STATUS OF THE SRS INTEGRATION INTO THE TOTEM DAQ SYSTEM

ON BEHALF OF TOTEM DATA ACQUISITION GROUP

F.S.CAFAGNA[1], A.FIERGOLSKI[1,3], M.QUINTO[1,2] E.RADICIONI[1]

(INFN-BARI[1], UNIVERSITY OF BARI[2], WARSAW UNIVERSITY OF TECHNOLOGY[3])

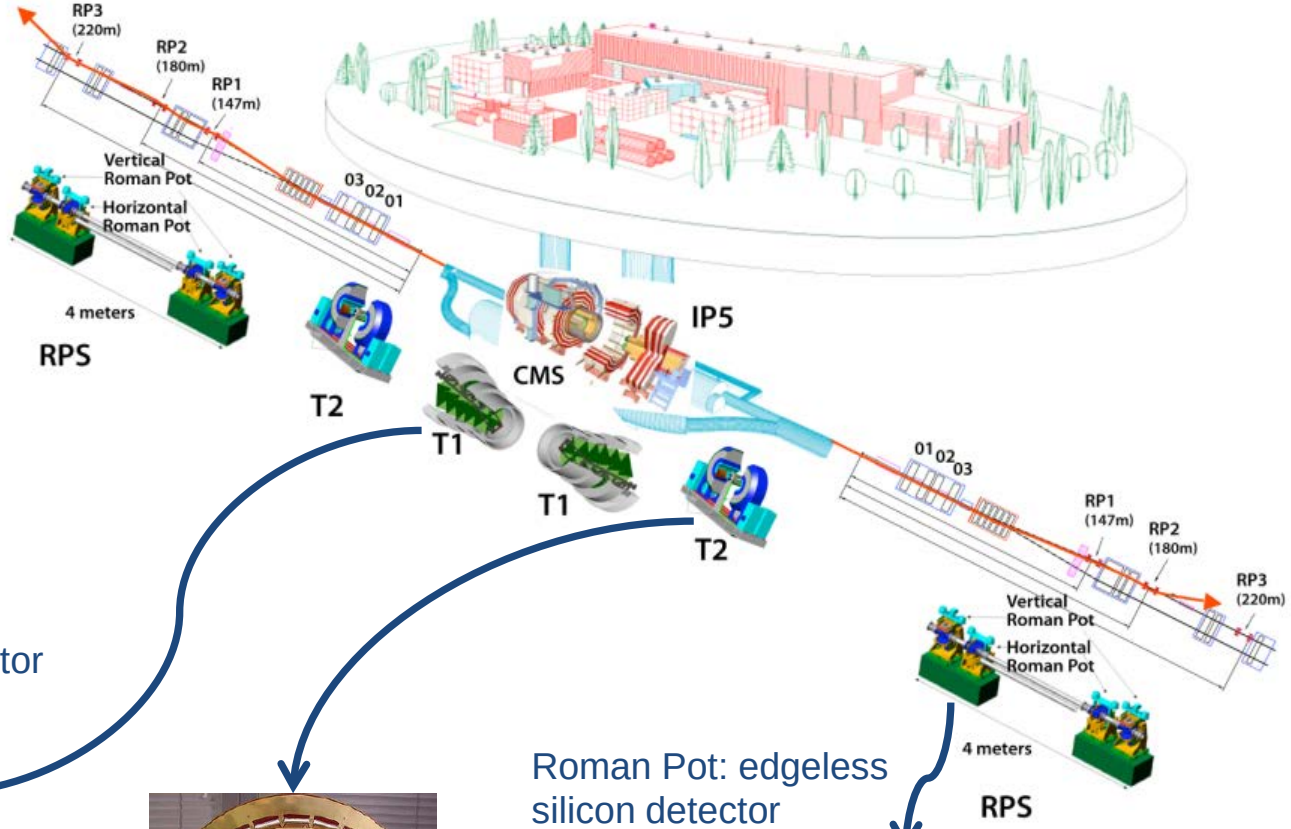
RD51 Mini week – CERN, June, 16-19 2014

Outline

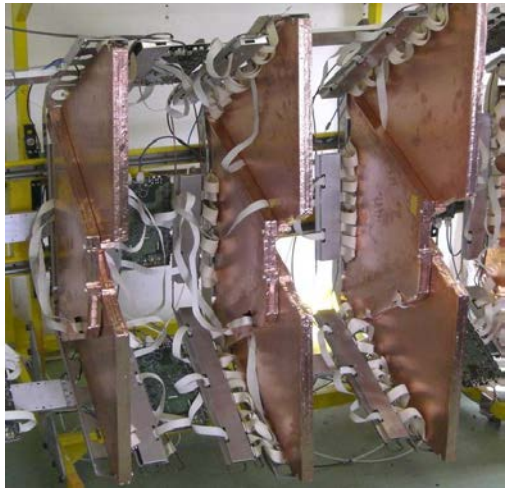
- Consolidation of the TOTEM DAQ System with SRS;
- Status of firmware development;
- SRS readout stress test results;
- Hardware procurement;
- Firmware design & development strategies;
- Outlook

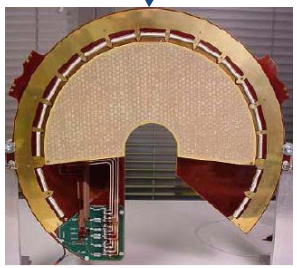
TOTEM Experiment at LHC

- TOTEM measures total p-p cross section at the LHC energies and studies diffractive processes.
- TOTEM adopts three detectors symmetrically placed at the Interaction Point 5 (T1, T2, Roman Pot).
- All TOTEM detector adopts common readout and trigger electronics. The VFAT chip provides readout and trigger capabilities.



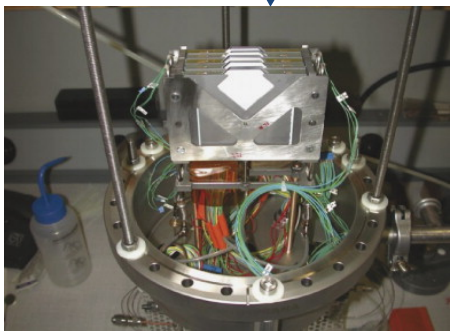
T1: Cathode Strip Chamber detector



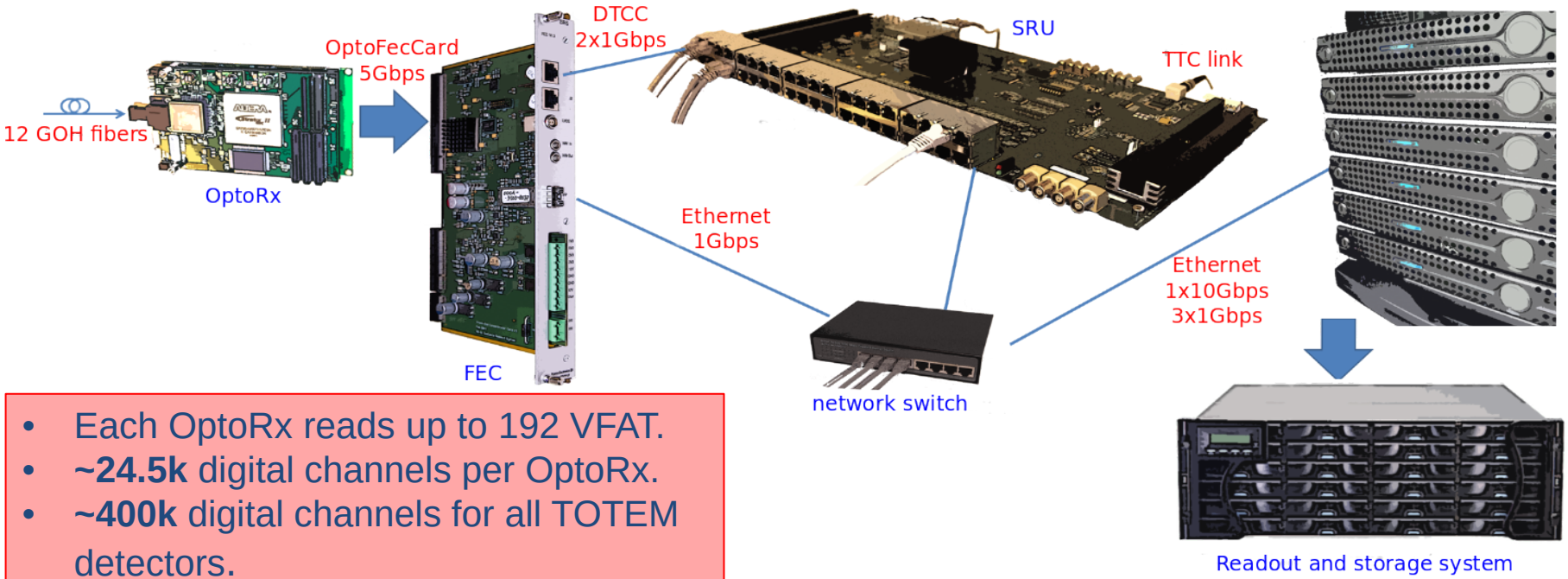


T2: triple GEM detector

Roman Pot: edgeless silicon detector



Consolidation of TOTEM DAQ with SRS System



- Each OptoRx reads up to 192 VFAT.
- **~24.5k** digital channels per OptoRx.
- **~400k** digital channels for all TOTEM detectors.

• SRS provides a **cost effective** alternative to the present VME based solution.
• The Ethernet based readout offers many off-the-shelf hardware solutions (NICs, switches) that naturally improve their performances driven by the market.
• SRS offers many possibilities of implementing **data reduction and filtration** by leveraging the SRS resources at different stages in the DAQ chain.
• SRS flexibility allow system extension for future TOTEM upgrades.

SRS Firmware development and test for TOTEM

- Implement a robust Ethernet data transmission using UDP:
  - Complete review of the UDP stack;
  - Development of readout software for both DATE and stand-alone application using C++ and Boost libs.
- Extensive stress test in different configurations:
  - 1 FEC to 1PC;
  - 4 FEC to 1PC via switch;
  - 2x2 FEC to 2 PC via switch.
- Implement the LHC TTC full set of command and timing signals through DTCC Links SRU -> FECs -> OptoRx;
- Full TTCrx test of timing and fast commands propagation to the font-end.

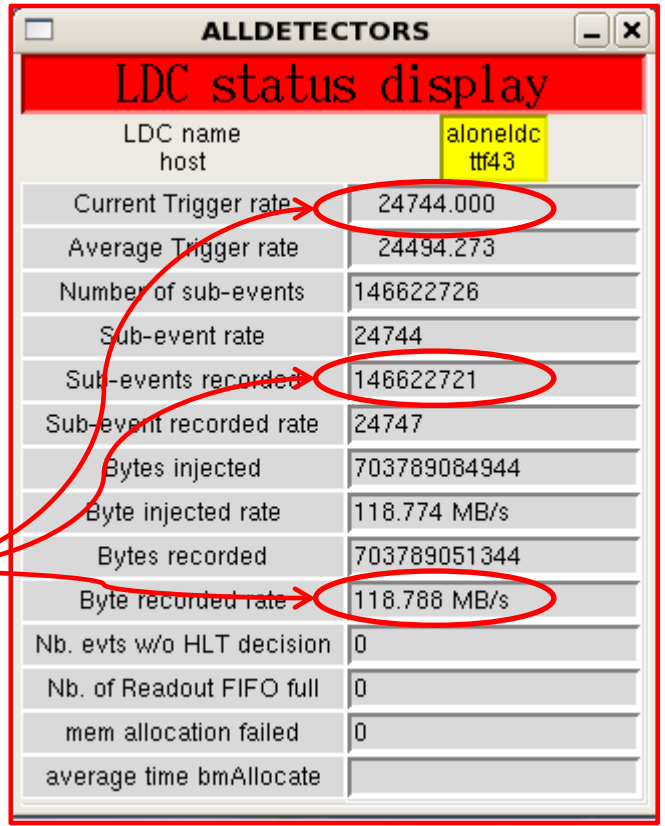
Implementation of a new UDP Stack block

- UDP Stack is freely distributed by opencore.org in which we are collaborating for testing and developing.
- AXI4-Stream standard bus;
- Support for 255 entries ARP table (essential for multi-host connection);
- IP TX and UDP RX, TX interfaces;
- **Lower latency**. One FIFO memory level in the TEMAC, **no memory to memory copy**;
- **Lower memory and logic resource usage**;
- Good performance observed during readout test

	Slices	Slice Reg	LUTs	LUTRAM	BRAM/FIFO
UDP Stack and TEMAC	843	1375	1343	20	9

50% less LUTs

- Trigger rate at flat top ~25kHz
- Readout bandwidth close to the link limit 118MB/s
- System stability over more than 140M events



UDP based readout pitfalls

- Ethernet protocols such as TCP/UDP are lossy by design. Packet loss can occur:
  - TCP-IP solves the issue of packet loss implementing retransmission -> very costly to implement on FPGA a TCP/IP offload engine.
  - Dropped packet are simply lost using UDP.

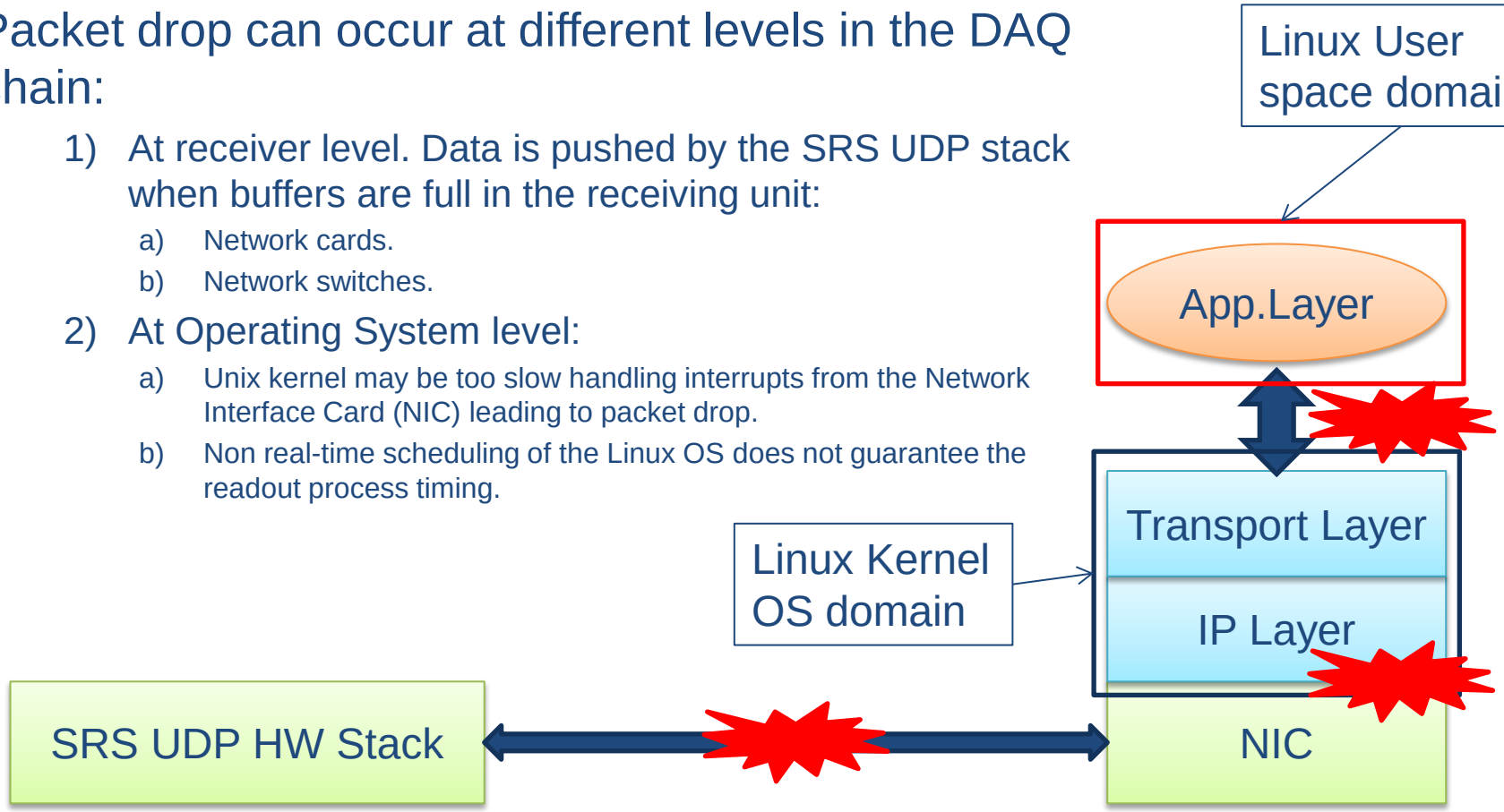
- UDP based readout CONS:
  - ✓ Packet loss rates are not acceptable for long-running experiments with event building such as TOTEM. Event synchronization must be guaranteed.
- UDP based readout PROS:
  - ✓ UDP stack is as lightweight as simple to implement on FPGA devices.

Is it possible to have a reliable UDP readout?

UDP based readout pitfalls

Packet drop can occur at different levels in the DAQ chain:

1) At receiver level. Data is pushed by the SRS UDP stack when buffers are full in the receiving unit:
   a) Network cards.
   b) Network switches.
2) At Operating System level:
   a) Unix kernel may be too slow handling interrupts from the Network Interface Card (NIC) leading to packet drop.
   b) Non real-time scheduling of the Linux OS does not guarantee the readout process timing.



Is it possible to have a reliable UDP readout?

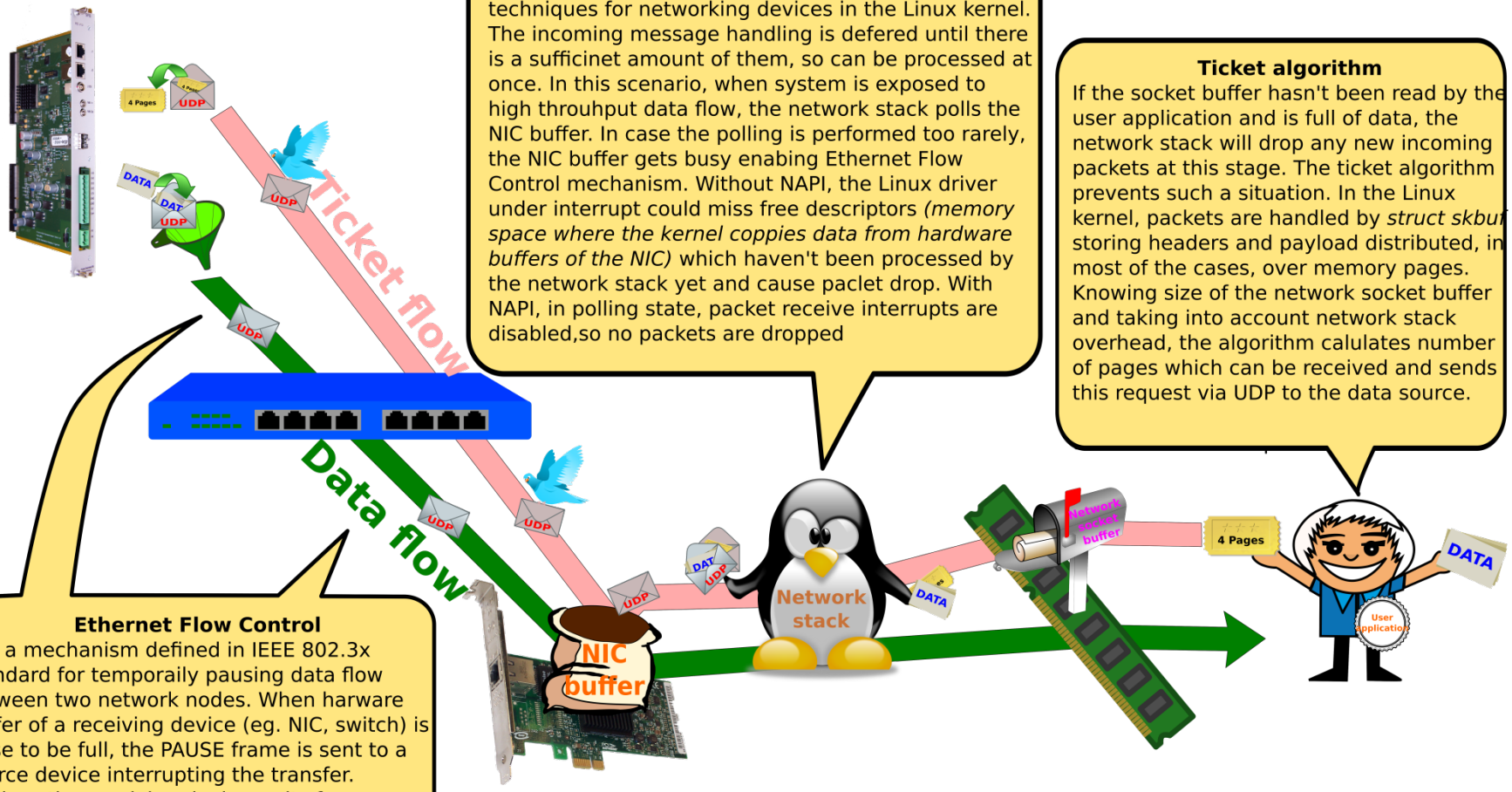
Answer is: YES! ... And here is the recipe:

New API

It is an interface to use interrupt mitigation techniques for networking devices in the Linux kernel. The incoming message handling is defered until there is a sufficinet amount of them, so can be processed at once. In this scenario, when system is exposed to high throuhput data flow, the network stack polls the NIC buffer. In case the polling is performed too rarely, the NIC buffer gets busy enabing Ethernet Flow Control mechanism. Without NAPI, the Linux driver under interrupt could miss free descriptors *(memory space where the kernel coppies data from hardware buffers of the NIC)* which haven't been processed by the network stack yet and cause paclet drop. With NAPI, in polling state, packet receive interrupts are disabled,so no packets are dropped

Ticket algorithm

If the socket buffer hasn't been read by the user application and is full of data, the network stack will drop any new incoming packets at this stage. The ticket algorithm prevents such a situation. In the Linux kernel, packets are handled by *struct skbuf* storing headers and payload distributed, in most of the cases, over memory pages. Knowing size of the network socket buffer and taking into account network stack overhead, the algorithm calulates number of pages which can be received and sends this request via UDP to the data source.



Ethernet Flow Control

It is a mechanism defined in IEEE 802.3x standard for temporaily pausing data flow between two network nodes. When harware buffer of a receiving device (eg. NIC, switch) is close to be full, the PAUSE frame is sent to a source device interrupting the transfer. Further, the receiving device waits for space to be released from the buffer and afterwards sends RESUME frame resuming transmission.

Presented at Real-time conference RT2014, Japan

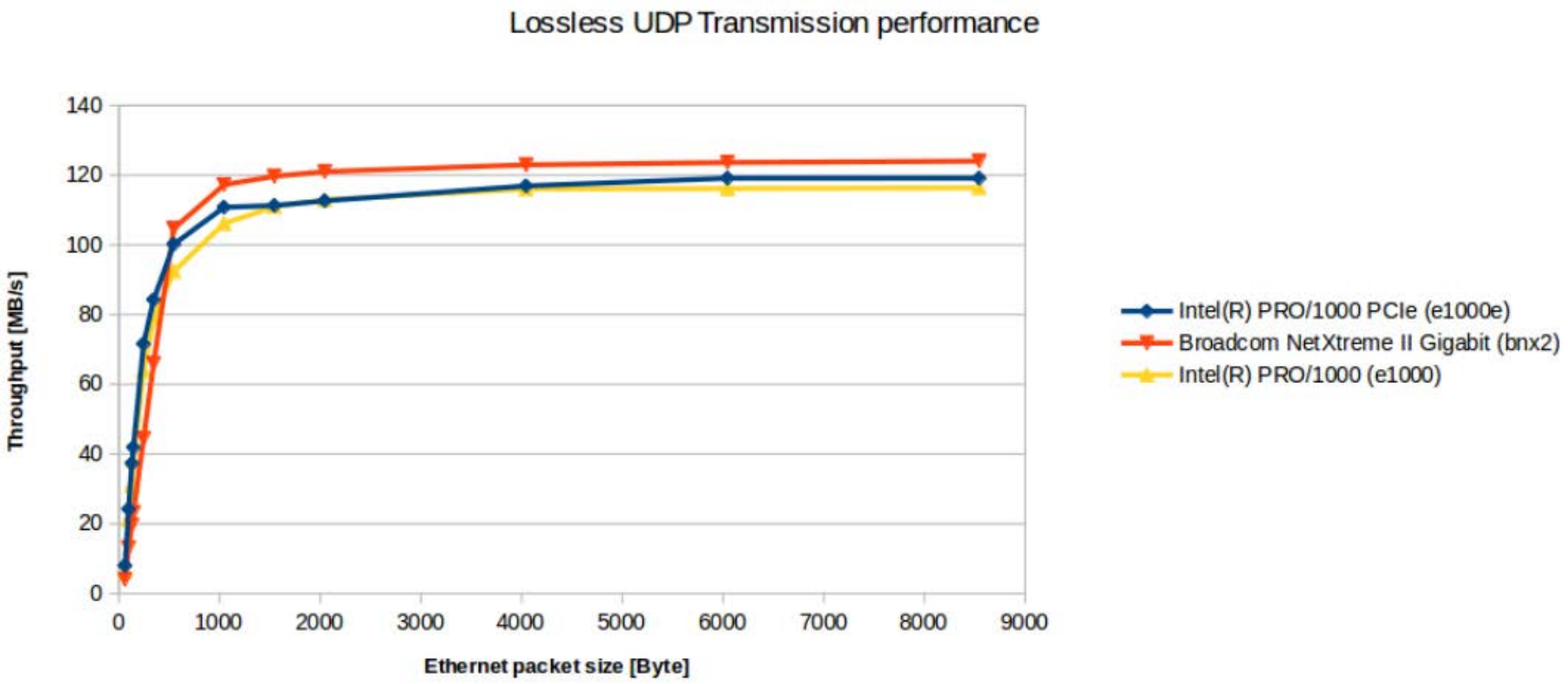
UDP Readout stress test results

First test scenario:

- 1 FEC streaming data to 1 PC.
- Processor Intel i3.
- Linux Kernel 2.6 Intel.
- Several NICs tested: PCI NIC e1000, PCIe NIC e1000e, Broadcom NetXtreme.
- Writing to a standard HD drive ~65MByte/s.
- Generating others high demand IO processes active in parallel on the same CPU and filesystem.

Packet Error Rate (PER) < $9*10^{-11}$ BERR < $2.5*10^{-15}$

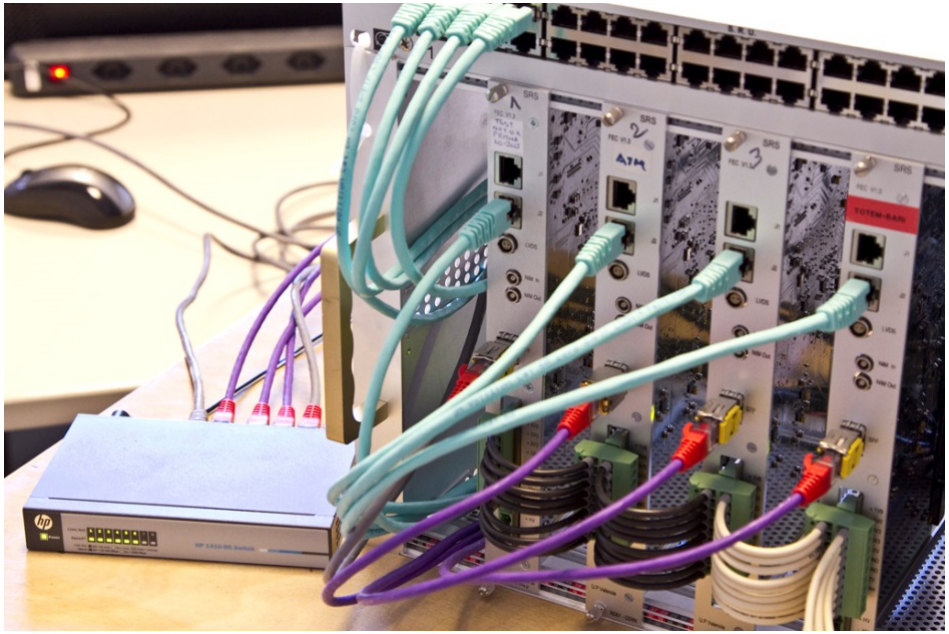
UDP Readout stress test results



UDP Readout stress test results

Second test scenario:

- 4 FEC streaming data @1Gbit/s to 1 PC receiving @1Gbit/s
- Processor Intel Xenon
- Linux Kernel 2.6 Intel
- Broadcom NetXtreme
- **Commercial switch** form HP and data server managed 3Com 4200G switch

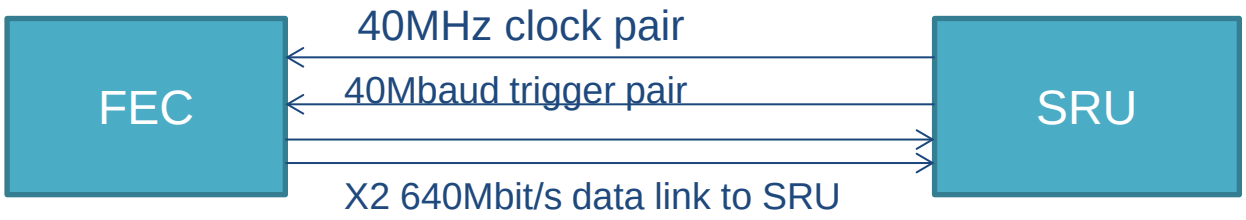


Packet Error Rate (PER) < $9.9*10^{-10}$

- Single packet is drop observe once in a while.
- Performance worst than point-to-point connection due to the switch, but it is still acceptable.
- Simply relying on the Ethernet Flow Control + the ticketing mechanism the system rate adjust to 1GBit and cope with system fluctuations.
- **No trigger rate tuning needed!!!**

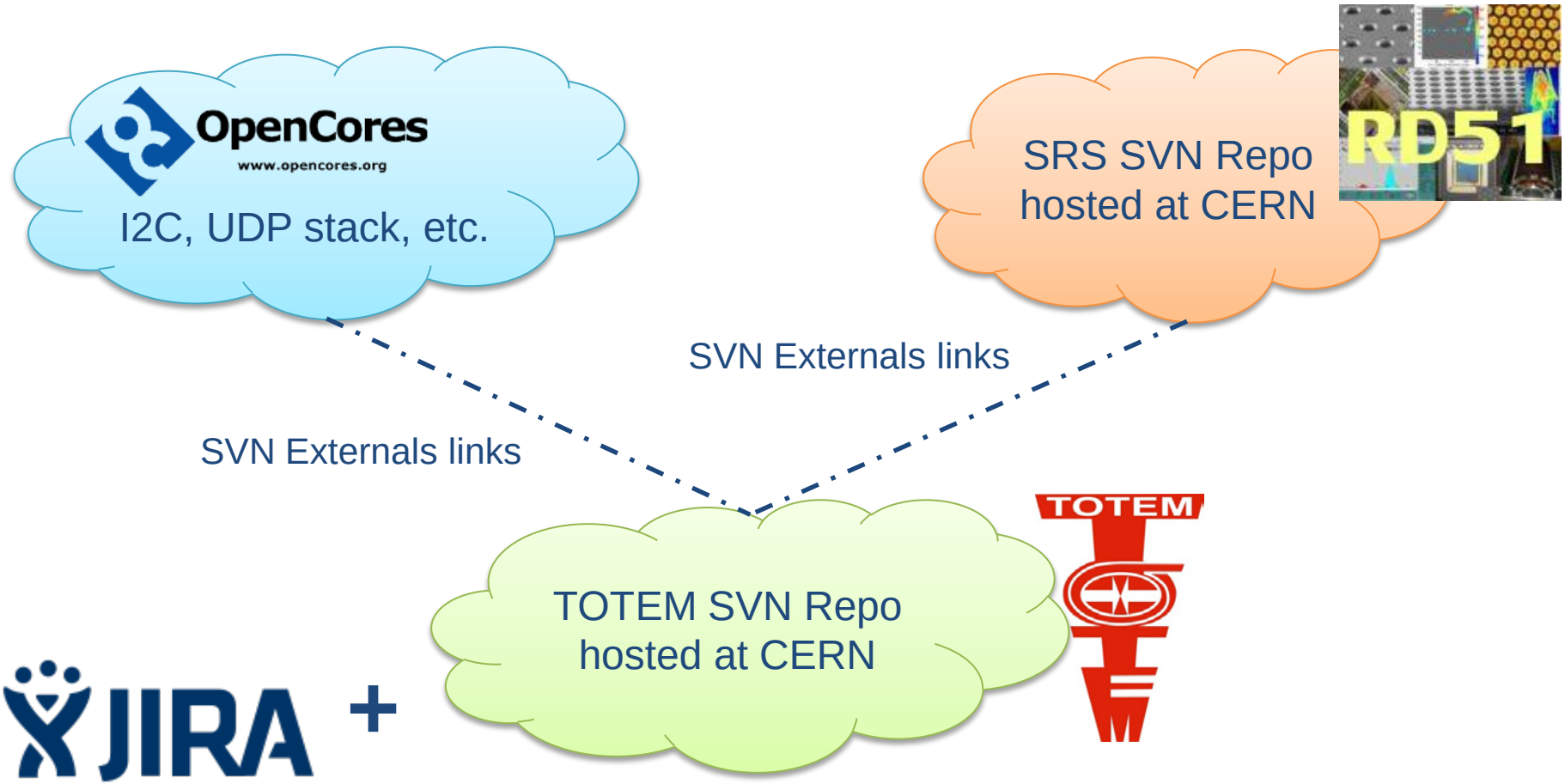
DTCC Link Implementation for TOTEM

- The DTCC Link trigger frame for TOTEM is 8 bit long + 8 bit control word.
- The Link Baud rate is 40Mbaud, compliant with TTCci system requirements.

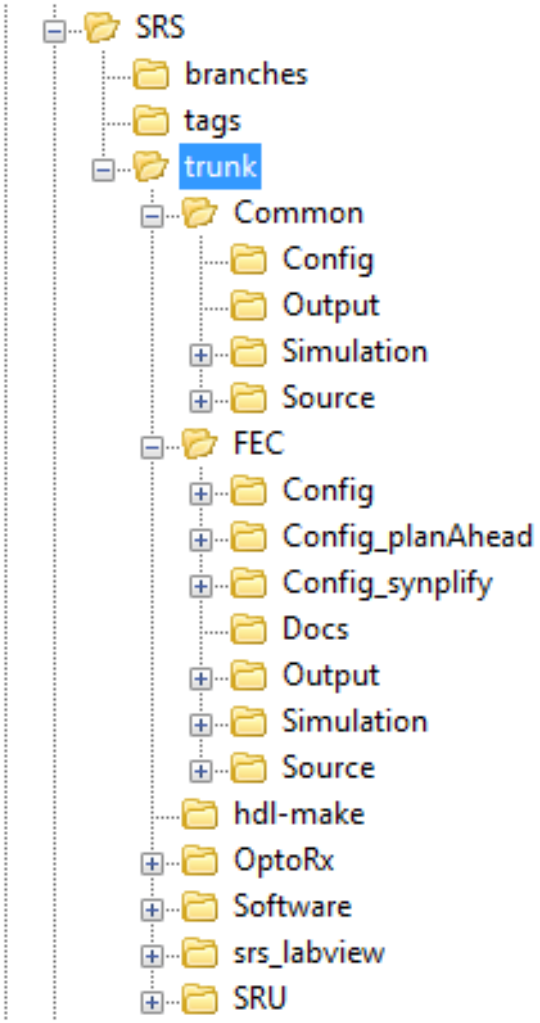


- TTCrx Commands are broadcasted to all DDTC Links:
  - Level 1 Accept;
  - BC0, EC0, OC0;
  - Resync;
  - Back-pressure
- This DTCC implementation is TOTEM specific; however it uses the DTCC firmware core blocks developed and offered to us by A.Tarazona.
- It can be integrated in the main release.
- Unfortunately the main SRS SVN Repo has, so far, not up-to-date DTCCL code.

Firmware design and development strategies

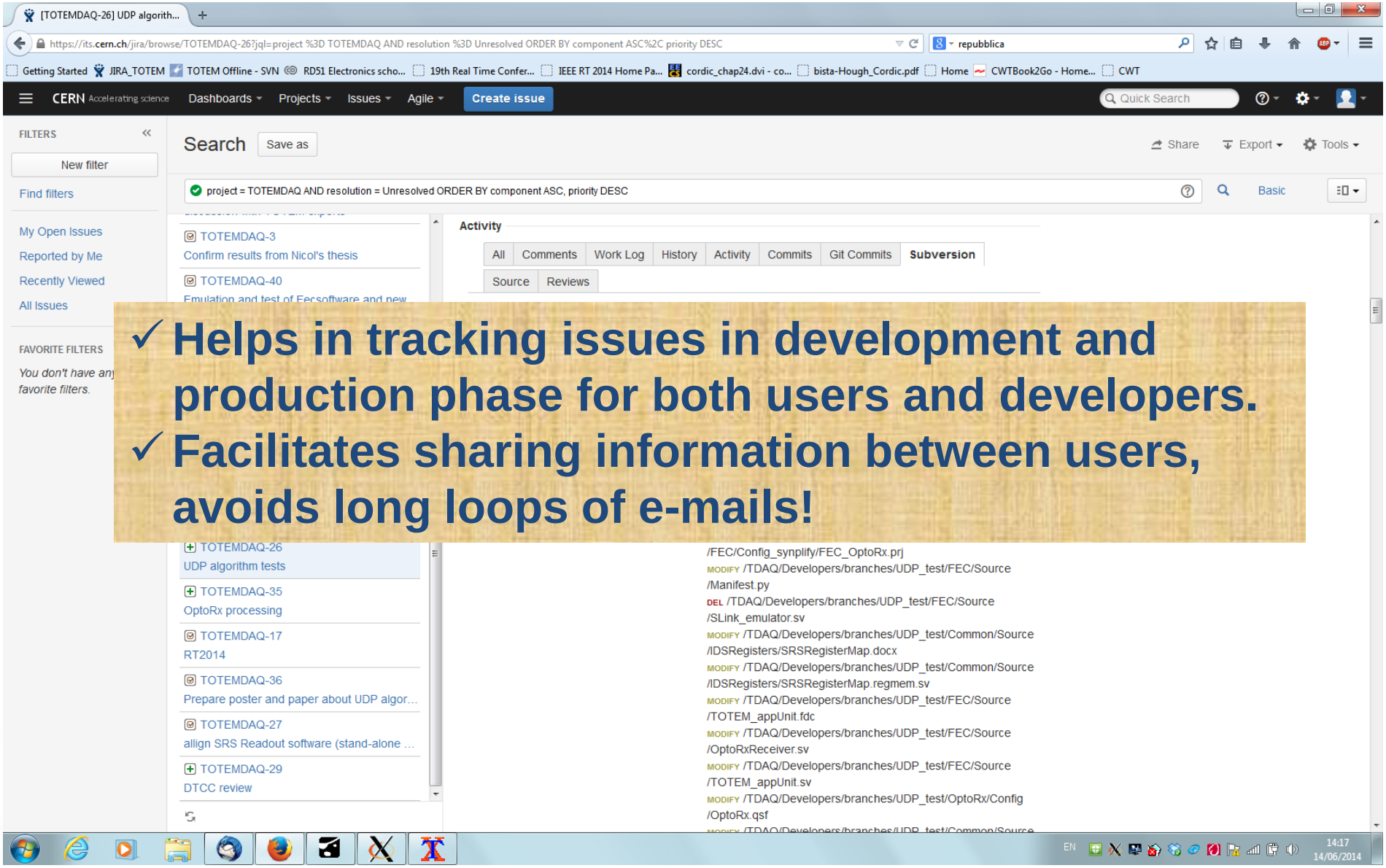


TOTEM SVN Structure and release policy



- All subsystem projects (SRU, FEC, OptoRx) sit in the same repository trunk.
- Software package (Slow Control, readout, drivers, APIs) is common to all subsystems.
- No SW binaries files on Repo.
- FPGA programming binary files go to a dedicated directory: Output.
- Tags include the full project: all subsystems + software -> This implies compatibility and interoperability of software and firmware versions.
- The TOTEM SVN Librarian ask developers to pass through a test procedure before making a tag.
- Commit to tagged version in /tag is not allowed of course!

JIRA and SVN


[TOTEMDAQ-26] UDP algorith...
https://its.cern.ch/jira/browse/TOTEMDAQ-26?jql=project %3D TOTEMDAQ AND resolution %3D Unresolved ORDER BY component ASC%2C priority DESC
Getting Started
JIRA_TOTEM
TOTEM Offline - SVN
RD51 Electronics scho...
19th Real Time Confer...
IEEE RT 2014 Home Pa...
cordic_chap24.dvi - co...
bista-Hough_Cordic.pdf
Home
CWTBook2Go - Home...
CWT
CERN Accelerating science
Dashboards
Projects
Issues
Agile
Create issue
Quick Search
FILTERS
New filter
Find filters
My Open Issues
Reported by Me
Recently Viewed
All Issues
FAVORITE FILTERS
You don't have any favorite filters.
Search
Save as
Share
Export
Tools
project = TOTEMDAQ AND resolution = Unresolved ORDER BY component ASC, priority DESC
Basic
TOTEMDAQ-3
Confirm results from Nicol's thesis
TOTEMDAQ-40
Activity
All
Comments
Work Log
History
Activity
Commits
Git Commits
Subversion
Source
Reviews
✓ Helps in tracking issues in development and production phase for both users and developers.
✓ Facilitates sharing information between users, avoids long loops of e-mails!
TOTEMDAQ-26
UDP algorithm tests
TOTEMDAQ-35
OptoRx processing
TOTEMDAQ-17
RT2014
TOTEMDAQ-36
Prepare poster and paper about UDP algor...
TOTEMDAQ-27
allign SRS Readout software (stand-alone ...
TOTEMDAQ-29
DTCC review
/FEC/Config_synplify/FEC_OptoRx.prj
MODIFY /TDAQ/Developers/branches/UDP_test/FEC/Source/Manifest.py
DEL /TDAQ/Developers/branches/UDP_test/FEC/Source/SLink_emulator.sv
MODIFY /TDAQ/Developers/branches/UDP_test/Common/Source/IDSRegisters/SRSRegisterMap.docx
MODIFY /TDAQ/Developers/branches/UDP_test/Common/Source/IDSRegisters/SRSRegisterMap.regmem.sv
MODIFY /TDAQ/Developers/branches/UDP_test/FEC/Source/TOTEM_appUnit.fdc
MODIFY /TDAQ/Developers/branches/UDP_test/FEC/Source/OptoRxReceiver.sv
MODIFY /TDAQ/Developers/branches/UDP_test/FEC/Source/TOTEM_appUnit.sv
MODIFY /TDAQ/Developers/branches/UDP_test/OptoRx/Config/OptoRx.qsf
EN
14:17
14/06/2014

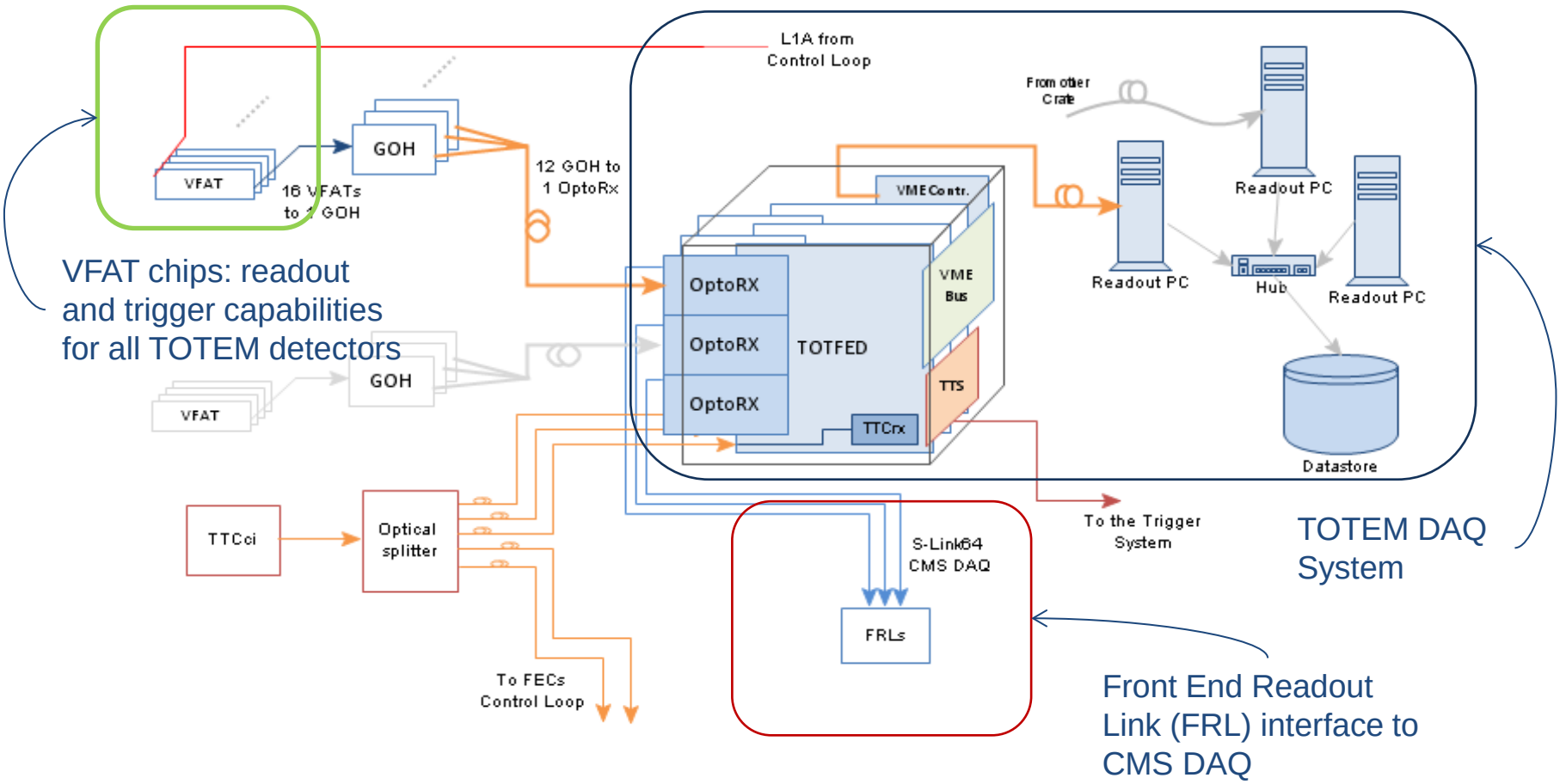
Outlook and future work

- SRS System has been fully qualified for the TOTEM DAQ Consolidation program:
  - Stable readout under stress conditions for several hours.
  - Successful of field test at IP5.
  - Full back-compatibility with legacy system (TTC System, Software).
- Target data reduction and 2nd level trigger algorithm leveraging the FPGA processing power on SRS.
- Port the design to FEC_V6 beta cards, we have received 2 FEC_V6.
- Proceed with purchasing of full set of hardware ~20FEC.
- Extend DTCCL Links SRU <-> ~20 FECs.


Thank you for your attention!
Monopoli PT

Back-up

The Current TOTEM DAQ System Architecture



VFAT chips: readout and trigger capabilities for all TOTEM detectors

TOTEM DAQ System

Front End Readout Link (FRL) interface to CMS DAQ

- In the TOTEM standalone configuration, the VME bus bandwidth limits the trigger rate to 1kHz.