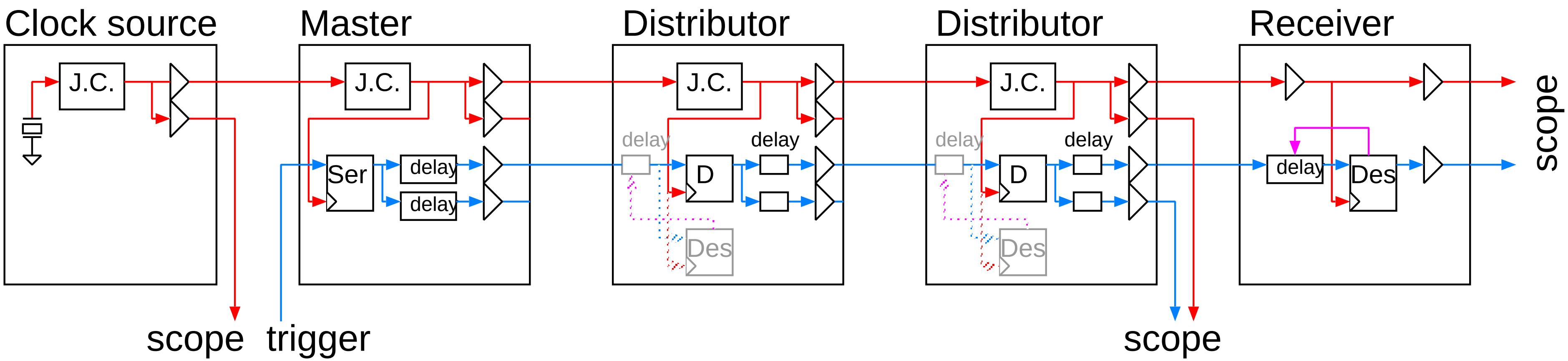
- **At least 2 types are needed, can’t be unified**
  - **master module, trigger encoder**
  - **distributor, no encoding/decoding**
- **And four more types for fibers (may be merged later)**
  - **(sender / receiver) ⊗ (single / double pair link)**
- **Additional firmware for test setup**
  - **clock source (to emulated KEKB RF clock)**
  - **receiver, trigger decoder (to emulate frontend board)**

Clock source



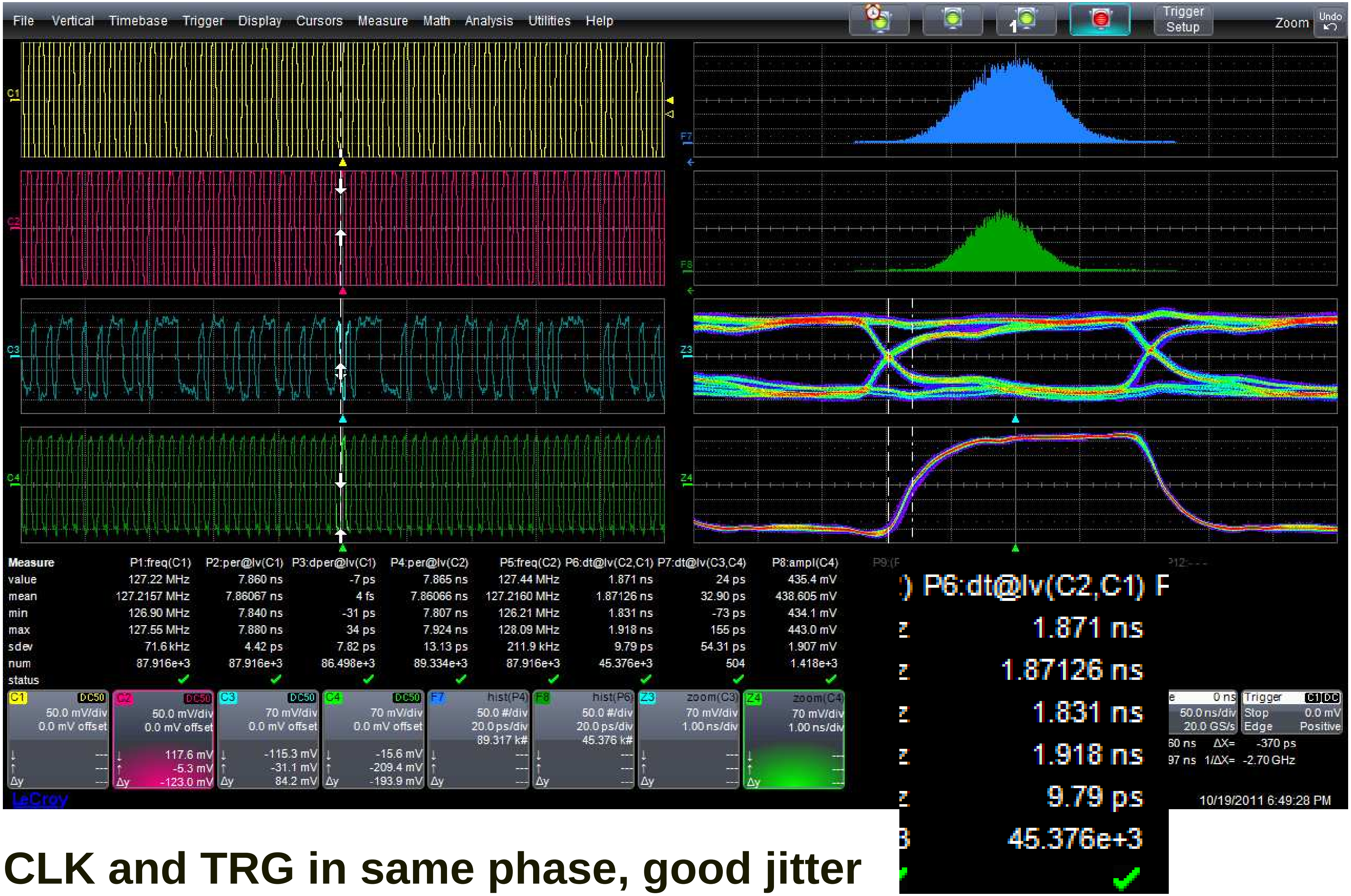
- **Clock output only**
- **4 ps jitter**

Master



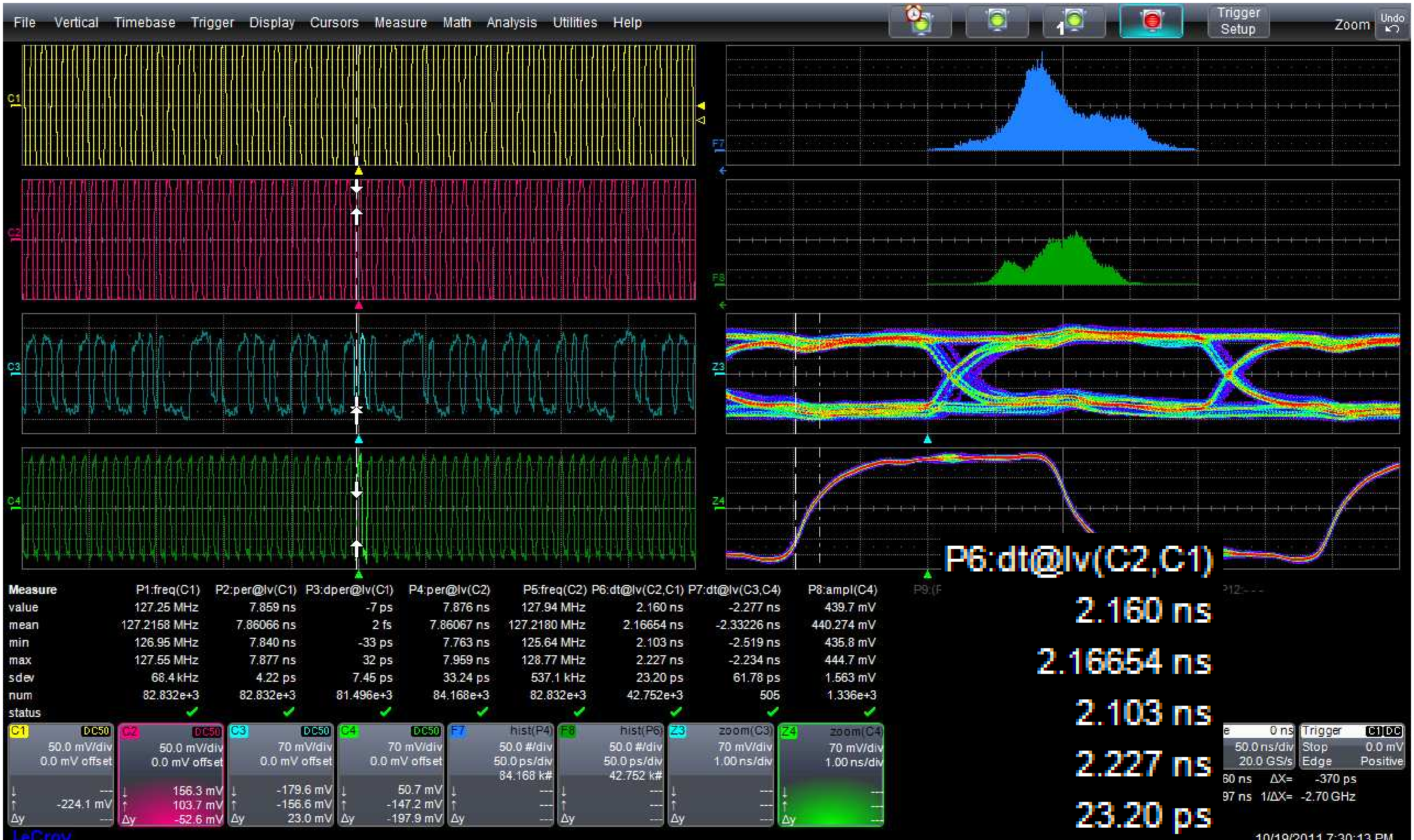
- **Jitter cleaning before using the clock**
- **Trigger signal is encoded only here**
- **Hand-adjustable delay for each channel to reduce the skew**
- **CLK jitter is strongly correlated with the phase difference between CLK and TRG . . . smallest jitter TRG has to have edges around 0.0 to 0.8 ns after CLK, and already 1.8 ns difference (90 degree) makes the jitter very bad**

Good case



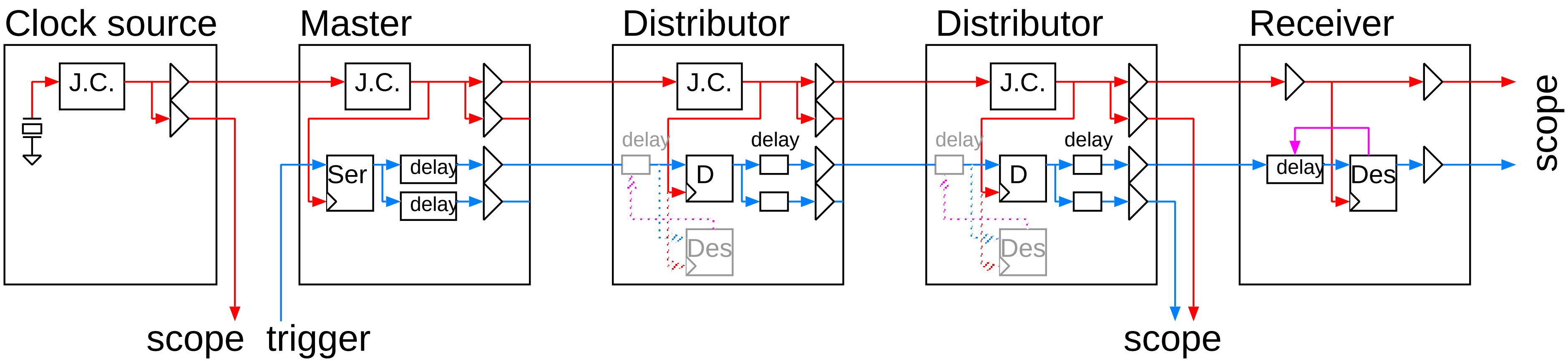
CLK and TRG in same phase, good jitter

Bad case



CLK and TRG out of phase, bad jitter

Receiver



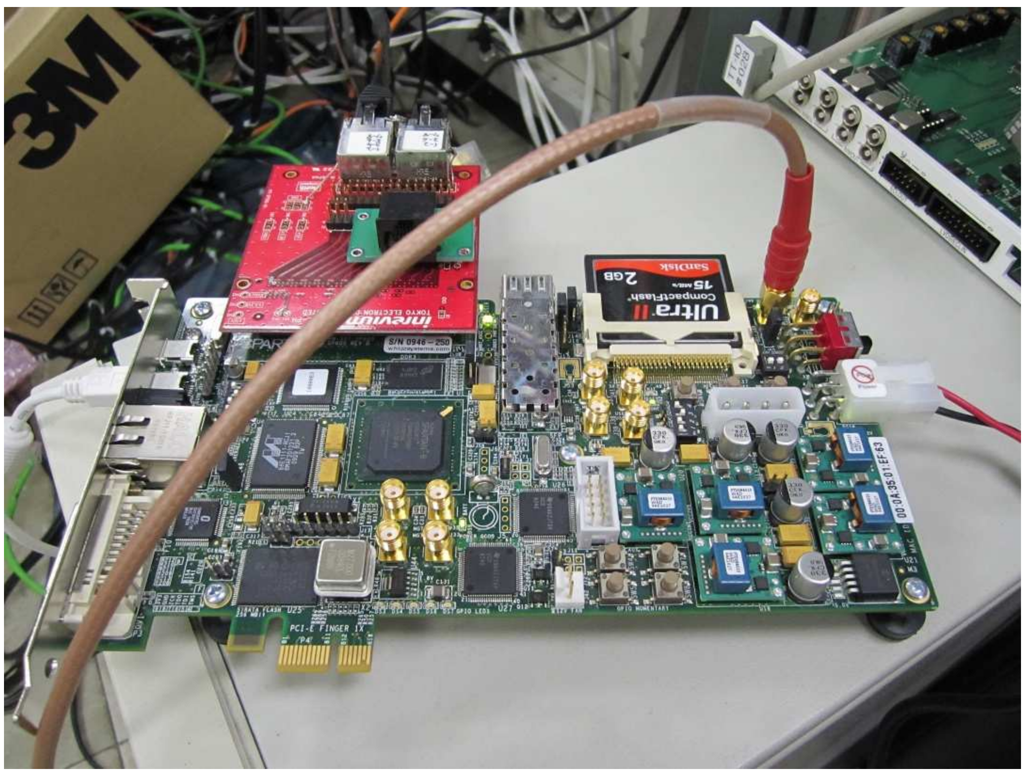
- **Trigger signal is decoded only here**
- **Auto delay adjustment by checking the comma charactor in the decoded trigger serial data**
- **Jitter cleaning should not be done to emulate the frontend**
- **15m cable has to be used for input**
- **During the jitter measurement, TRG signal should not be on the cable to the scope**
- **O(10ps) jitter still seems to be acheivable (prev.pages)**

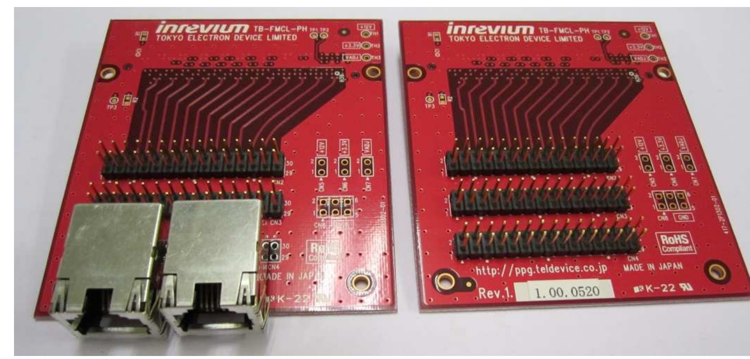
15m cable directly to scope

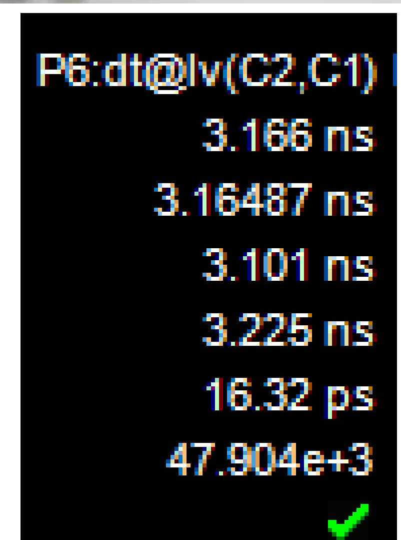
After 15m, TRG eye gets smaller, but jitter is still good

Spartan6 Receiver

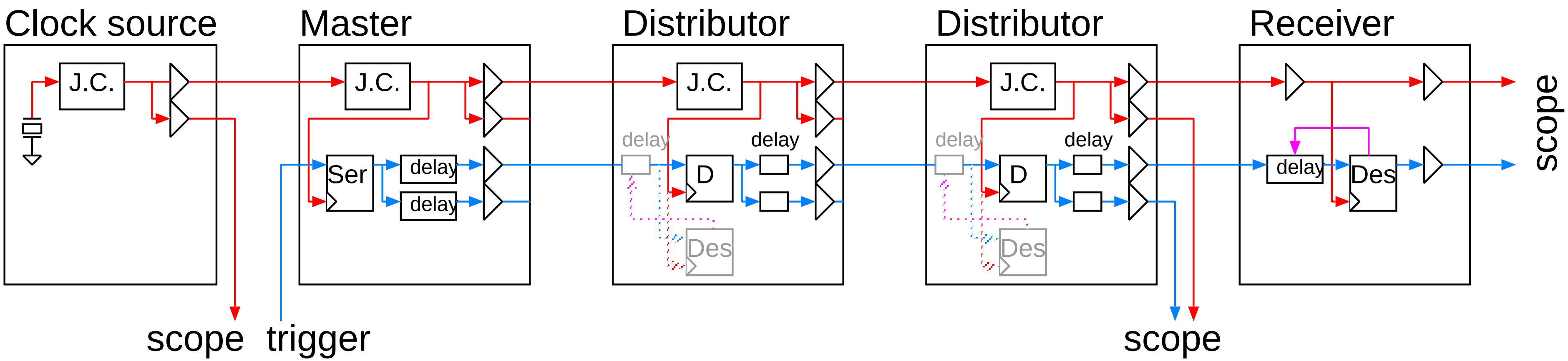
- Universal pin header FMC card, modified to attach 2x RJ-45
- TRG/CLK delivered to the Spartan6 evaluation board (SP605)
- CLK output from SP605 to measure jitter







Distributor



- **Ch-by-ch skew is as large as 1.8ns, module dependence $\sim$0.3ns**
  - **If nothing is done, some of the channels give bad jitters**
  - **Delay calibration is needed**
- **Delay output needed only for output? (current design)**
  - **CLK/TRG phase is constant for any length of cable**
- **No data-driven auto delay adjustment for now**

Backup2 (PXD-DAQ WS slides)

DHH interface

- Two timing distribution scheme co-exist for now
  DHH to receive clock from FTSW or DHHC
- I prefer the FTSW scheme as it provides uniform fast control system (and slow visualization) throughout Belle II
- For FTSW scheme
  - 2 FTSW to serve 20 DHH at each of forward and backward
  - or 3 FTSW to serve 20 DHH for both timing and JTAG
- No problem to provide 5 or 7 FTSW boards for tests in Germany