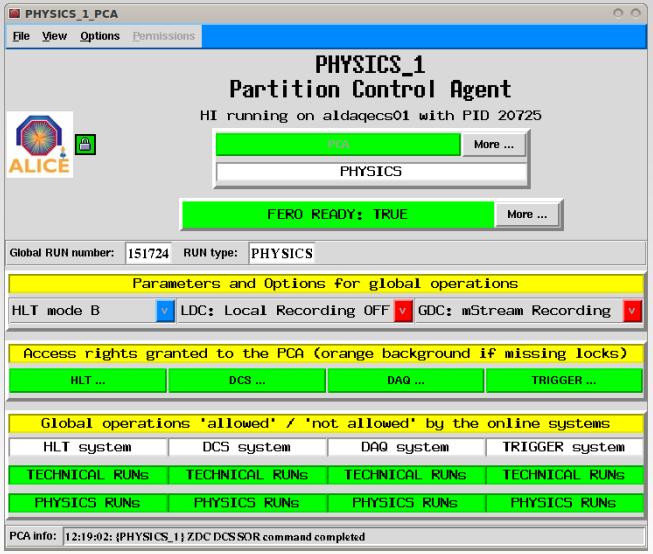
# The new frontier of the DATA acquisition using 1 and 10 Gb/s Ethernet links

**Filippo Costa** on behalf of the ***ALICE DAQ group***

# DATE software

# DATE (ALICE Data Acquisition and Test Environment)





**ALICE** is a general purpose detector designed to study the physics of strongly interacting matter and the quark-gluon plasma in nucleus-nucleus collisions at the ***CERN Large Hadron Collider*** (**LHC**).
The software framework of the **ALICE DAQ (data acquisition system)** is called

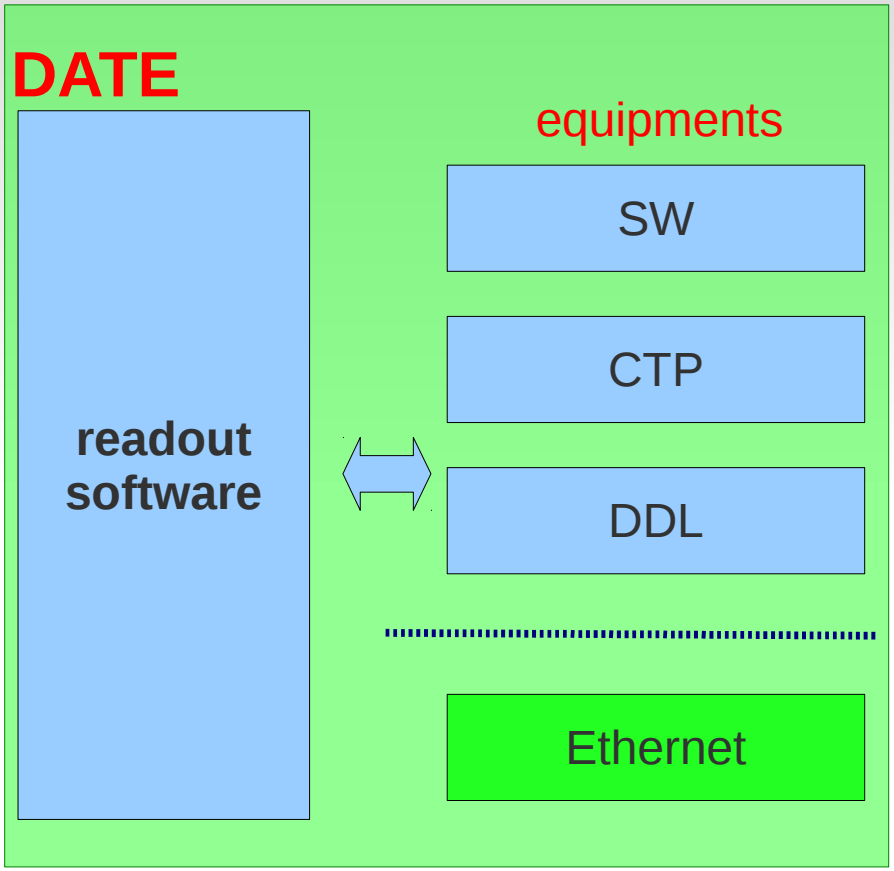
# DATE

**(Data Acquisition and Test Environment)**

The current setup can deliver**:**

- a readout capability up to **1 Tb/s**,
- an aggregate event-building bandwidth above **7.5 GB/s**,
- a storage capability up to **4.5 GB/s** to mass storage.

# How DATE collects data?

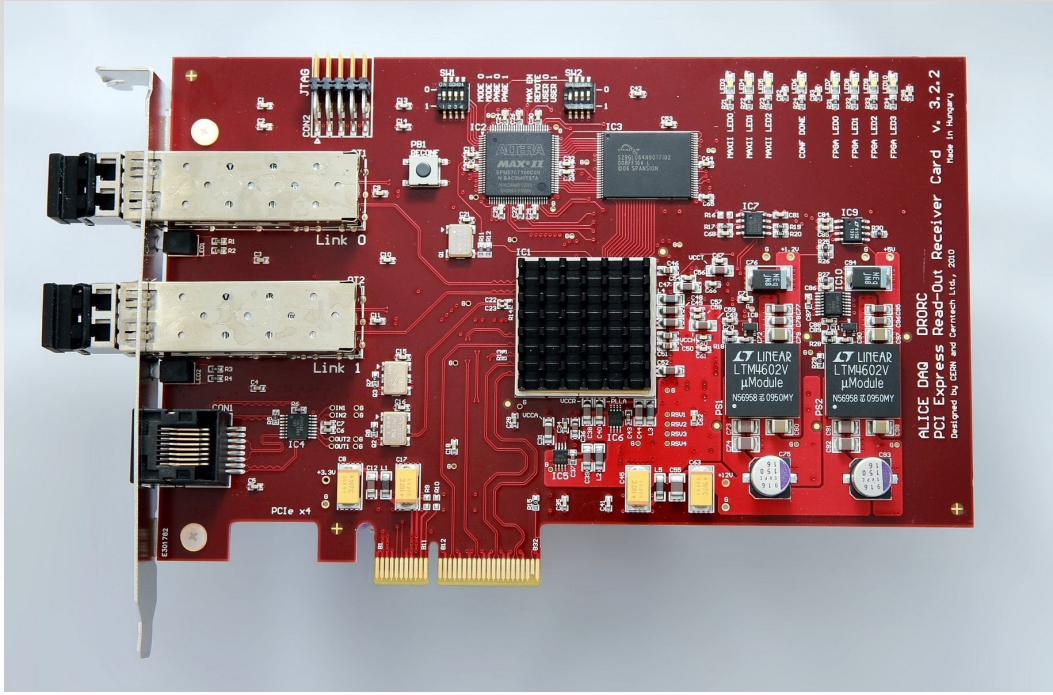


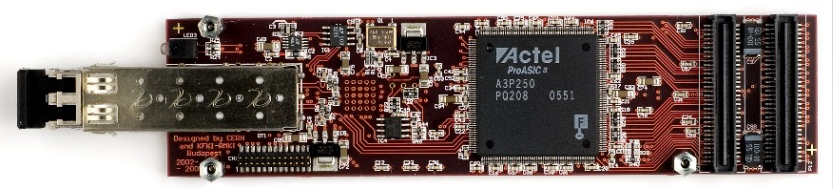

***Readout*** is the software package, part of DATE, that collects data.
The ***equipments*** are software modules used by **readout** to read data from different data sources.

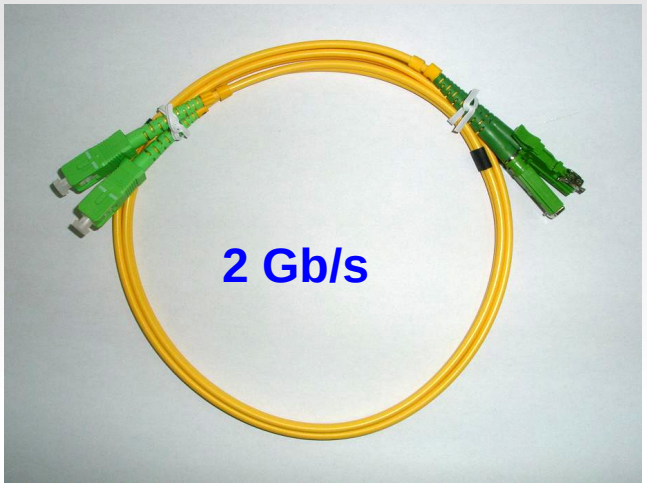
Used by ALICE experiment to collect data.

New Ethernet equipment.

# RORC & the DDL




The complete ALICE DAQ system contains about **500 DDLs**, that currently transmit in total **2.5 GB/s** event data, with a dead-time of less than 10%.

**READOUT SPEED**

| | 2010 | 2011 |
|---|---|---|
| **p-p** | 1.2 GB/s | Readout speed 20 GB/s assuming a compression of a factor ~7 |
| **pb-pb** | 2.5 GB/s | |

# *The Ethernet equipment ... Why to develop it?*

# HIGH PERFORMANCE @ 10 Gb/s

| | |
|---|---|
| Current Trigger rate | 11198.800 |
| Average Trigger rate | 10962.062 |
| Number of sub-events | 1589499 |
| Sub-event rate | 11198 |
| Sub-events recorded | 1589496 |
| Sub-event recorded rate | 11198 |
| Bytes injected | 140041220040 |
| Byte injected rate | 986.641 MB/s |
| Bytes recorded | 140040779520 |
| Byte recorded rate | 986.641 MB/s |

**THROUGHPUT 990 MB/s**

# Hardware and spare components

SUN 10 GBE XFP SR PCI Express

INTEL 10 Gb AT PCI Express

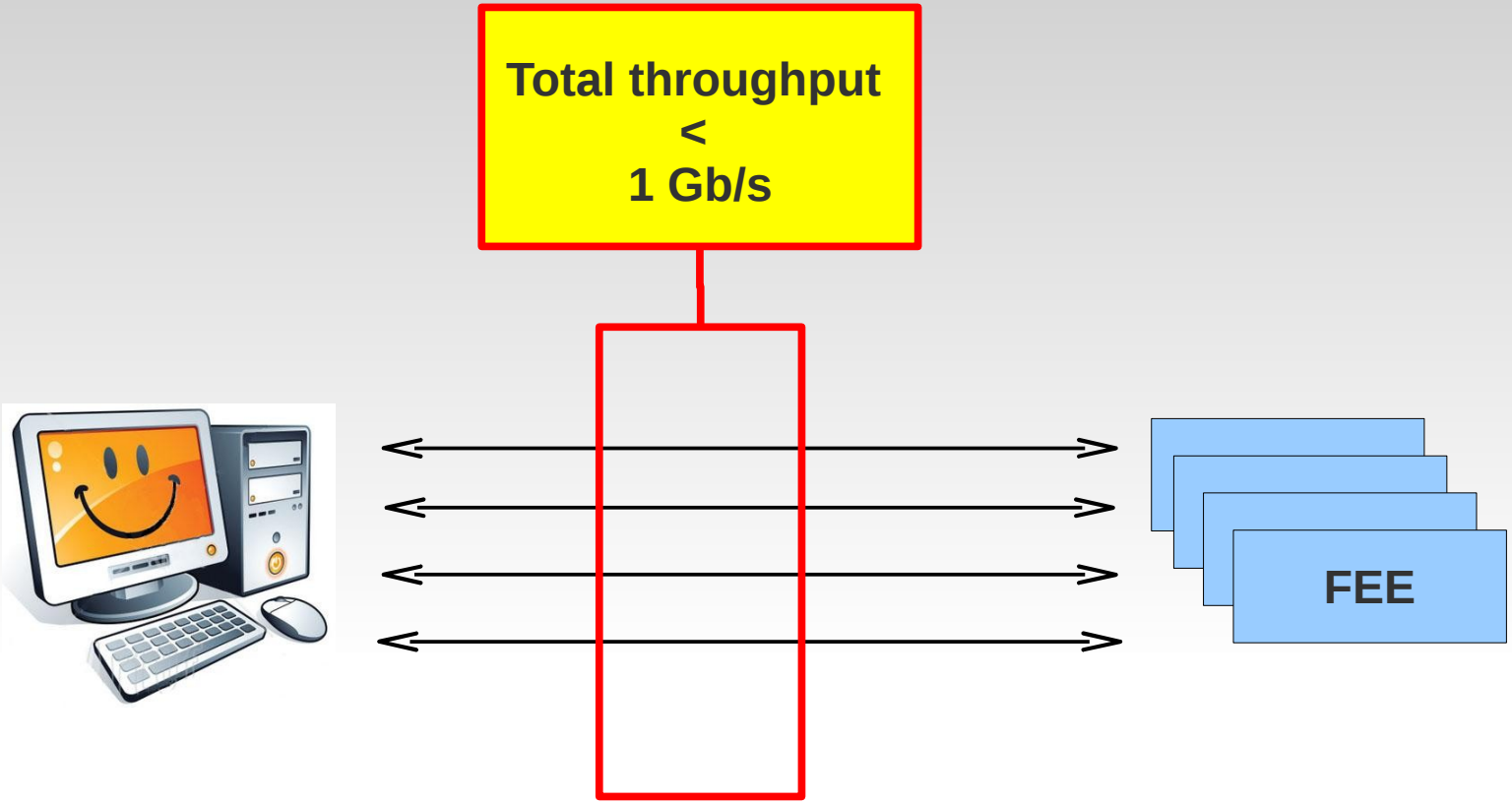
HP 10 Gb Dual Port PCI Express

Commercial transmission links with a throughput of **10 Gb/s** are a reality at an affordable price. Companies like **INTEL, HP, SUN**, provide several network boards with different options:

- ***single/dual port,***
- ***copper or optical fiber.***

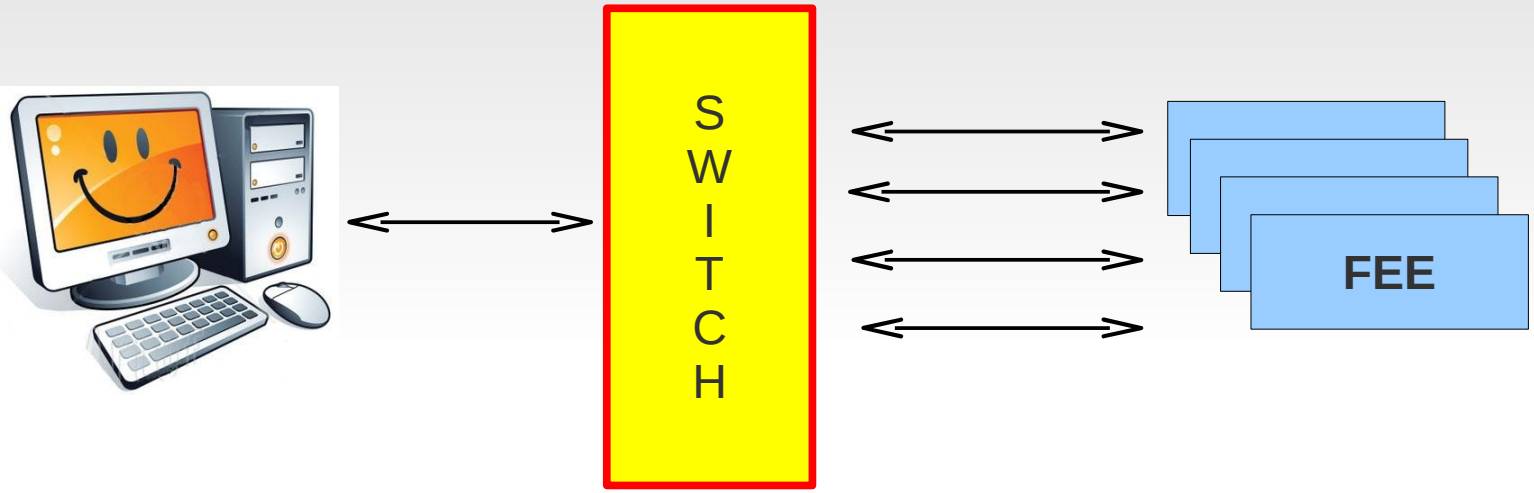
Spare components are easy to find, available on the market.
The driver, provided by the company itself, ensures the compatibility of the board with commonly used O.S.s

# Flexibility of the system



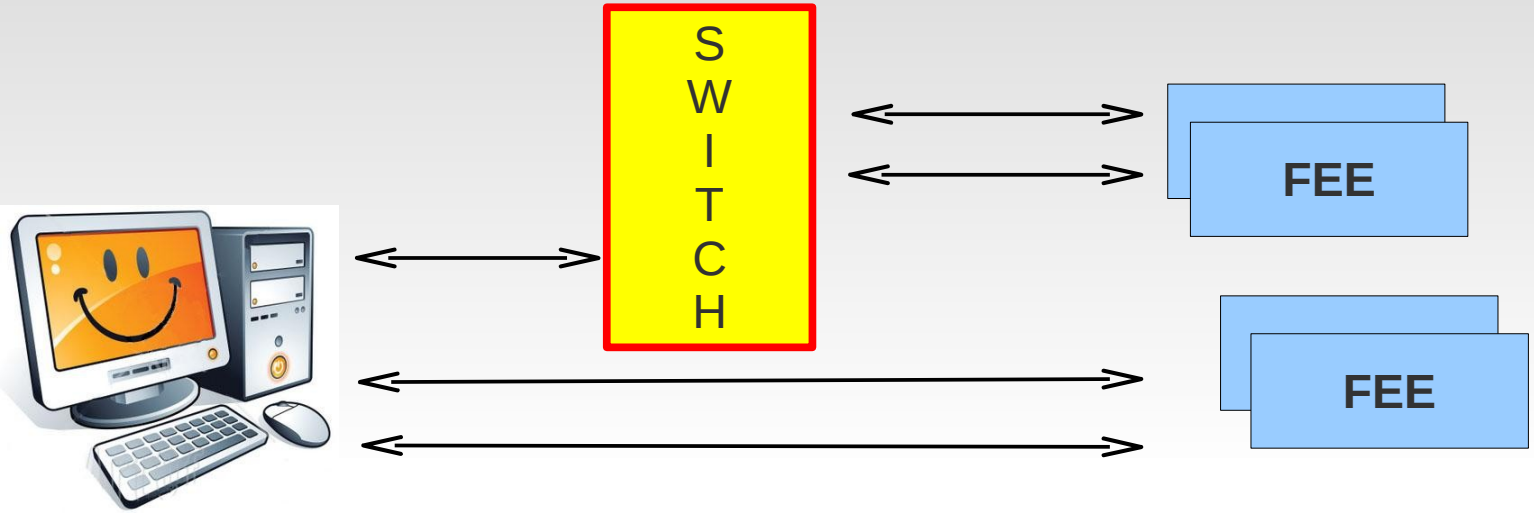

With some links you are forced in a P2P (point to point) configuration.
To connect all the readout board you need to install several cards, even if the total throughput could be read with a single link.

# Flexibility of the system



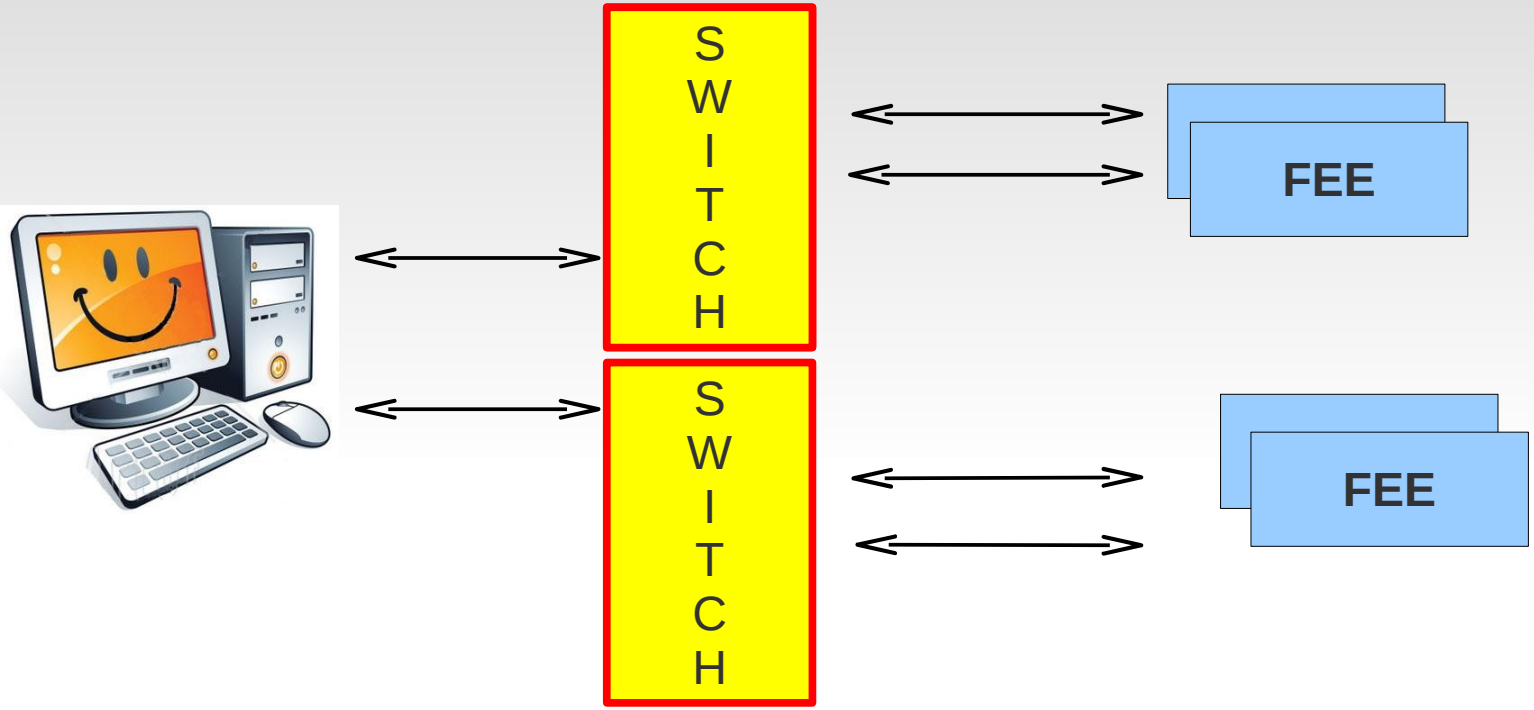

Using a switch it would be possible to collect data coming from all the FEE using a single NIC (network interface card) installed in the DAQ PC.

# Flexibility of the system




Still it would be possible to use a P2P connection if the system requires specific configuration.

# Flexibility of the system

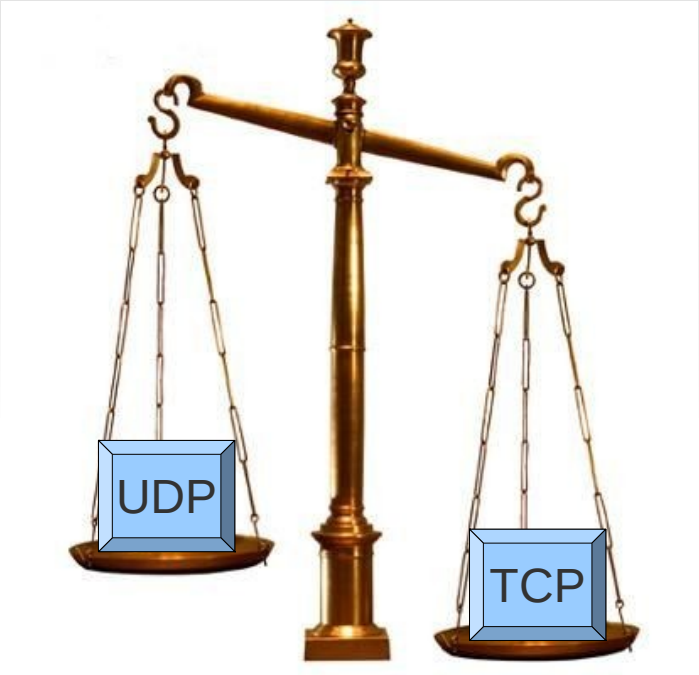



Or share the bandwidth in 2 switches if the throughput is too big to be accepted by a single one

# *Protocol*

# Used protocol

## TCP or UDP?



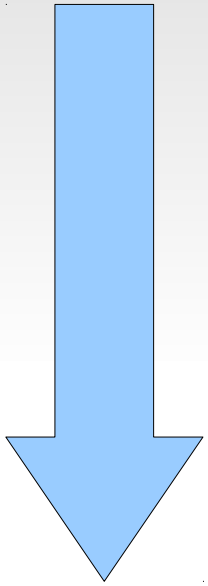

For the DATE Ethernet equipment we decided to choose one of these two protocols.
But which one was the most appropriate to be used in a real data acquisition system?

# TCP / UDP main characteristics

| TCP | UDP |
| --- | --- |
| **The GOOD:**<br>• provides reliable, ordered delivery of a stream of bytes. | **The GOOD:**<br>• simple and fast protocol,<br>• easy to be implemented in hardware,<br>• does not require big resources. |
| **The BAD:**<br>• heavy protocol to be implemented in hardware,<br>• each packet needs to be acknowledged and at high rate can become a problem. | **The BAD:**<br>• no reliability, ordering, or data integrity provides an unreliable service and datagrams may arrive out of order. |

# UDP chosen for DATE

UDP



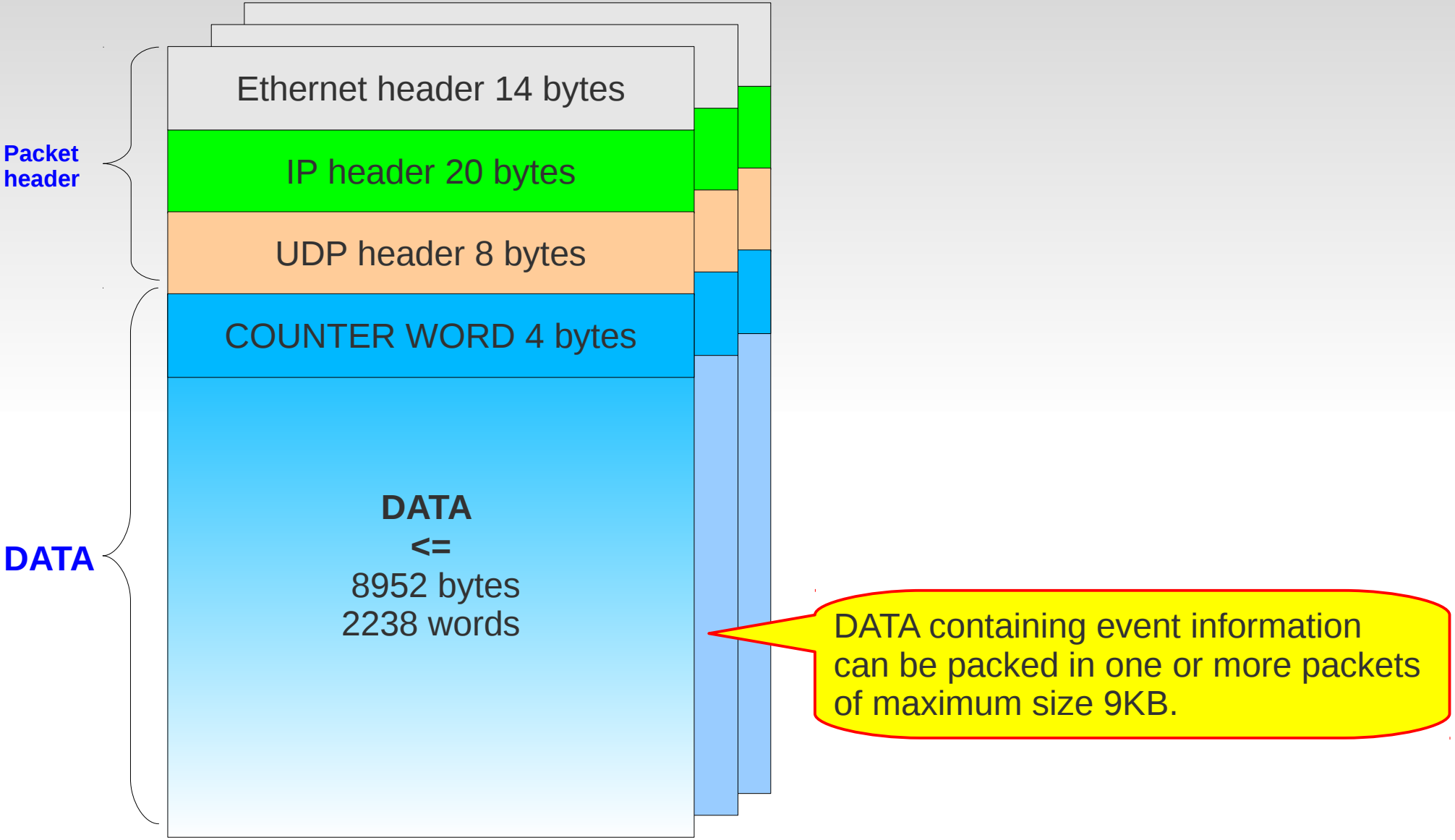
We decided to use the **UDP protocol** for the Ethernet Equipment in DATE.
Even if it is an unreliable protocol, can be easily implemented in hardware and the needed checks to ensure a good data acquisition can be added in the software

DATE

# Data format

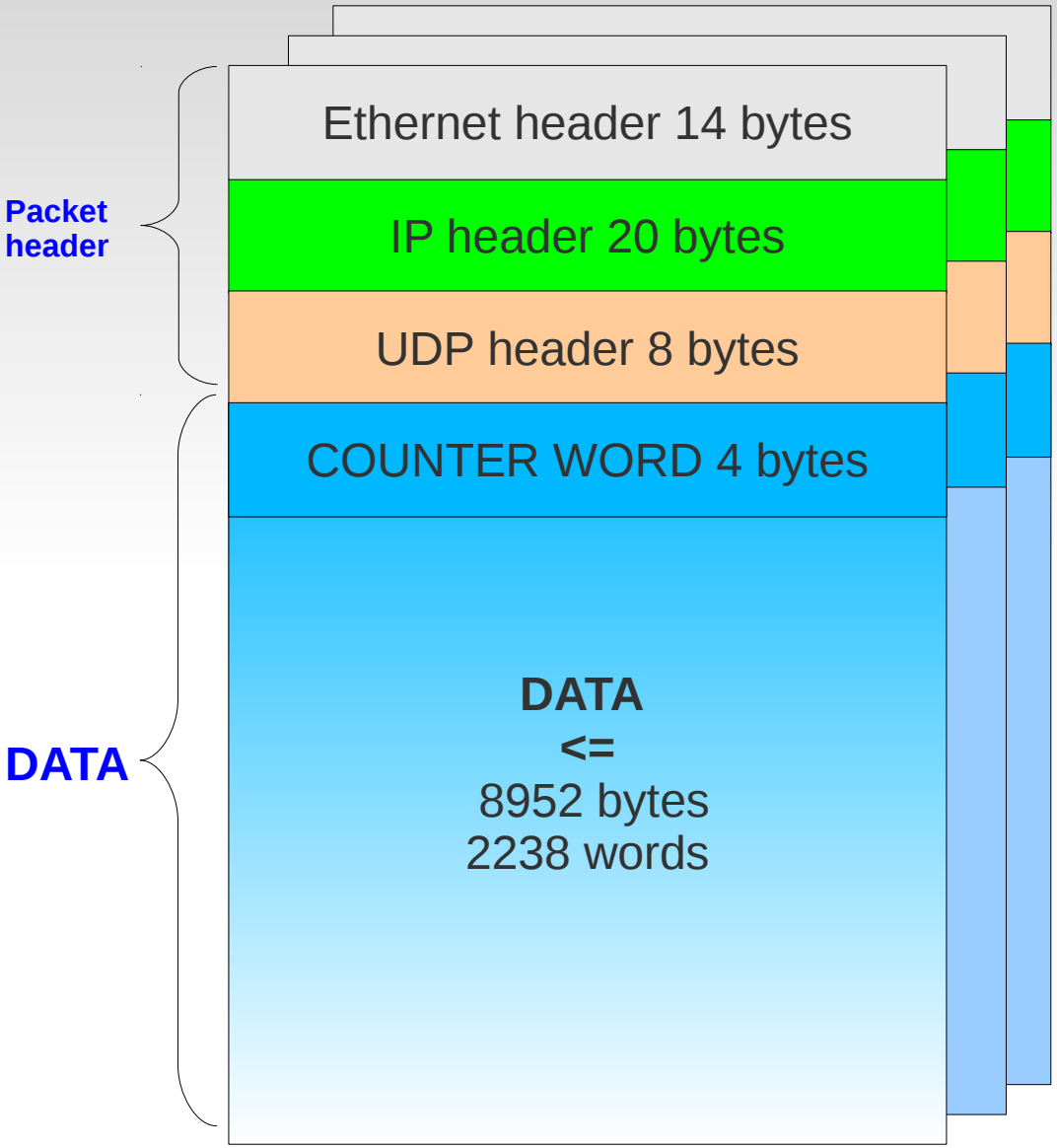
# Data format

**UDP packets of ~9KB (JUMBO FRAMES)**

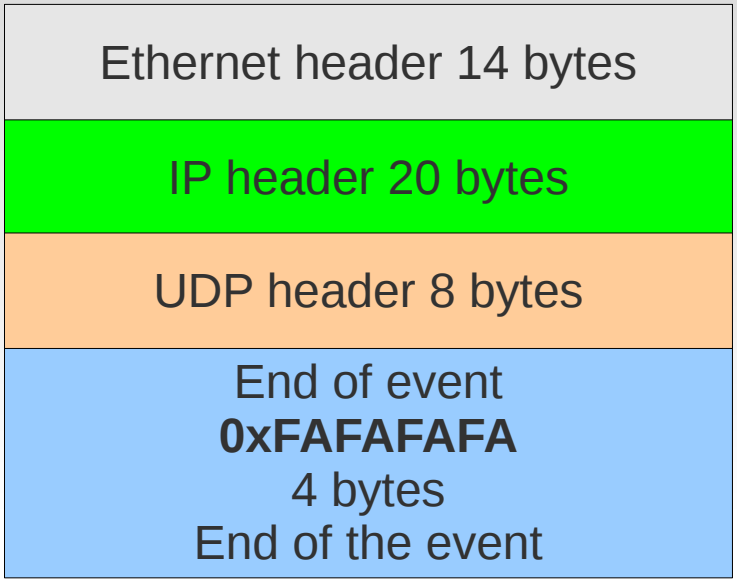


# Data format

**UDP packets of ~9KB (JUMBO FRAMES)**



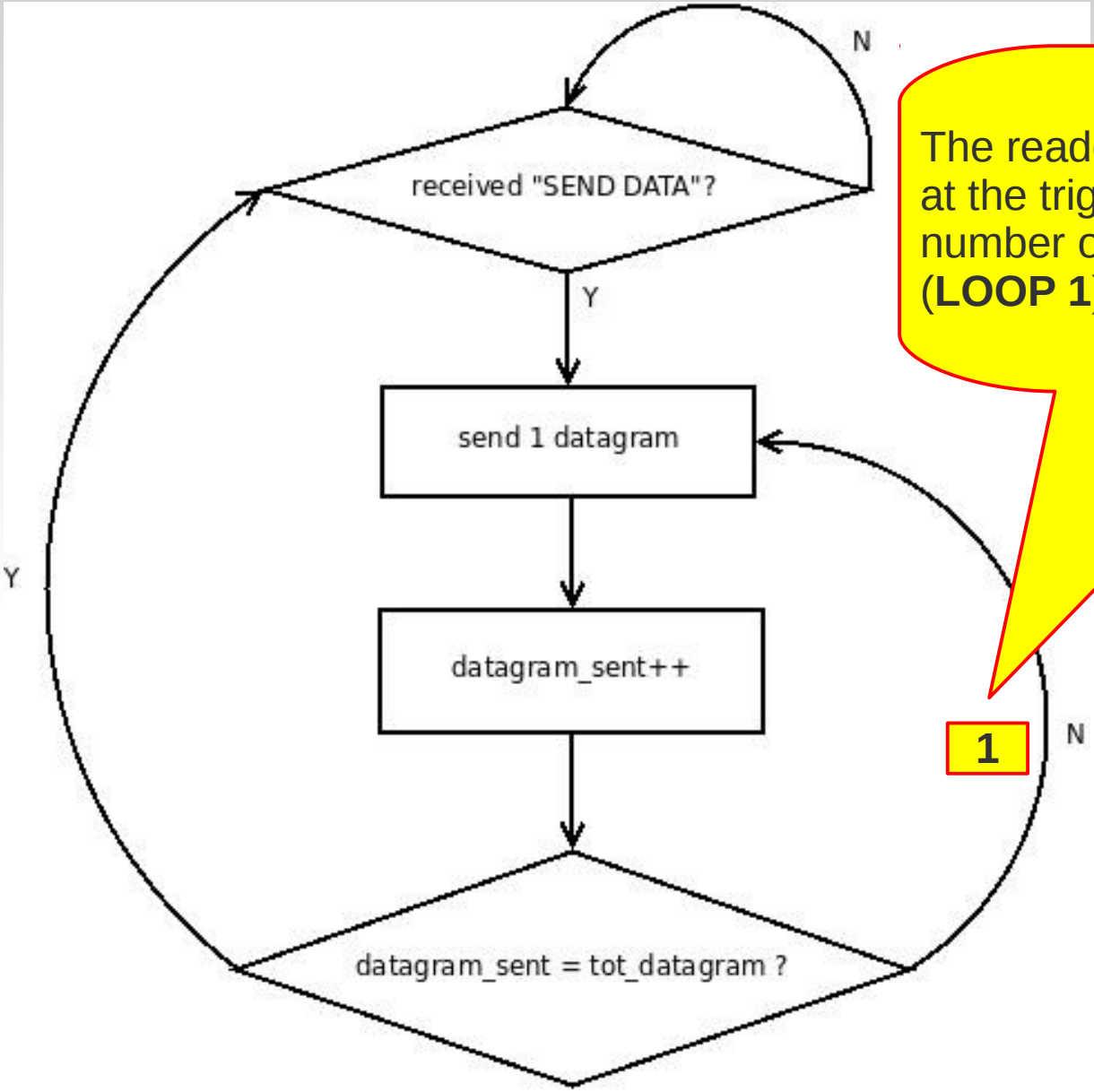

**UDP packet of 46 bytes**




The event is closed when **DATE** receives and recognizes the word **0xFAFAFAFA**

# Busy algorithm

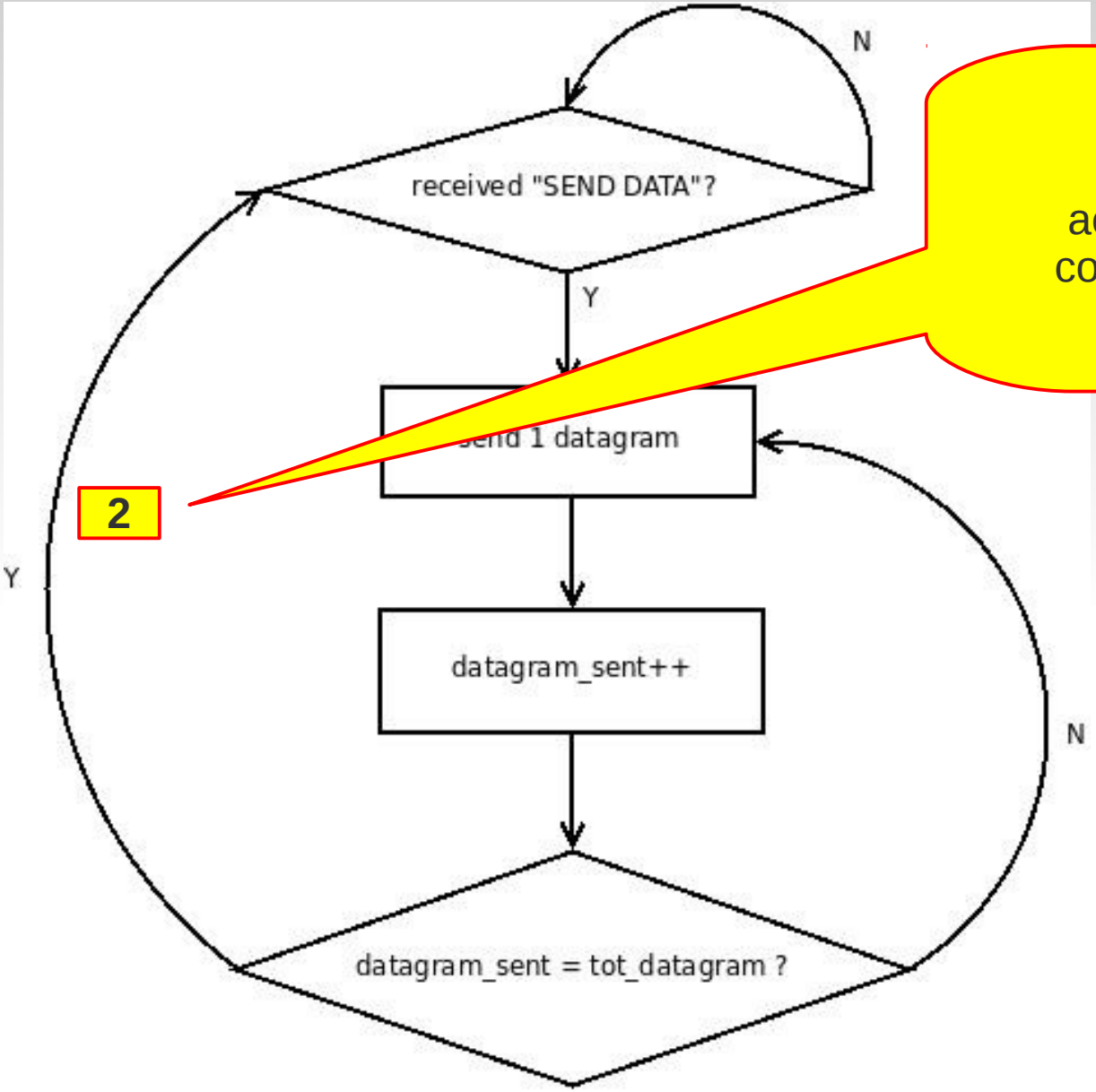
# BUSY algorithm FEE




The readout board will send data to **DATE** at the trigger rate until it reaches tot_datagram number of packets sent (**LOOP 1**).

**1**

# BUSY algorithm FEE




It will wait for an acknowledgment from **DATE** to continue to send data (**LOOP 2**).

# *Error handling*

# Error checks in the software

| | | | | |
|---|---|---|---|---|
| FATAL | 19:07:34 | pcaldrefudp | equipmentList_ | Arming RorcData: (ERROR 12204) the readout board (10.0.0.6) is not responding to the ping |
| ERROR | 19:07:34 | pcaldrefudp | readout | Error 12204 in routine ArmHw active equipment 2 |
| FATAL | 19:07:34 | pcaldrefudp | readout | Fatal error in SOR phases, see details above |
| ERROR | 19:07:35 | pcaldrefudp | runControl | READOUT start phase timeout on ALONELDC |

FEE

V

X

During the initialization, the software checks if the FEE is on and reachable, otherwise there is no reason to continue with the start of run procedure

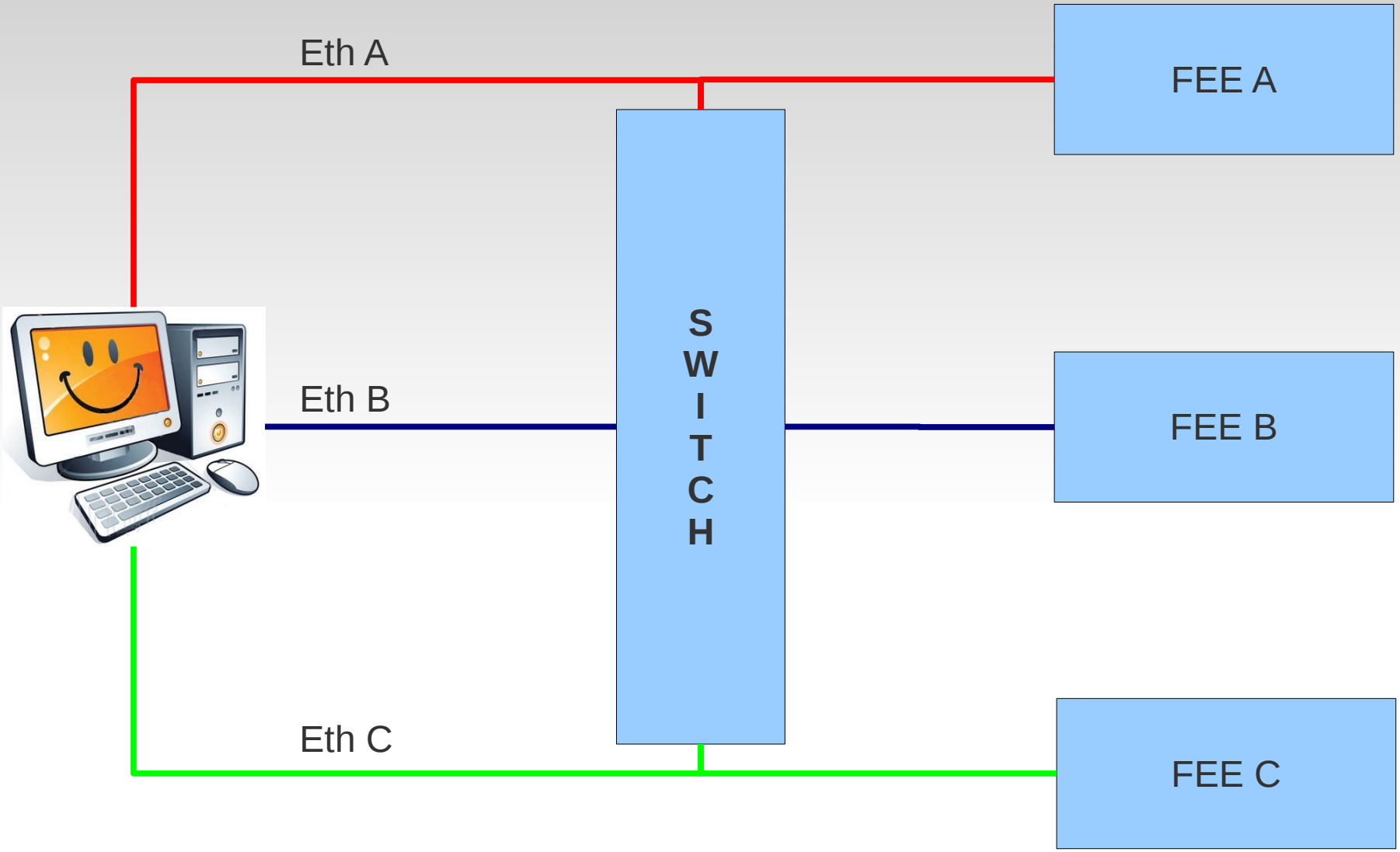
# Error checks in the software

| | | | | |
|---|---|---|---|---|
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 0 instead 9 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 1 instead 10 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 2 instead 11 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 3 instead 12 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 4 instead 13 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 5 instead 14 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 6 instead 15 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 7 instead 16 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 8 instead 17 ... run continues |
| ERROR | 18:44:44 | pcaldrefudp | equipmentList_ | PACKET ORDER MISMATCH (eqId 1) @ EV 10668 received 9 instead 18 ... run continues |

The UDP equipment checks the packet order, displaying error messages if a packet gets lost or duplicated.
The behavior of the software in this case is ***PASSIVE***, it does not take action during the data taking.
The message contains all the necessary information to find back where the problem happened:

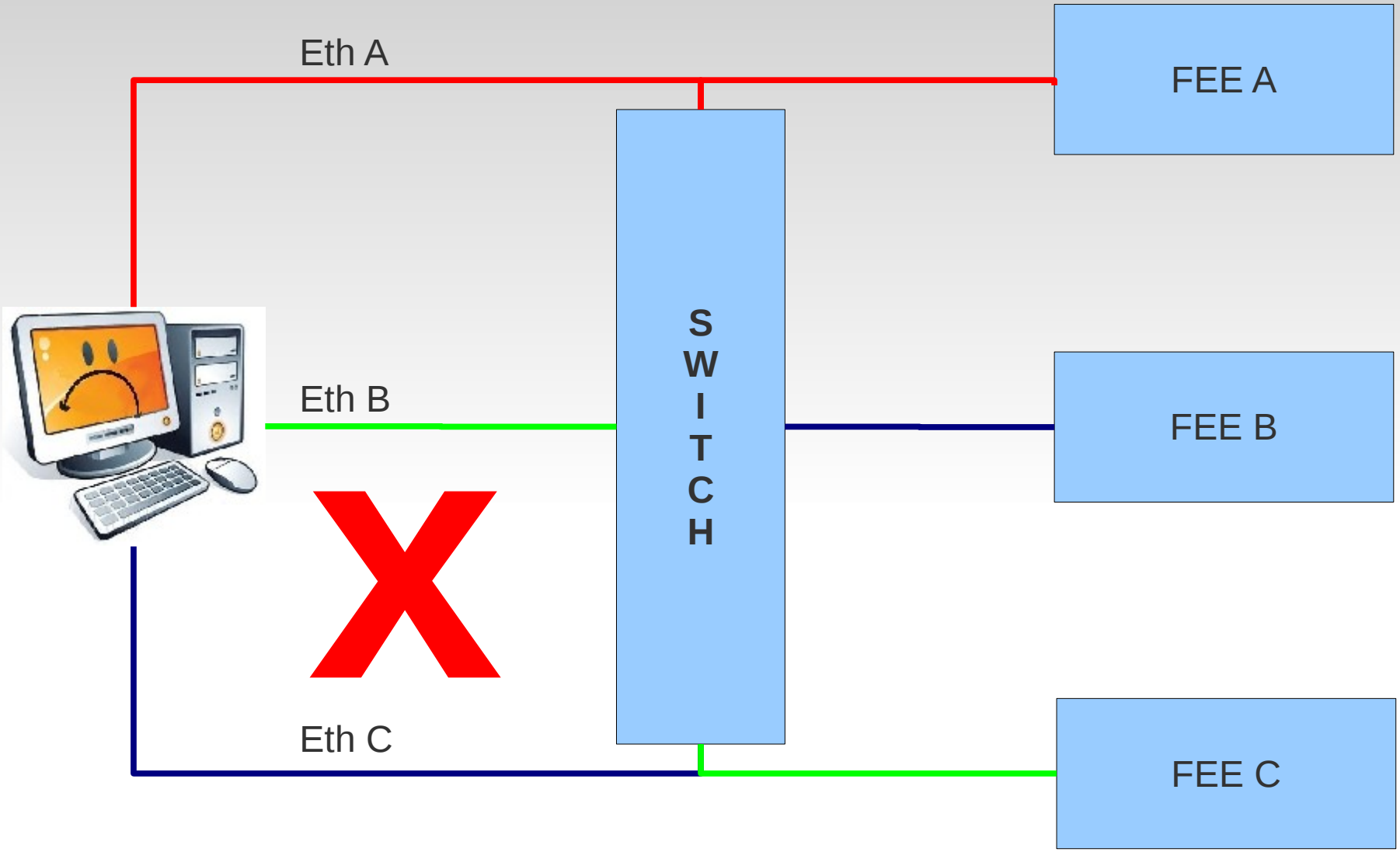
- **ID of the link,**
- **event number,**
- **which packet was lost or duplicated.**

# Error checks in the software




The software checks that a FEE board sends the information to the right network interface, as configured during the **slow control**.

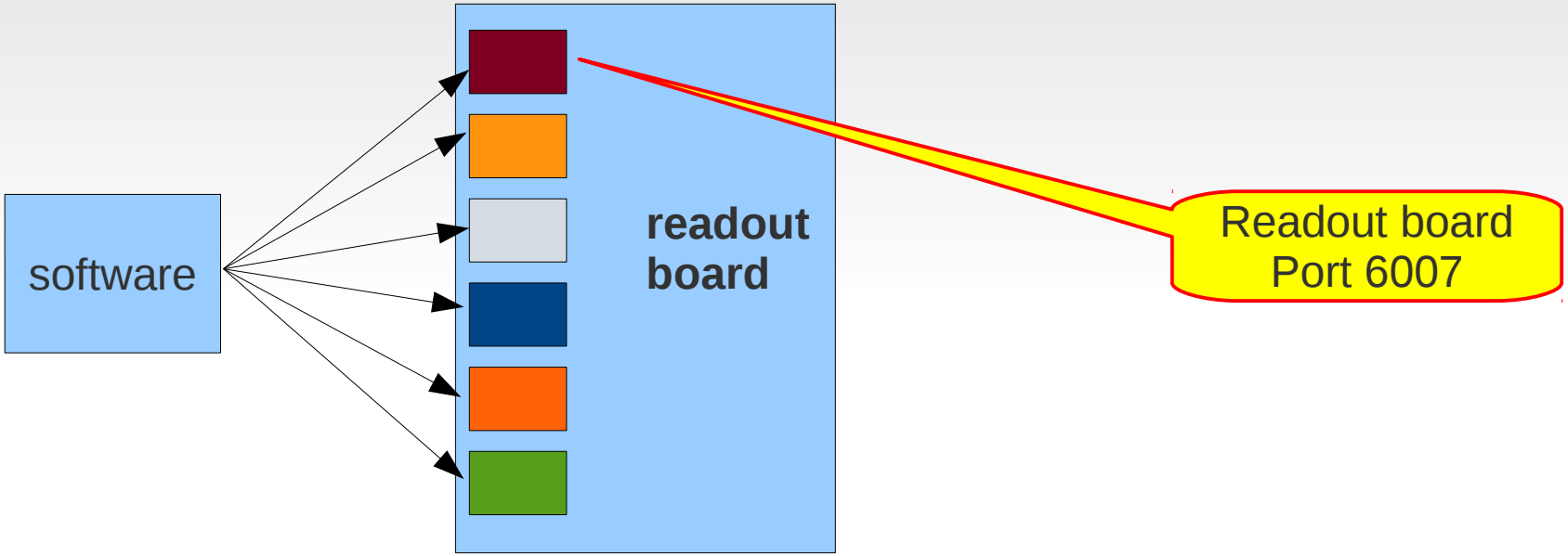
# Error checks in the software



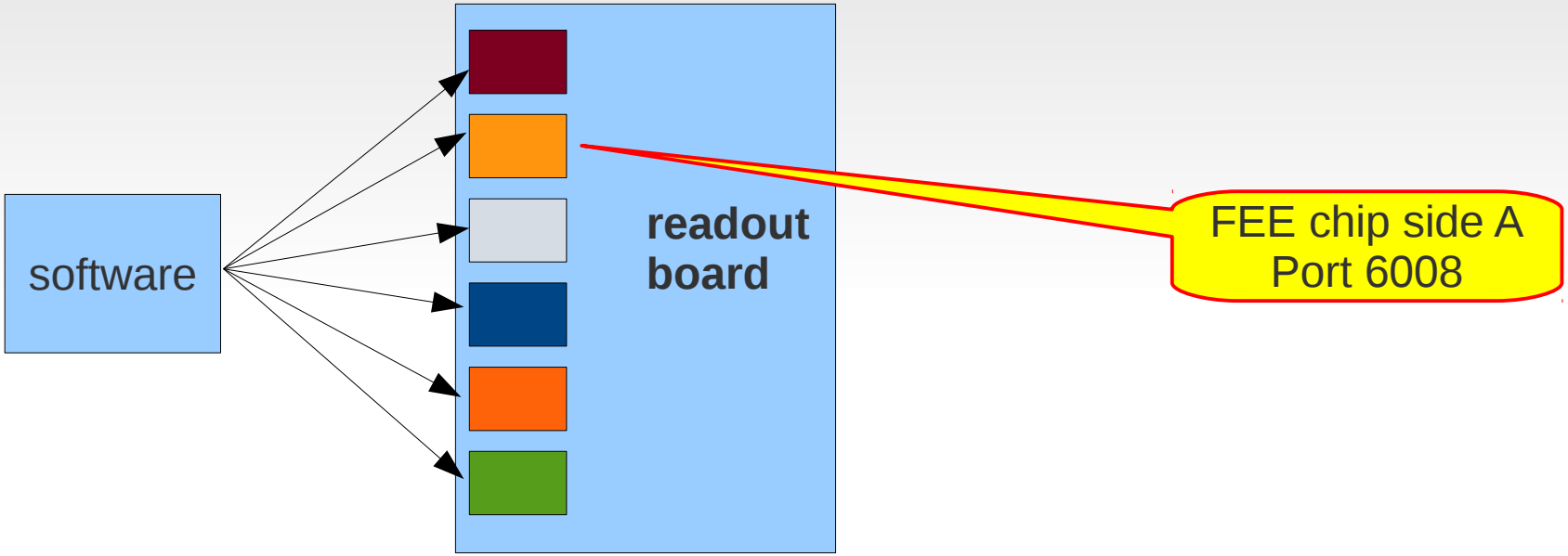
# *SLOW control*

# Slow control … IP and ports

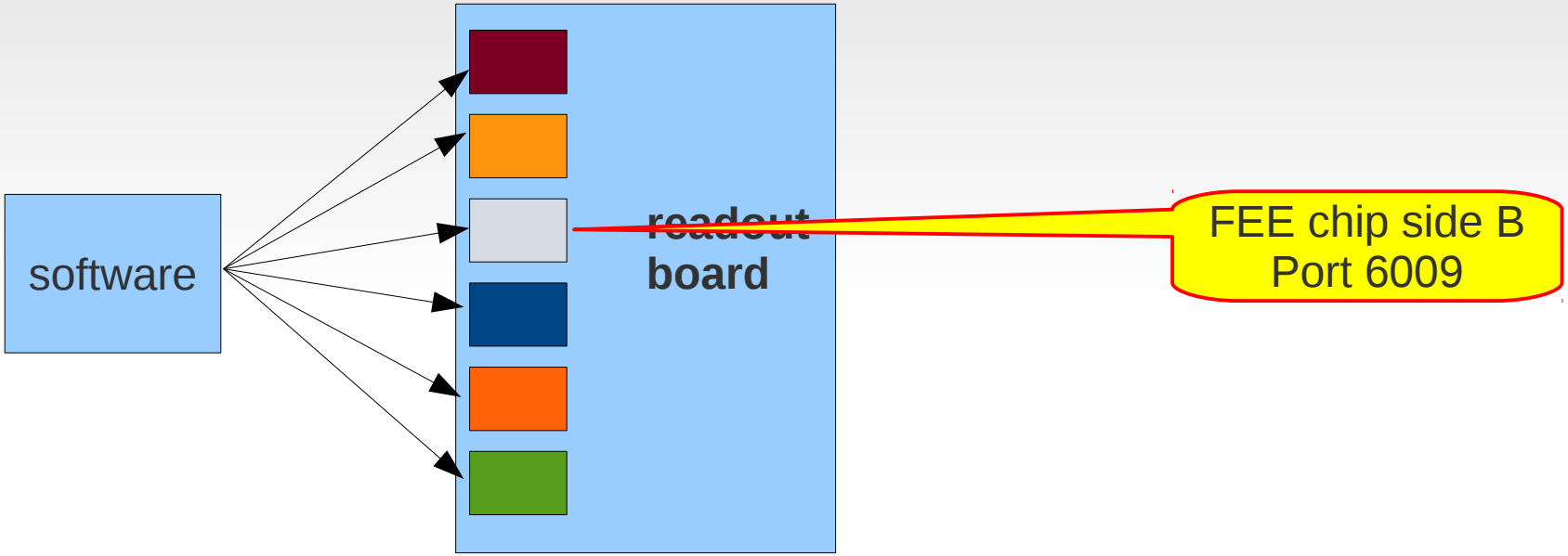
Using different ports **DATE** can address different components connected to the readout board



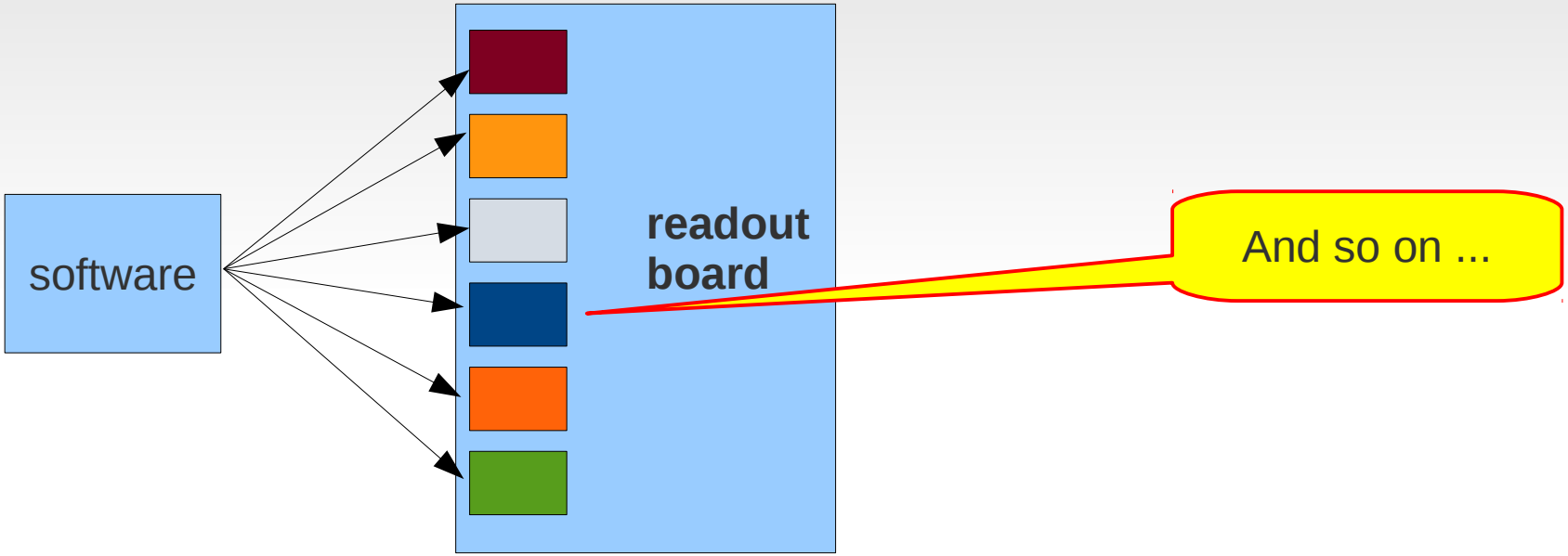
# Slow control … IP and ports



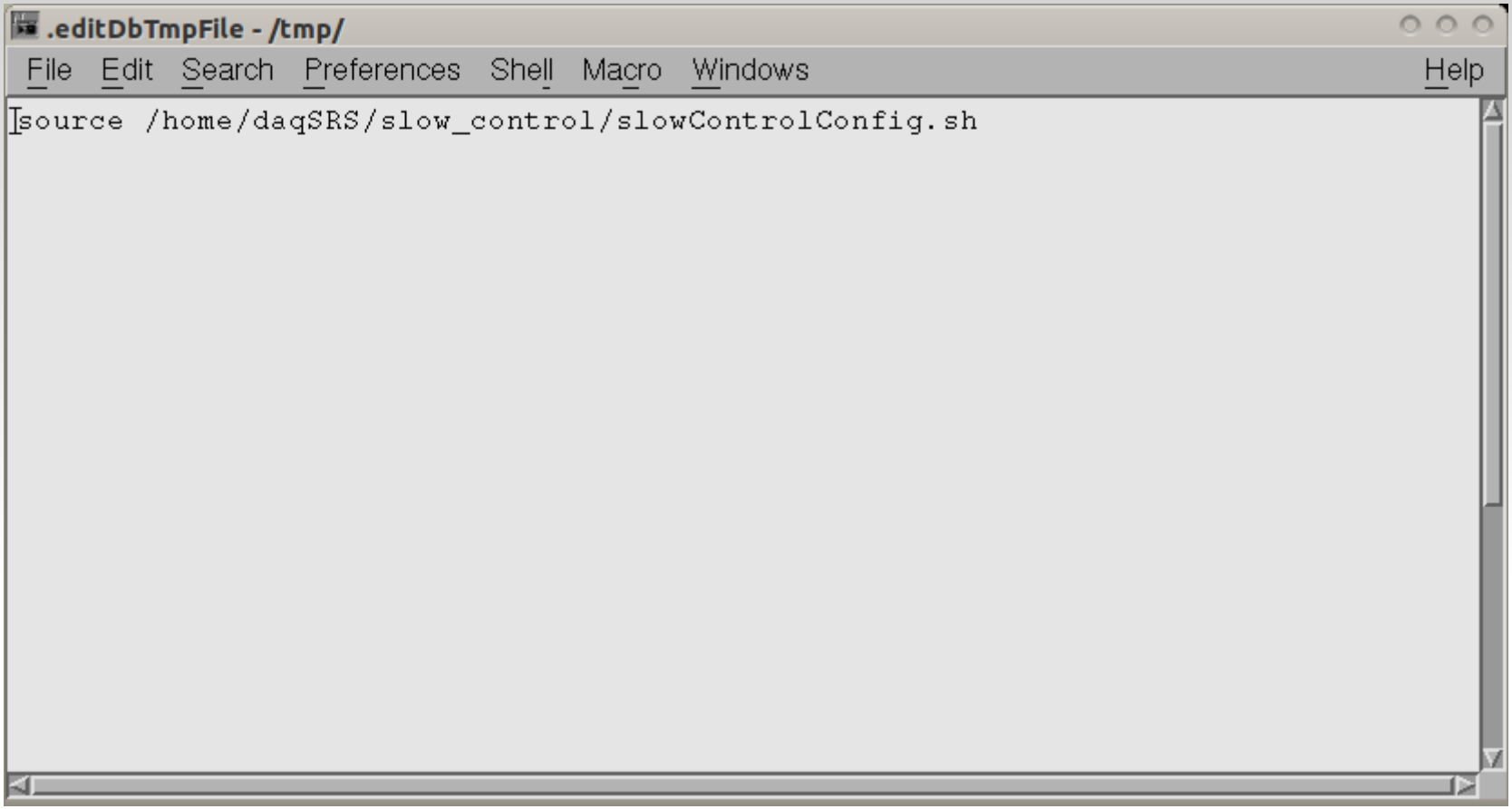
# Slow control … IP and ports



# Slow control … IP and ports



# Slow control … IP and ports




**DATE** will execute a script during the **Start Of Run phase** configuring the FEE using the same link used for the data taking.

# Slow control … IP and ports

```
#! /bin/bash

for i in 0 1 2 3 4 5 6
do
echo "======== CONFIGURATION OF SRS CARD $i"
echo ""
echo "set 10.0.$i.2 -> 10.0.$i.3"
/home/daqSRS/slow_control/slow_control /home/daqSRS/slow_control/set_ip$i.txt
usleep 100000
echo "ADC_$i config"
/home/daqSRS/slow_control/slow_control /home/daqSRS/slow_control/adc_card$i.txt
usleep 100000
echo "FEC_$i config"
/home/daqSRS/slow_control/slow_control /home/daqSRS/slow_control/fec${i}TextPulse.txt
usleep 100000
echo "APV_$i config"
/home/daqSRS/slow_control/slow_control /home/daqSRS/slow_control/apv$i.txt
usleep 100000
echo "PLL_$i config"
/home/daqSRS/slow_control/slow_control /home/daqSRS/slow_control/pll$i.txt
done
```

The script is a simple bash script … nothing fancy here

# Slow control … IP and ports

```
10.0.7.2
6039
80000000
00000000
aaaaffff
00000000
00000000
00000003
00000001
00000007
```

**IP of the board**

# Slow control … IP and ports

```
10.0.7.2
6039
80000000
00000000
aaaaffff
00000000
00000000
00000003
00000001
00000007
```

**PORT of the board**

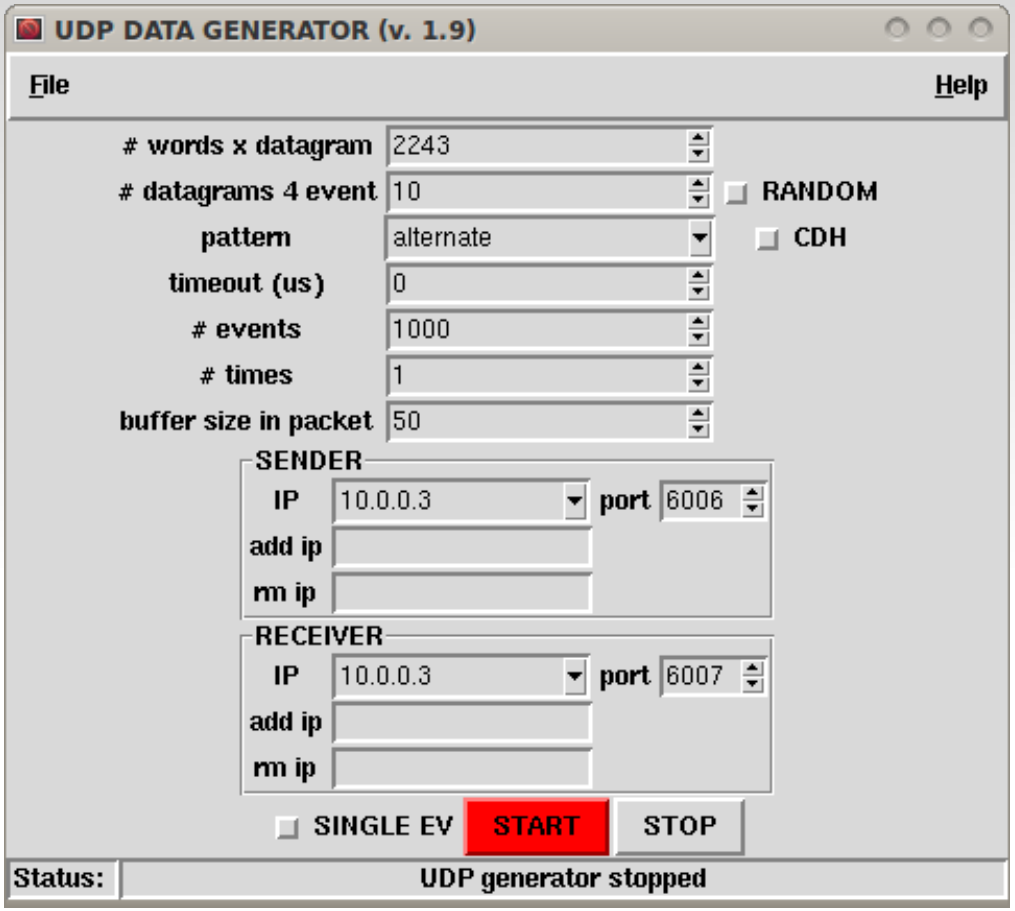
# Slow control … IP and ports

```
10.0.7.2
6039
80000000
00000000
aaaaffff
00000000
00000000
00000003
00000001
00000007
```

**CONFIG. data**

# *The SOFTWARE data generator*

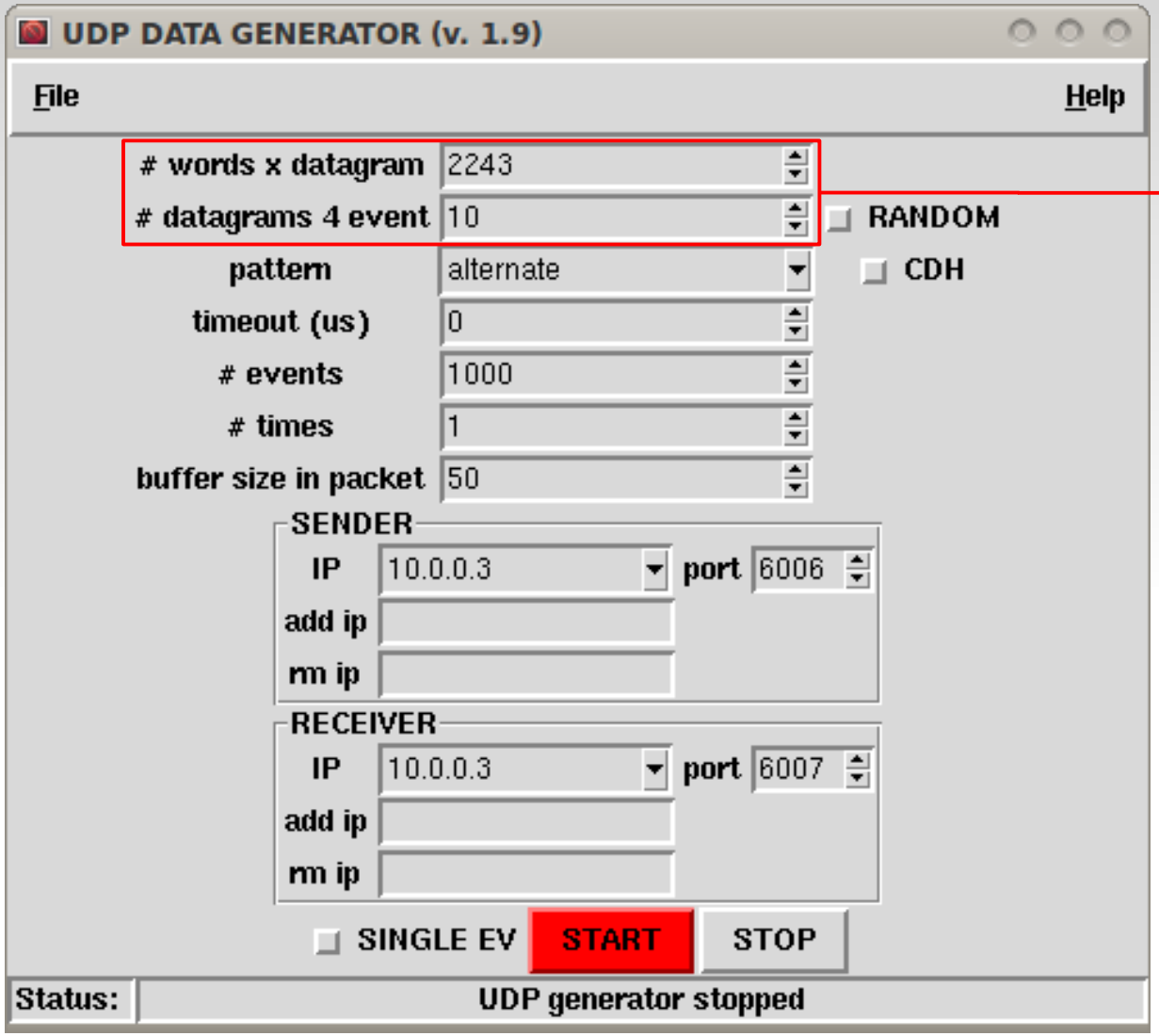
# SOFTWARE data generator




**UDP data generator**:

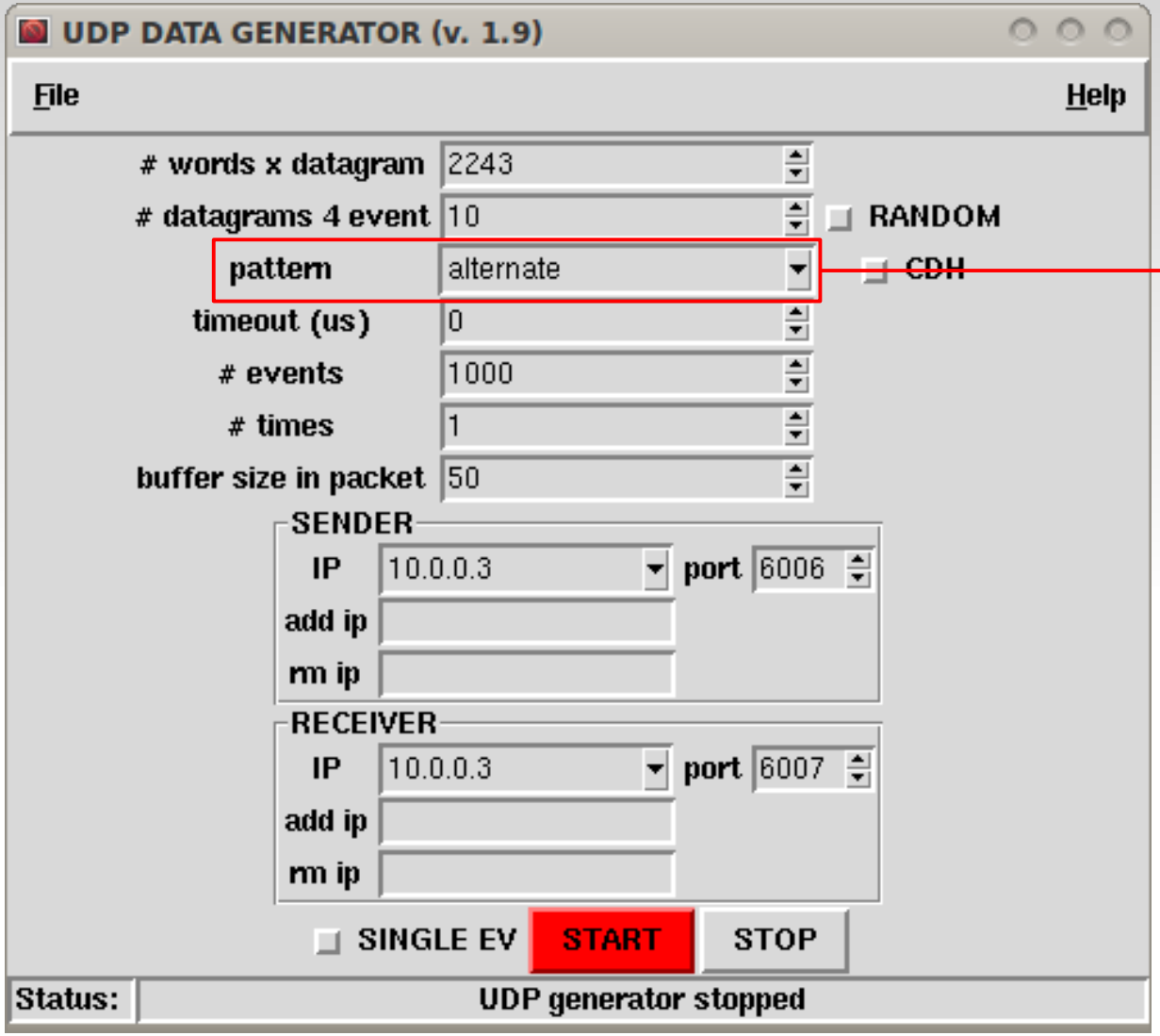
- part of the DATE package,
- simulates the behavior of the FEE,
- used to test the data acquisition,
- several parameters allow to modify the acquisition behavior.

# SOFTWARE data generator




size of each packet
packet number in a single event

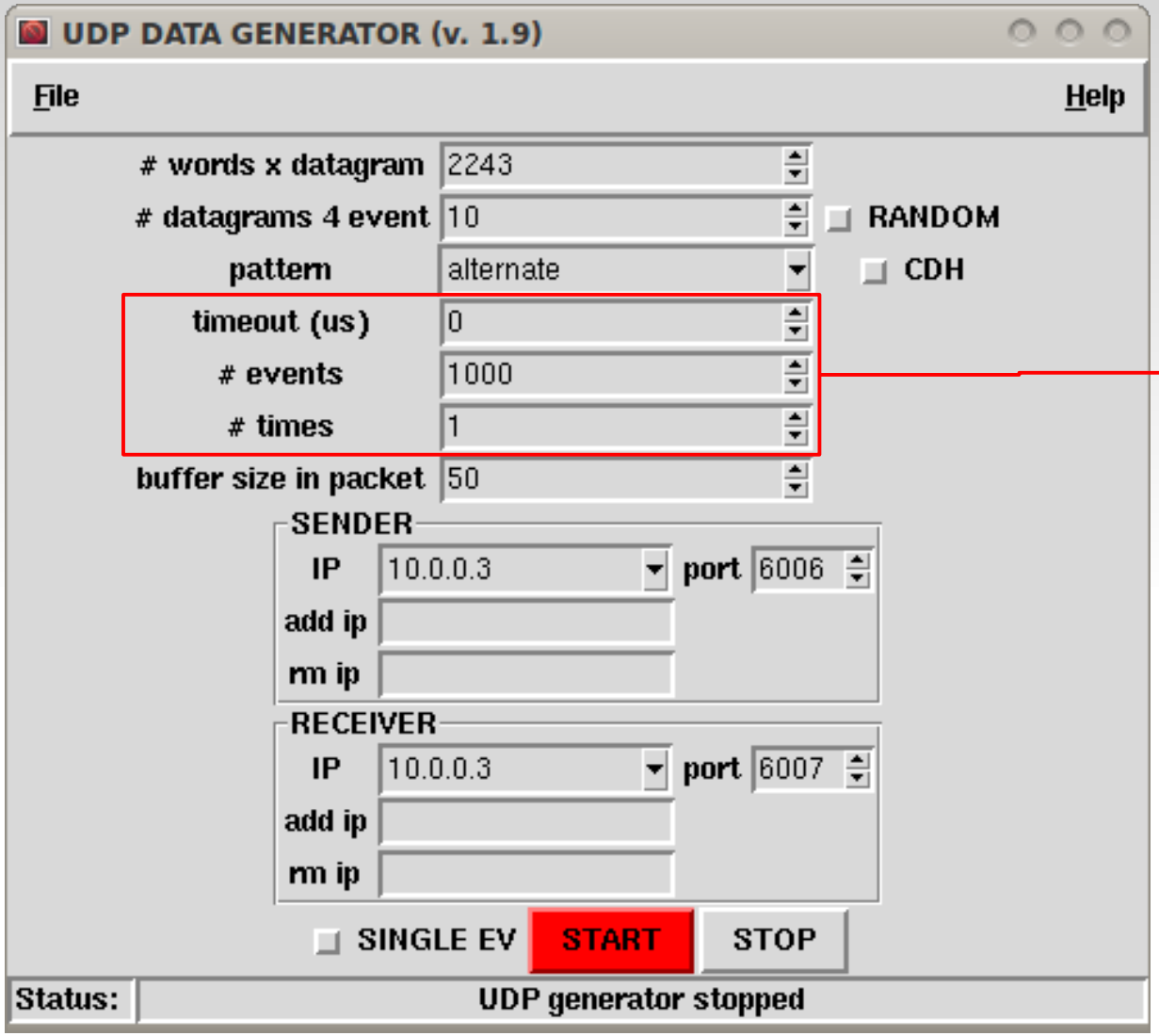
# SOFTWARE data generator




Pattern type:

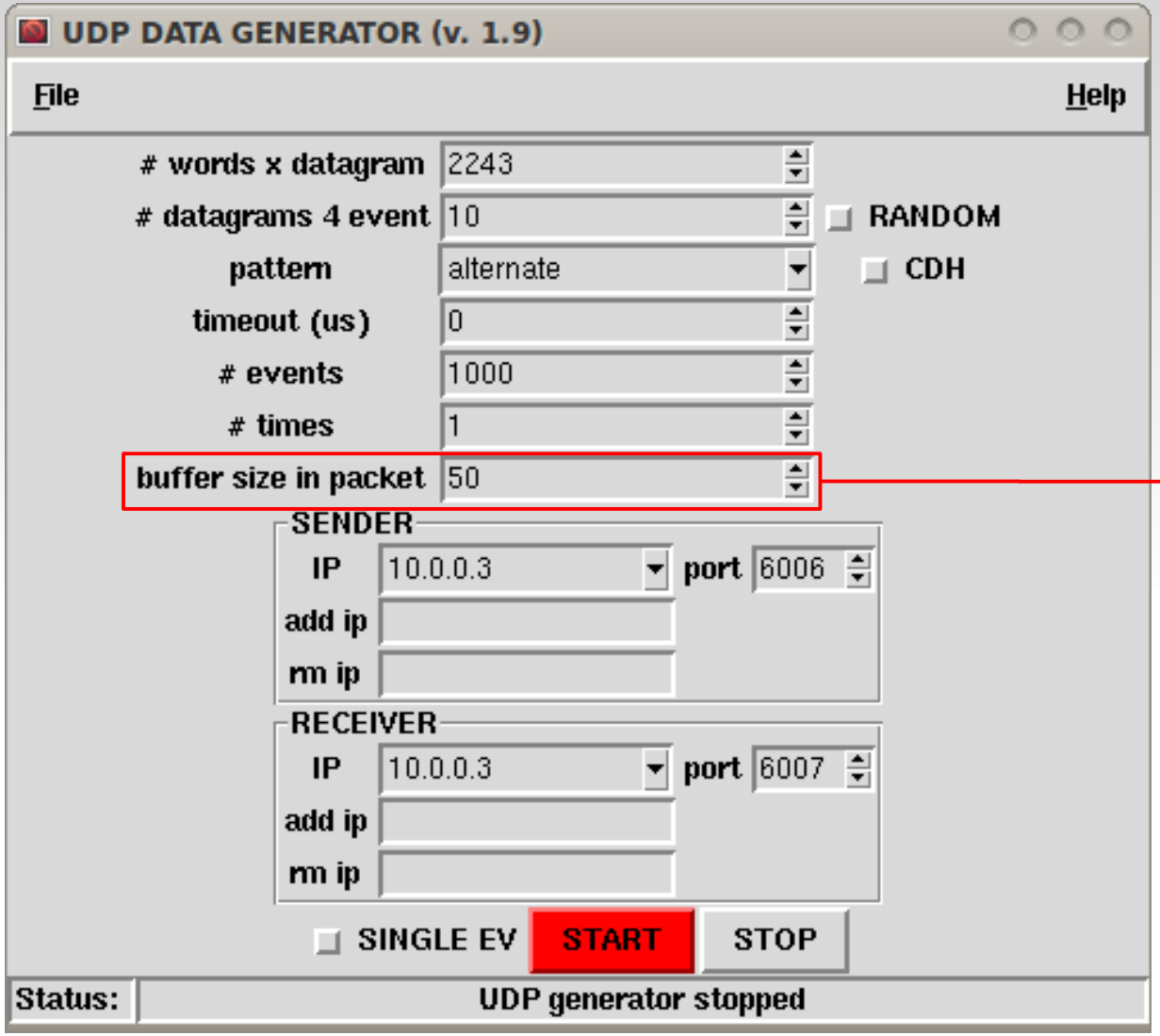
- alternate,
- constant,
- incremental.

# SOFTWARE data generator




Number of events to be sent out.
Timeout between 2 consecutive events

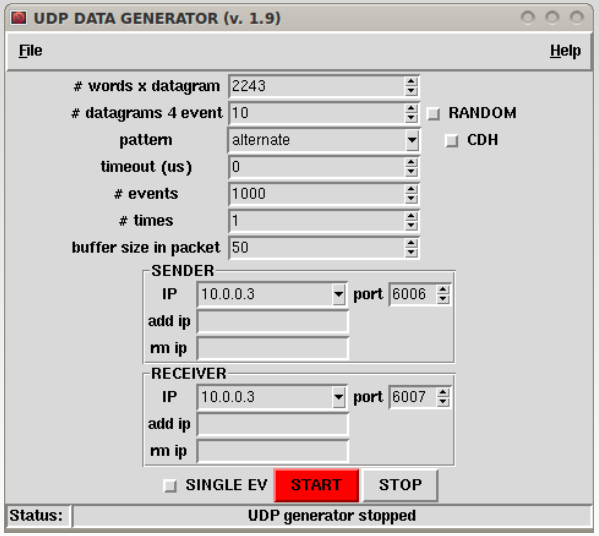
# SOFTWARE data generator




Number of packets to be sent out before waiting for an ack. from DATE (busy algorithm)

# SOFTWARE data generator


1 Gb/s




1 Gb/s connection

# *The HARDWARE data generator*

# HARDWARE data generator


**PCI Express Stratix II GX Edition Development Kit**
**ALTERA**

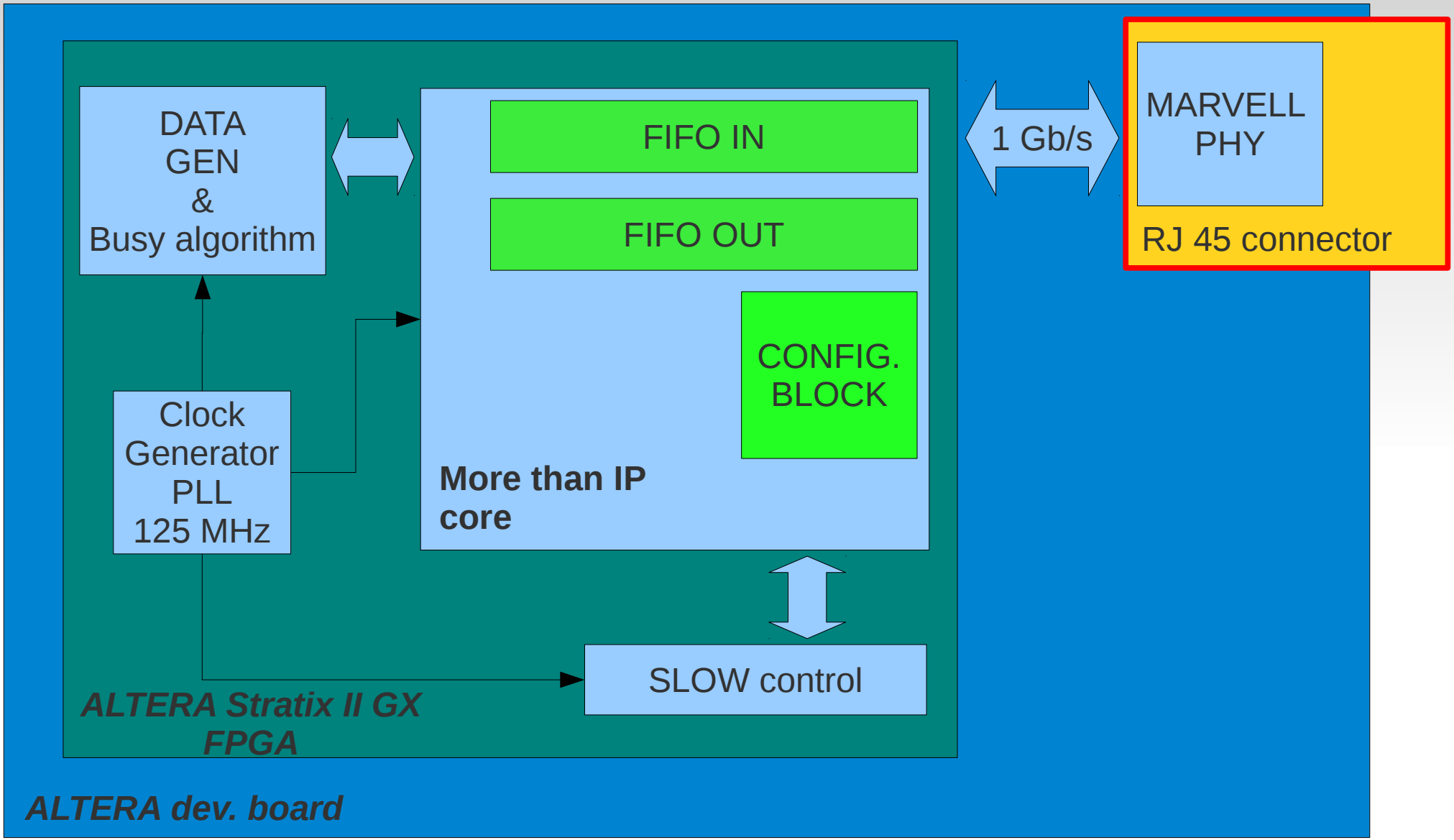
***Hardware***:
- PCI-Ex interface,
- SFP transceivers (SFP+ compatible),
- 1 Gb/s RJ45.

***VHDL***:
Used More than IP CORE to interface the code to the **Marvell 88E1111 GigE PHY Layer RJ45**

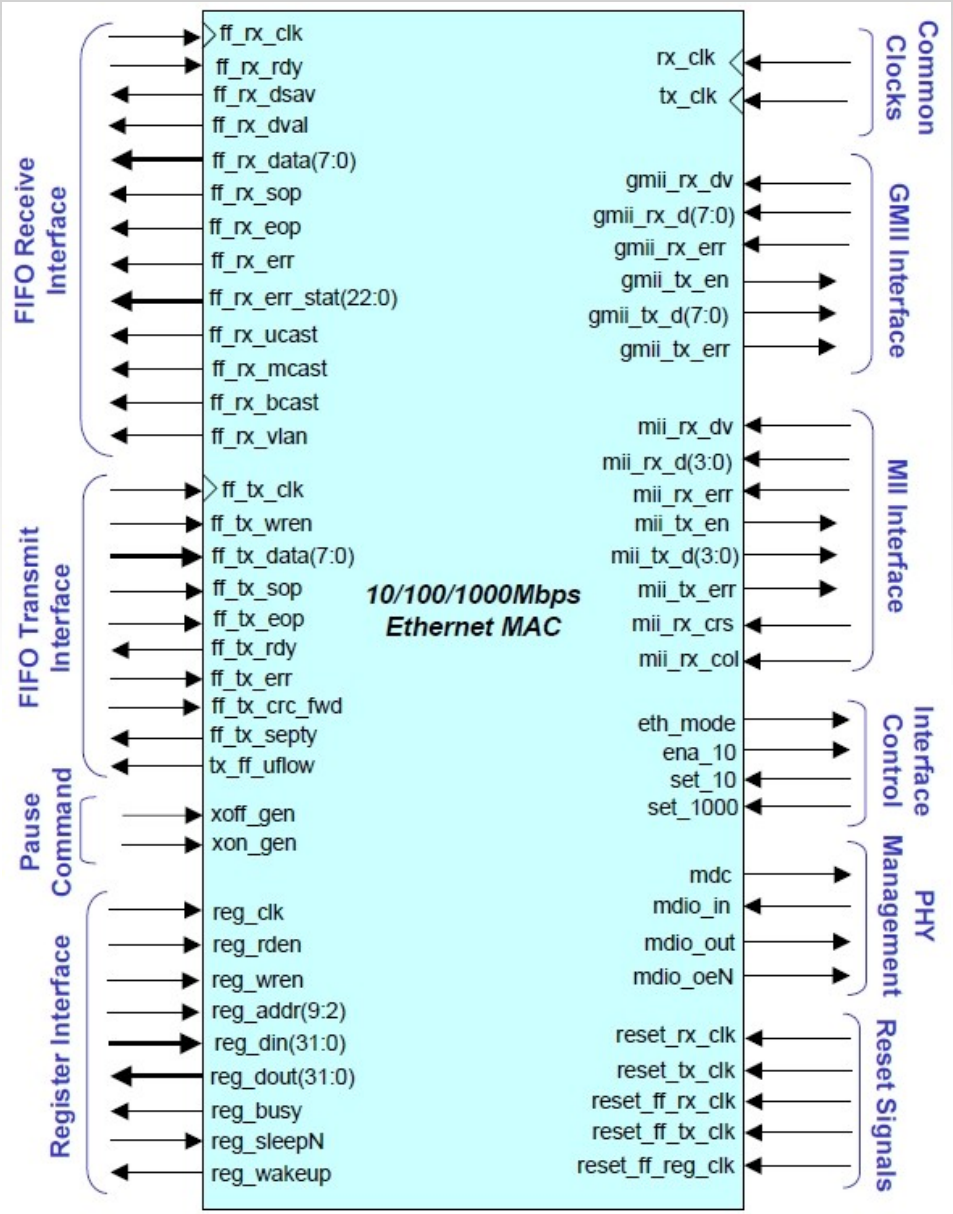
***Option of the core***:
- 8/32-Bit FIFO interface (IN/OUT),
- Clock 125 MHz.

***Protocol implemented*** *:*
- UDP,
- busy algorithm.

# FPGA details

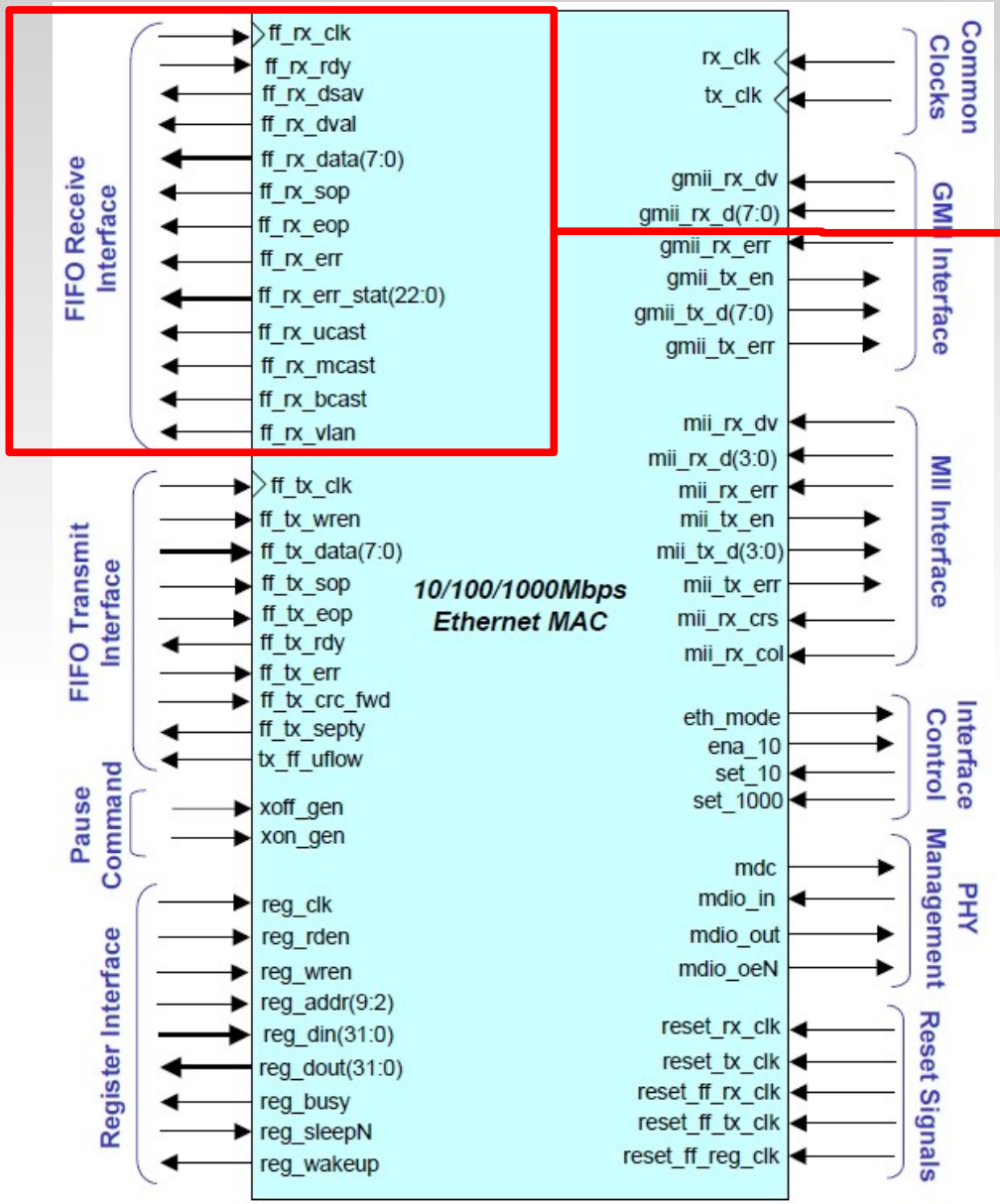


# IP CORE structure



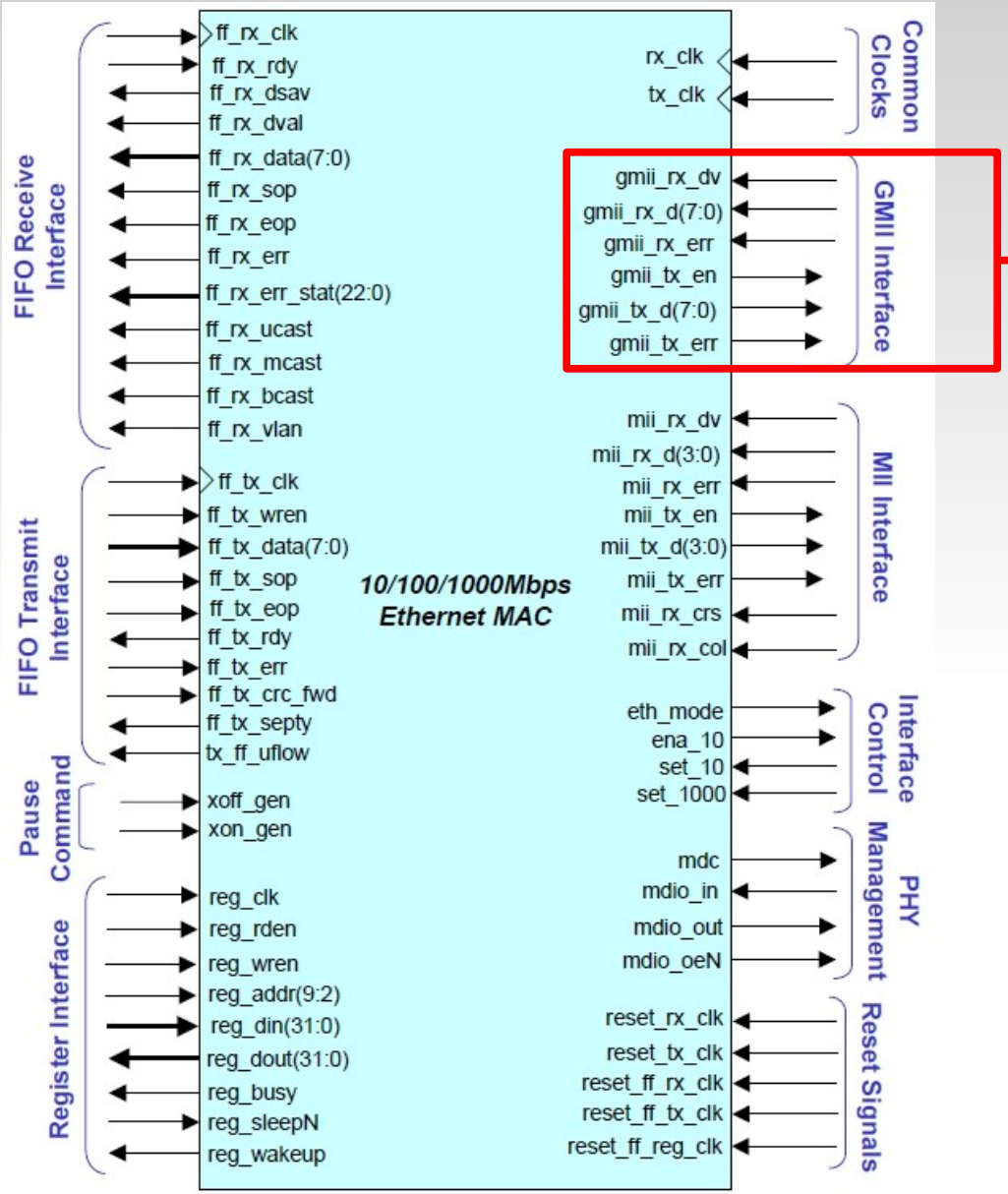

**MORE THAN IP CORE** :
1Gb/s link.
8 bit FIFO interface.
(the 32 bit doesn't differ too much)

# IP CORE structure



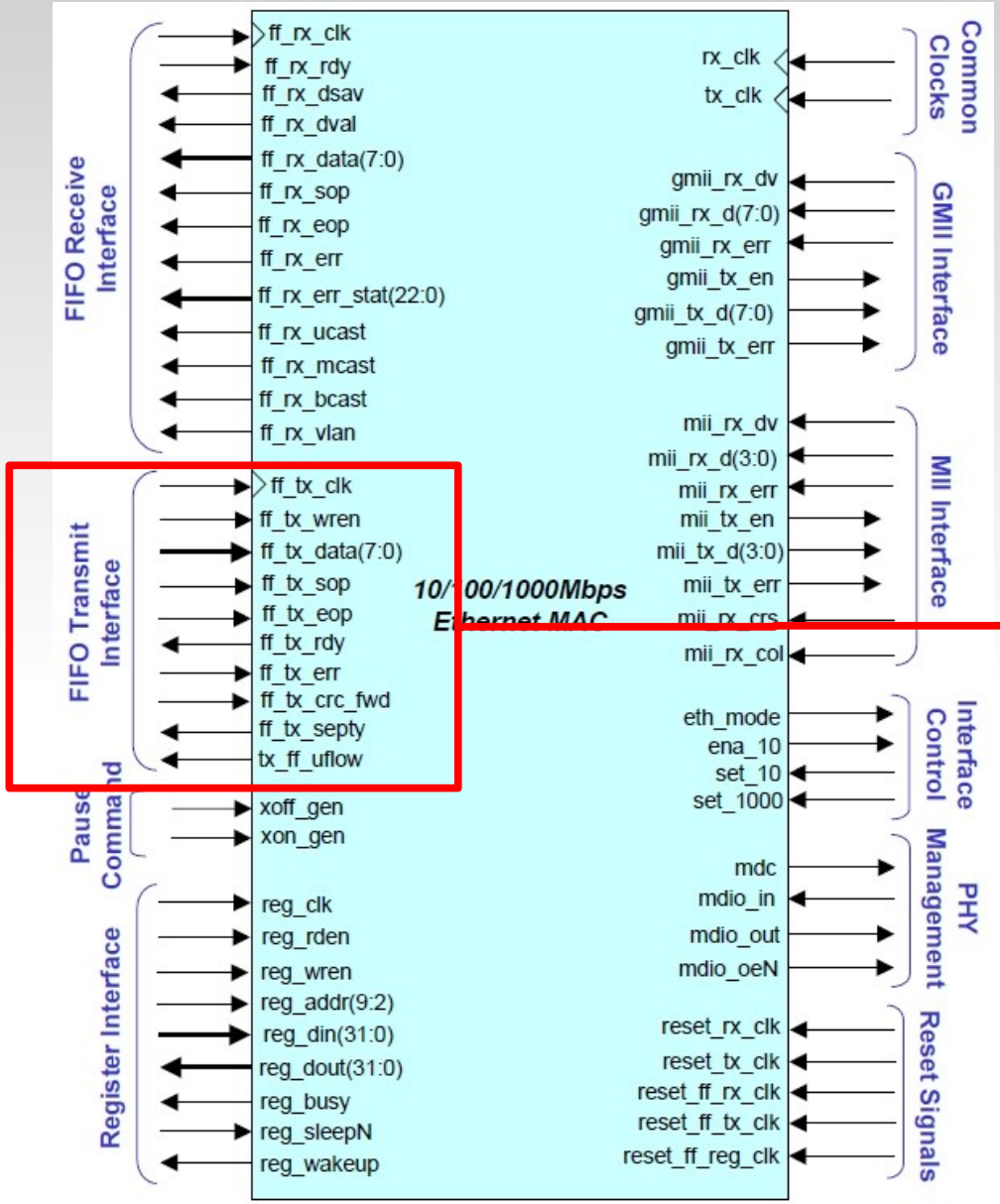

User VHDL code has to be connected to the ***FIFO Receive interface*** to read data coming from the Ethernet connector.

# IP CORE structure



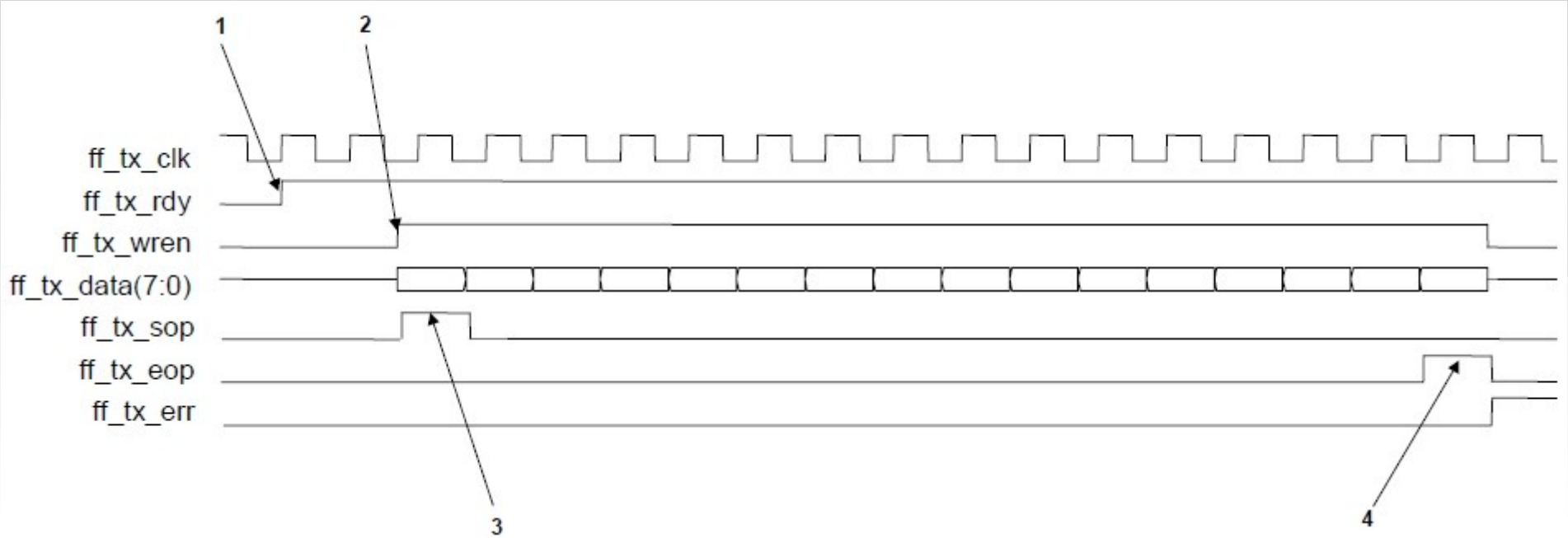

The **CORE** will interface itself to the **RJ45 connector** through this interface.

# IP CORE structure




User VHDL code has to be connected to the ***FIFO Transmit*** interface to send data through the Ethernet connector

# How to send data




These are the basic instructions to start to send data out:

**1.** When the **CORE** is ready to receive data asserts the signal ***ff_tx_rdy.***
**2.** The user application asserts ***ff_tx_wren*** when wants to start to write data.
**3.** The ***ff_tx_sop*** has to be raised when the first octet is wrote in the FIFO.
**4.** ***ff_tx_eop*** indicates the end of frame.

# HARDWARE data generator



The board was connected to a PC running DATE using 1GB/s link.
The communication is bi-directional.
DATE at the start of run sends the configuration to the board, after the board is configured DATE is ready to take data.

# *The TESTs*

# SETUP

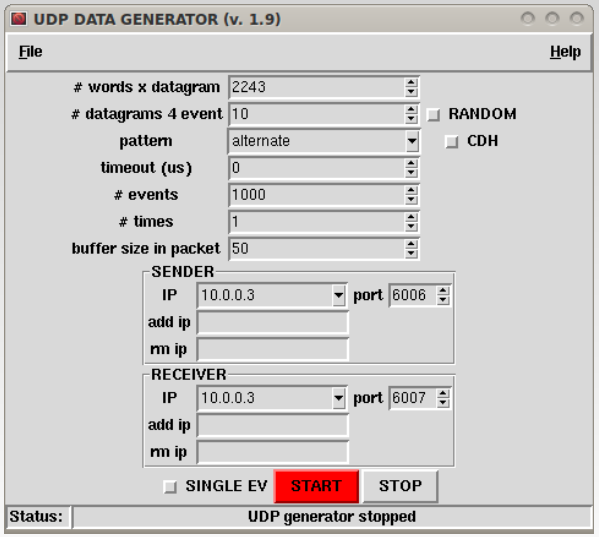
O.S.
CERN distribution SLC5 64 bit
Kernel 2.6.18-238.9.1.el5

PC (standard desktop machine)
CPU: Intel Core2 Duo 3.16GHz
RAM: 4 GB

INTEL 10 Gb AT PCI Express
Server adapter
PCI Express 16x compatible

# Test @ 1 Gb/s – SW data generator






| | |
|---|---|
| Current Trigger rate | 1435.400 |
| Average Trigger rate | 1302.167 |
| Number of sub-events | 39065 |
| Sub-event rate | 1435 |
| Sub-events recorded | 39064 |
| Sub-event recorded rate | 1434 |
| Bytes injected | 3508818436 |
| Byte injected rate | 128.927 MB/s |
| Bytes recorded | 3508548976 |
| Byte recorded rate | 128.873 MB/s |

Each event: 10 packets of 2200 words
Total event size: 88104 bytes (data + DATE header)
Acquisition rate: 1.4 Khz
Throughput: ~128 MB/s
(the maximum allowed by this link)

# Test @ 1 Gb/s – HW data generator


**1 Gb/s**

| | |
|---|---|
| Current Trigger rate | 1208.600 |
| Average Trigger rate | 770.088 |
| Number of sub-events | 26183 |
| Sub-event rate | 1208 |
| Sub-events recorded | 26182 |
| Sub-event recorded rate | 1208 |
| Bytes injected | 2306731236 |
| Byte injected rate | 106.477 MB/s |
| Bytes recorded | 2306466936 |
| Byte recorded rate | 106.442 MB/s |

***FIFO 8 bit interface***

Each event: 10 packets of 2200 words

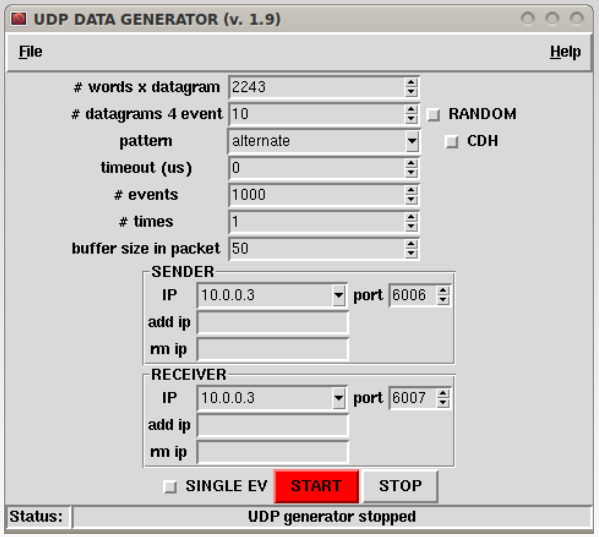
Total event size: 88104 bytes (data + DATE header)

Acquisition rate: 1.2Khz

Throughput: ~110 MB/s

# Test @ 10 Gb/s

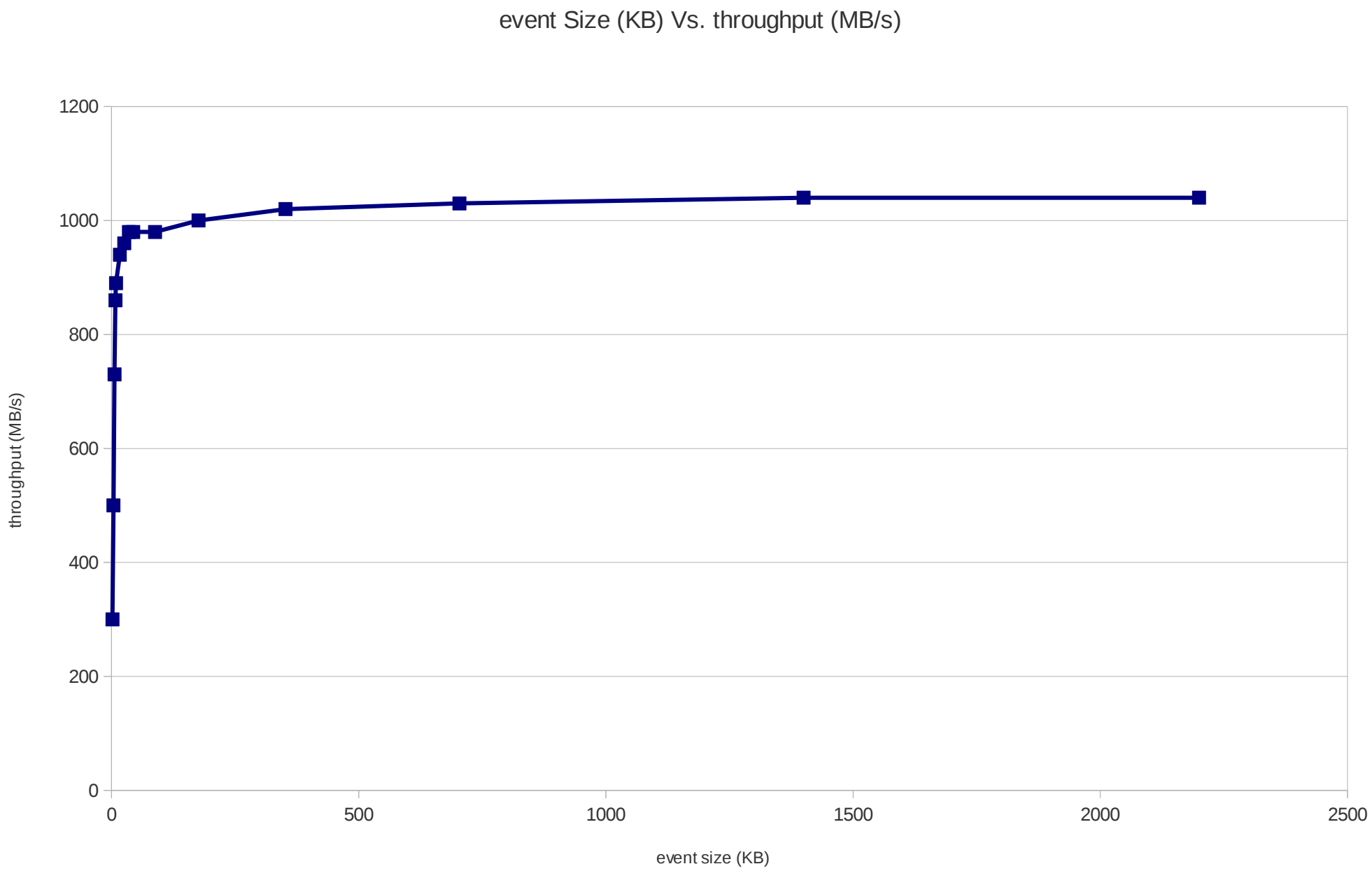





| | |
|---|---|
| Current Trigger rate | 11198.800 |
| Average Trigger rate | 10962.062 |
| Number of sub-events | 1589499 |
| Sub-event rate | 11198 |
| Sub-events recorded | 1589496 |
| Sub-event recorded rate | 11198 |
| Bytes injected | 140041220040 |
| Byte injected rate | 986.641 MB/s |
| Bytes recorded | 140040779520 |
| Byte recorded rate | 986.641 MB/s |

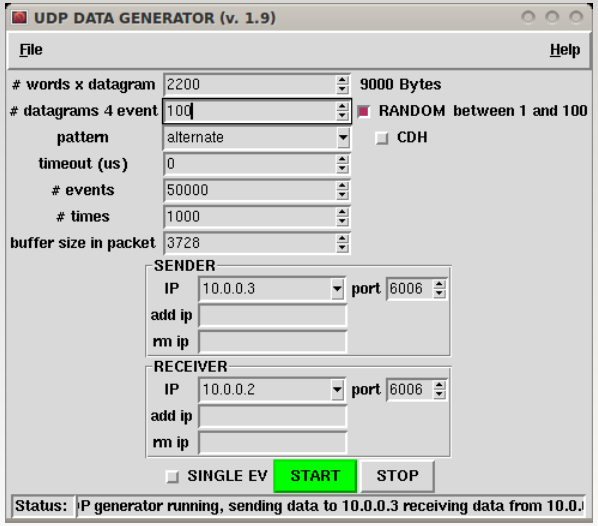
Each event: 10 packets of 2200 words
Total event size: 88104 bytes (data + DATE header)
Acquisition rate: 11 Khz
Throughput: ~990 MB/s

# EVENT SIZE Vs. THROUGHPUT



# Test @ 10 Gb/s






Each event: 1-100 packets of 2200 words
Total event size: random
Acquisition rate: 2.5 Khz
Throughput: ~1 GB/s

| | |
|---|---|
| Current Trigger rate | 2548.400 |
| Average Trigger rate | 10681.146 |
| Number of sub-events | 5596921 |
| Sub-event rate | 2548 |
| Sub-events recorded | 5596921 |
| Sub-event recorded rate | 2548 |
| Bytes injected | 504599281528 |
| Byte injected rate | 1.025 GB/s |
| Bytes recorded | 504598515720 |
| Byte recorded rate | 1.025 GB/s |

# Test with more than one client

## DATE Gb ethernet readout


| | |
|---|---|
| Current Trigger rate | 1372.400 |
| Average Trigger rate | 647.322 |
| Number of sub-events | 114576 |
| Sub-event rate | 1372 |
| Sub-events recorded | 114578 |
| ub-event recorded rate | 1372 |
| Bytes injected | 10291216456 |
| Byte injected rate | 123.268 MB/s |
| Bytes recorded | 10291216456 |
| Byte recorded rate | 123.268 MB/s |

1 Gb/s

1 Gb/s switch

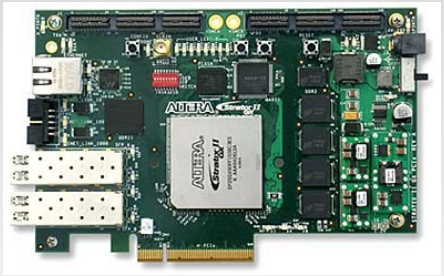
1 Gb/s

1 Gb/s

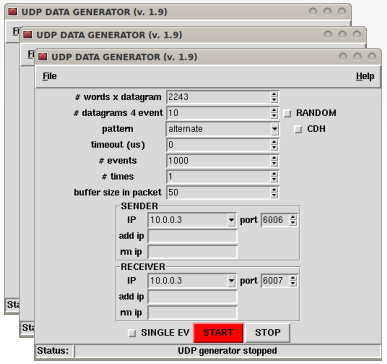
1 Gb/s

1 Gb/s

## HW



## 3 x SW

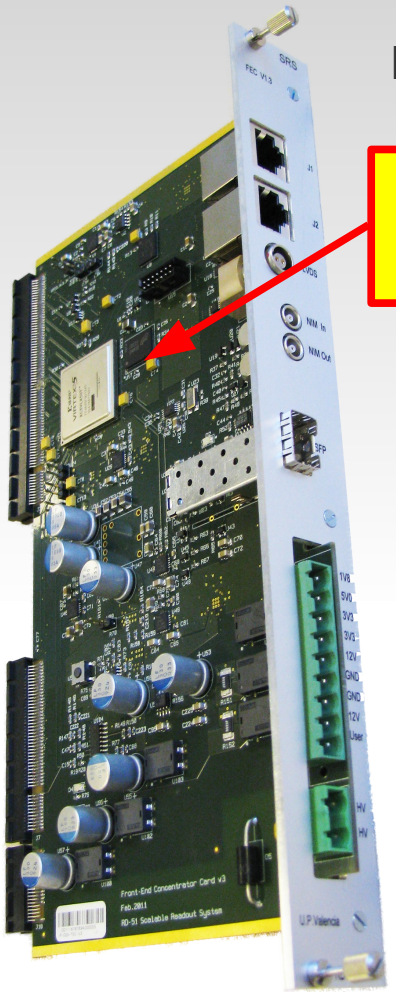



100 Mev => 1000M packets, without a single error bit.

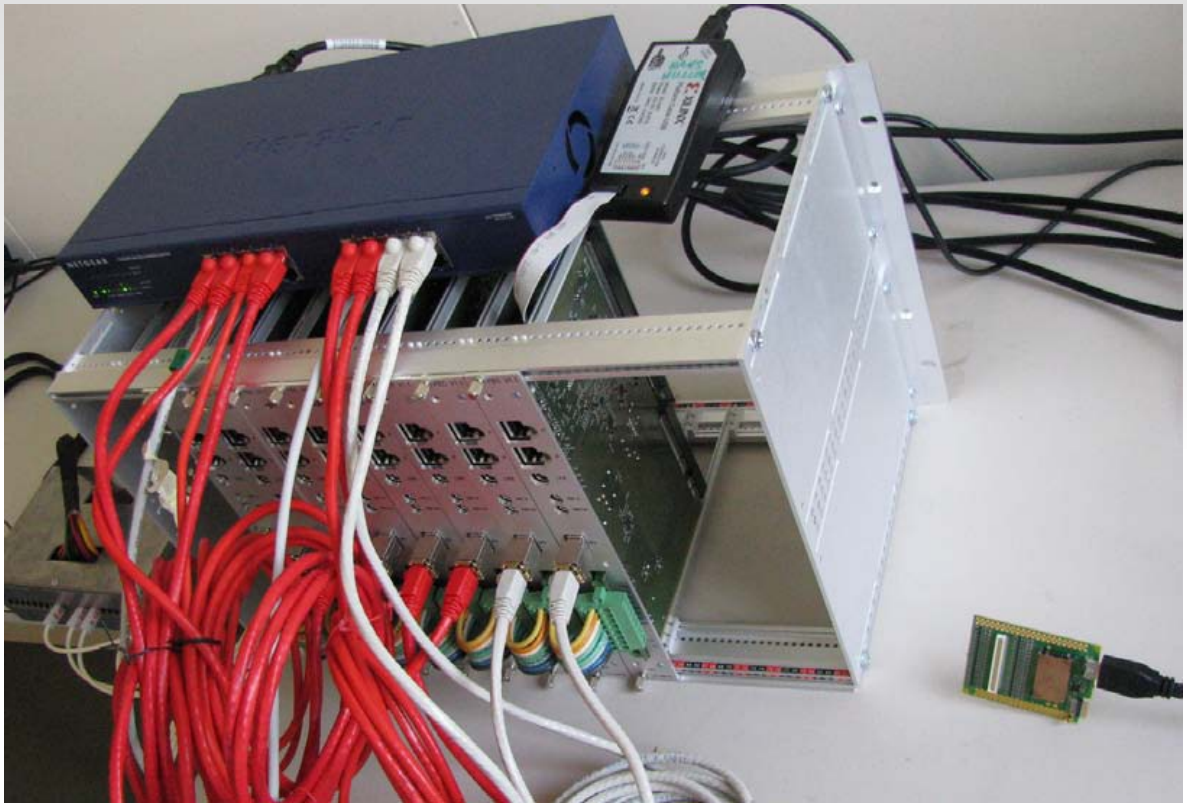
# Real application

# Using DATE and the UDP equipment in real application

**RD51** collaboration is currently using
**DATE and the UDP equipment** as **MAIN DAQ system**



**XILINX VIRTEX V / VI**



**8 readout boards + 1 SWITCH 1 Gb/s + DATE = readout system**

# Using DATE and the UDP equipment in real application

**RD51** collaboration is currently using
**DATE and the UDP equipment** as **MAIN DAQ system**


+


=

**portable DAQ system**

# Conclusions @ 1Gb/s

- Detectors with small throughput can build cheap readout system (PC + 1 Gb/s PCI-Ex card < 1000 USD).
- No time needed for “extra” development.
- Flexibility in the configuration.

# Conclusions @ 10 Gb/s

- Cost of the hardware is going to lower in the future.
  (10 Gb/s => 300/1000 USD)

- Prospect for higher throughput is very good:
  ***"Researchers create two 100 terabit per second optical connections***"
  http://www.engadget.com/2011/05/01/researchers-create-two-100-terabit-per-second-optical-connection/

- Good performance at 10 Gb/s.

# Thank you for your attention.

If you are interested in this development you can contact me at this address

***filippo.costa@cern.ch***

Or contact the **ALICE DAQ GROUP**:

***http://ph-dep-aid.web.cern.ch/ph-dep-aid/***