mmDAQ

(Muon Atlas MicroMegas Activity – ATLAS R&D)

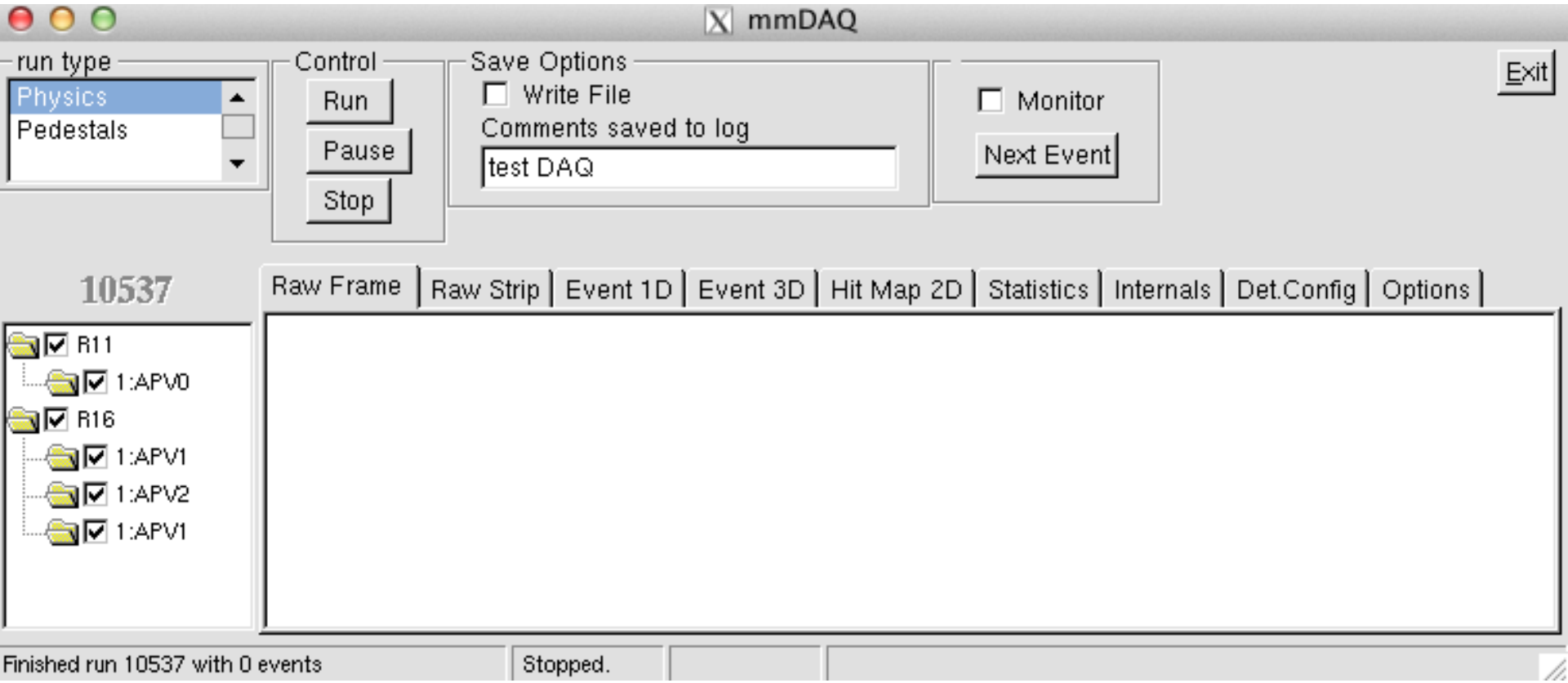
Marcin Byszewski

Outline

- **mmDAQ**
- mmDAQ v2
- Integration with ATLAS DAQ

mmDAQ

- Screenshot



mmDAQ

- C++ multithreaded
- Requires ROOT
- GUI online monitoring
- Online zero suppression
- Output to ROOT ntuples

Drawbacks:
Hard to configure

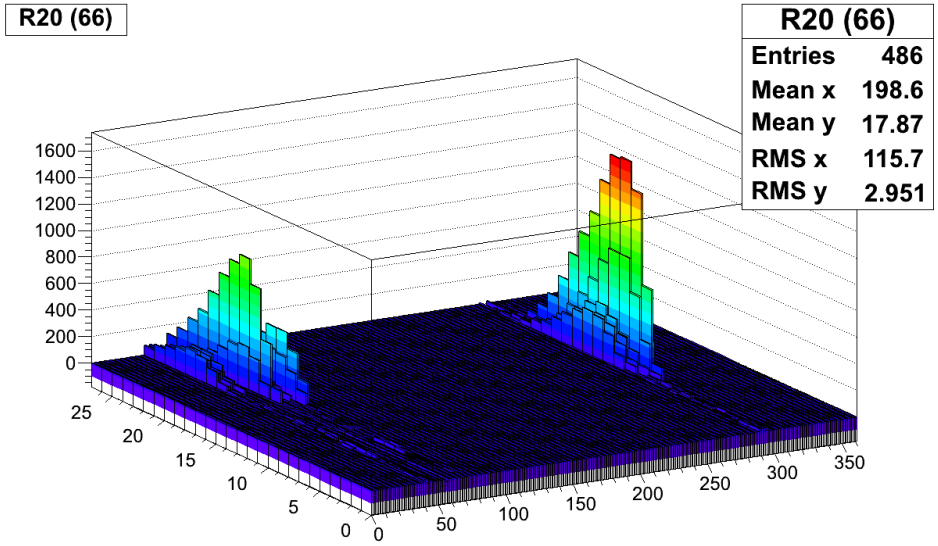
<u>Performance:</u>

- Desktop @Lab: 3 APV25 chips -> 400Hz
- Test Beam @CERN H6: 16 APV25 chips -> 200Hz
  - Intel(R) Core(TM)2 Duo CPU E8500 @ 3.16GHz
- Laptop @ATLAS: 9 APV25 chips:
  - Uninterrupted 8 months of data taking
  - Some runs with millions of triggers (internal)

mmDAQ data rates

Raw APV25 data rates: Zero suppressed

100 Hz **16 APV** **128 channels** **27 timebins** **2 bytes/word (16b)**	**5 mean occupancy** -> data reduction	0.04
6912 bytes/chip	270	
110592 bytes/FEC	4320	
11059200 bytes/s	432000	
10800 kbytes/s	422	
10.5 Mbytes/s	0.41	



Outline

- Desktop mmDAQ
- **mmDAQ v2**
- FEC zero suppression
- Integration with ATLAS DAQ

Work in progress: mmDAQ2

- Work in progress:
  - Most of the back-bone ready
  - Work on UI and data output

Characteristics:

- Single chips reading different Chambers/Planes/Readout strips (MBT0)
- Multiple FECs with different front-end HW side-by-side
  - ✓ Raw APV25
  - ✓ Zero-suppressed APV25
  - ✓ BNL chip with Arizona card
- Easy configuration
- *At this stage – still open for discussions/requests*

mmDAQ2

Modular:
adding new HW requires sub classing of 2 classes
Example for BNL chip
- Event class (defines data decoding) + Channel class (defines data format)

```
namespace mmdaq {

   class CUDPFrame;

   class CBnlSrsEvent : public CSrsEvent
   {
      static int SrsBnlNumberChannels;
      static size_t SrsBnlHeaderSize;
      static size_t SrsBnlChannelDataSize;

      uint32_t m_fec_idnumber;  ///< unique fec id
      unsigned m_bad_event_error_code;
      void decode_data(const std::vector<int16_t>& data16);
      void process_event();
   public:
      CBnlSrsEvent(const CUDPFrame* udpframe);
      virtual ~CBnlSrsEvent();
      virtual void write();
   };

}
```

```
namespace mmdaq {

   class SrsChannelId;

   class CBnlSrsChannel : public CSrsChannel
   {

   public:
      CBnlSrsChannel(const SrsChannelId& channel_id, const std::vector<in
      CBnlSrsChannel(const SrsChannelId& channel_id,
                     std::vector<int16_t>::const_iterator first,
                     std::vector<int16_t>::const_iterator last);

      virtual ~CBnlSrsChannel();
   private:

      std::vector<int16_t> m_data;

      virtual void process_channel(const CDaq* thedaq); ///< called by CE

   };


}
```

Outline

- Desktop mmDAQ
- mmDAQ v2
- **FEC zero suppression**
- Integration with ATLAS DAQ

FEC zero suppression

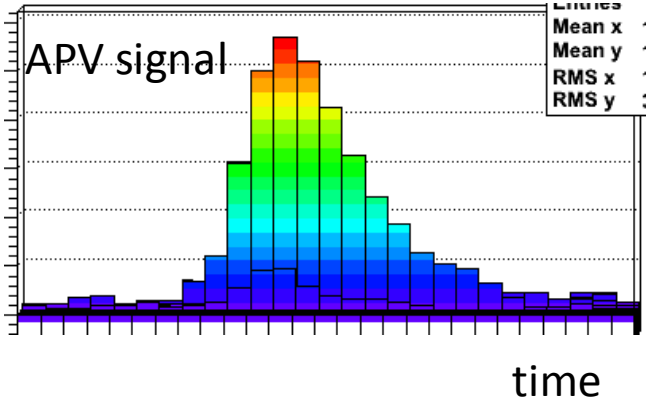
- APV25
  - analogue, 128 channels * number of time samples

- FEC FPGA firmware module
  - by R. Giordano and S. Martoiu
  - Possibility to bypass for a single APV25 chip (configure stages)
  - Configuration (pedestal measurement) must be run by user

  - Passed first tests with the mmDAQ
  - Remotely controlled by DCS (by G. Iakovidis)
  - At debugging stage
  - Max rate not tested

FEC zero suppression

- Based od mmDAQ ZS
- Suppression cut:



$Q_{sum} / N_{time\_bins} < Ped_{stddev} * A$

Q_{sum} = sum of charge

N_{time_bins} = number of time samples

Ped_{stddev} = Pedestal std.dev.

A = user defined factor (about 0.8 in mmDAQ)

(see Sorin's talk for implementation details)

FEC zero suppression

- Data Format:

```
typedef char BYTE;              // 8-bit word
typedef unsigned int WORD32;    // 32-bit word
typedef unsigned short int WORD16; // 16-bit word

struct APV_HEADER
{
BYTE        APV_ID;    // APV Identifier number on the FEC card (0 to 15)
BYTE        N_CHANNELS; // the number of channels which will be following the header
BYTE        N_SAMPLES;  // the number of samples per channel
BYTE        ZS_ERROR;   // Error code from the Zero Suppression Block, meaning have to be defined
WORD32      RESERVED;   // 32 bits reserved for future use
};

struct CHAN_INFO
{
BYTE            RESERVED;          // 8-bits reserved for future use
BYTE        CHAN_ID;           // Channel identifier, 0 – 127, future: 128 common mode , 129 for error codes
WORD16      CHANDATA[N_SAMPLES]; // 16 bit words, actual data will be 13-bits wide
};
```

Outline

- Desktop mmDAQ
- mmDAQ v2
- **Integration with ATLAS DAQ**

Micromegas in ATLAS @ winter shutdown

- Plan to install during this Xmas shutdown:
  - 8x9 cm MBT0 chamber on the edge of MBTS
  - 1.2x0.5m^2 test chamber (4 active planes, X+U strips)

- HW
  - All equipped with APV25 chips (4 + 4*10 = 44 chips)
  - 25 m HDMI cables (to be tested)
  - 4-5 FECs
  - DTC links to 1 SRU unit
  - Optical S-Link fibre to a ROS in USA15 cavern

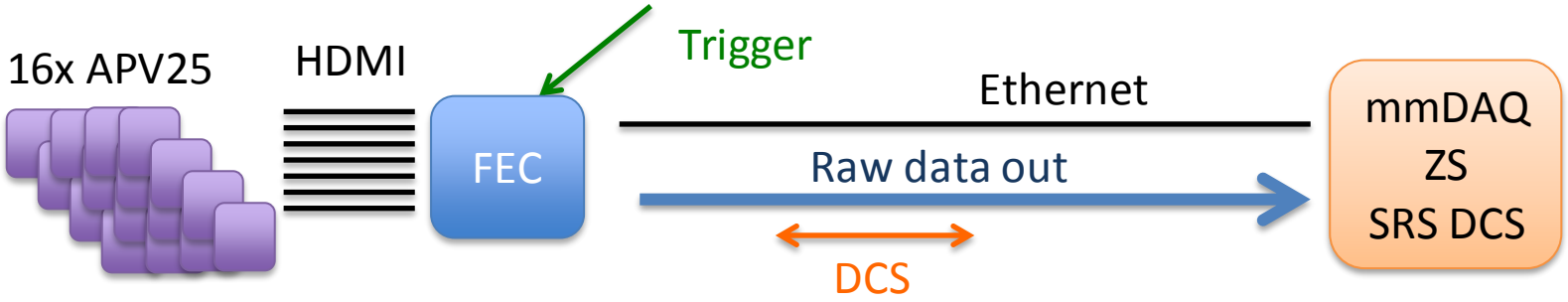
Micromegas in ATLAS @ winter shutdown

- All equipped with APV25 chips (4 + 4*10 = 44 chips)
- 25 m HDMI cables (to be tested)
- 4-5 FECs
- DTC links to 1 SRU unit
- Optical S-Link fibre to a ROS in USA15 cavern

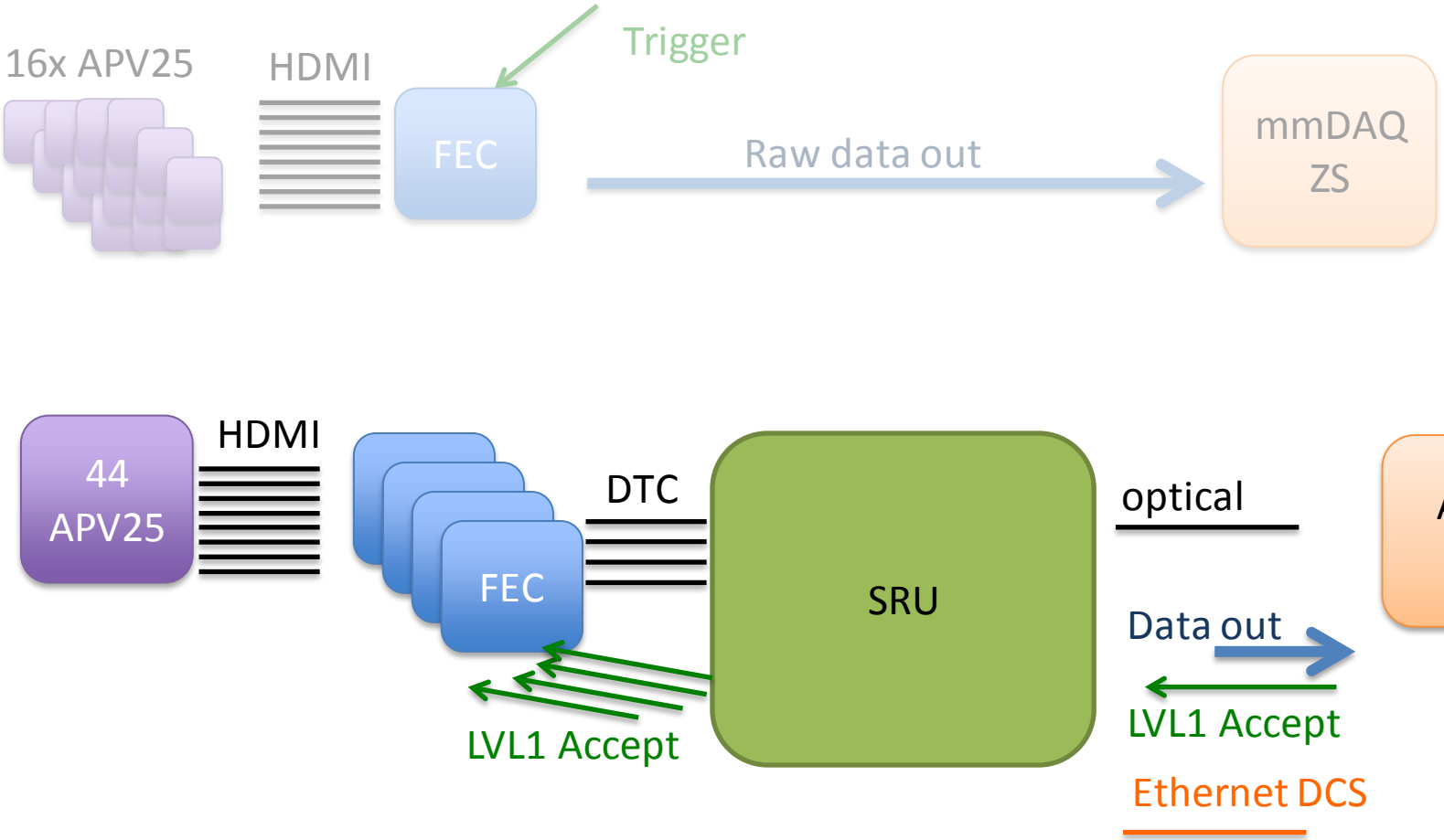
- MAMMA's Lab Setup
  - 16 APV chips
  - 1 FEC (raw) (trigger in, raw data to UDP packets)
  - Ethernet to mmDAQ PC (zero suppression, storage)
  - SRS DCS

- 2011 Xmas Installation
  - 44APV chips
  - 16 FEC (ZS) (trigger in, ZS, data to DTC links)
  - DTC to SRU
  - to mmDAQ PC (zero suppression, storage)

Current MAMMA's Lab setup



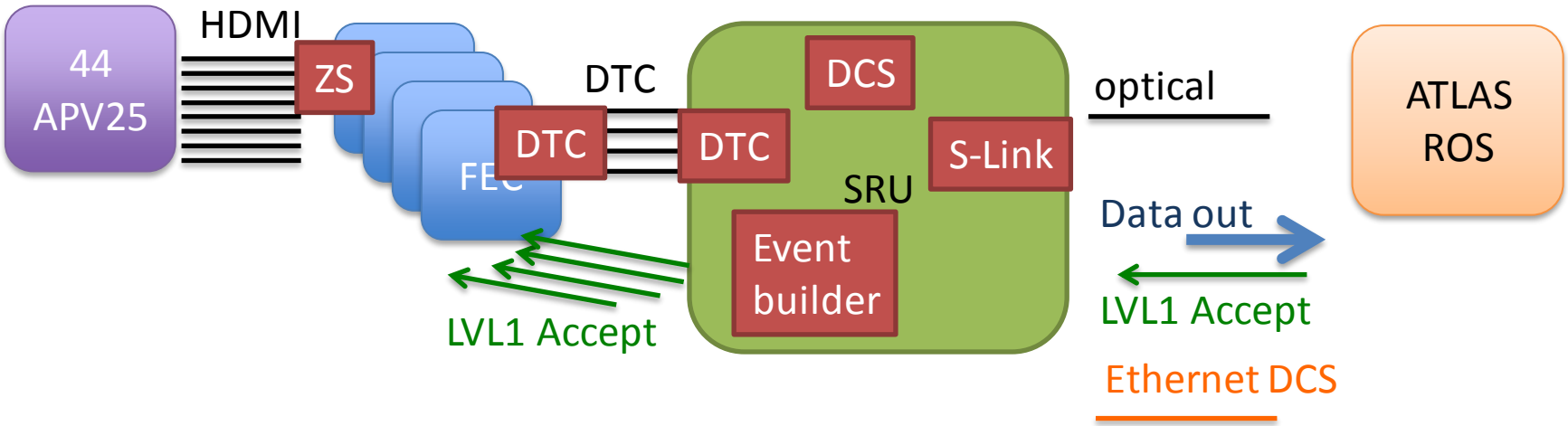
Integration into ATLAS DAQ



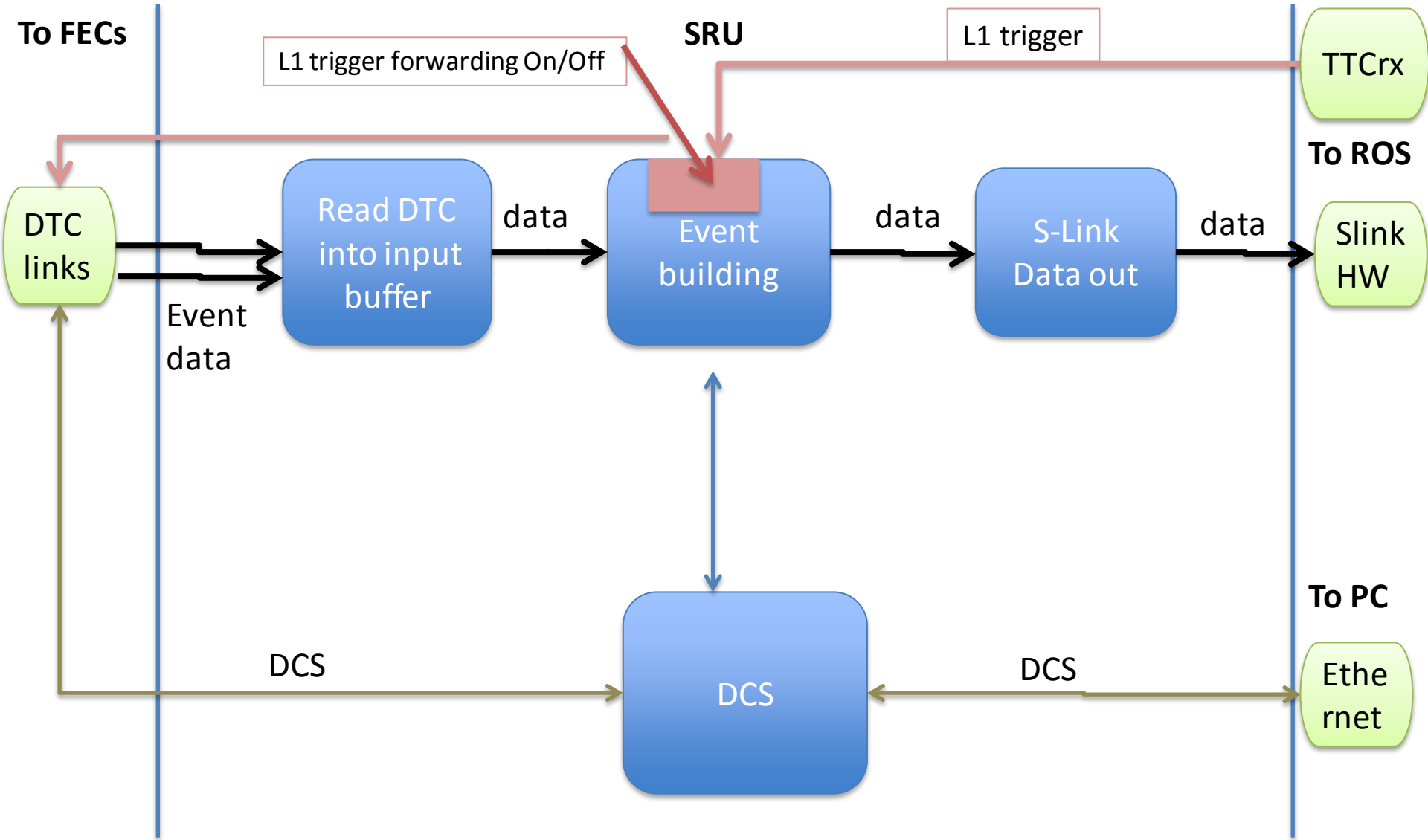
Integration: Challenge 1

Missing FPGA firmware

Enormous software development effort on a short time scale



SRU FW blocks for ATLAS DAQ

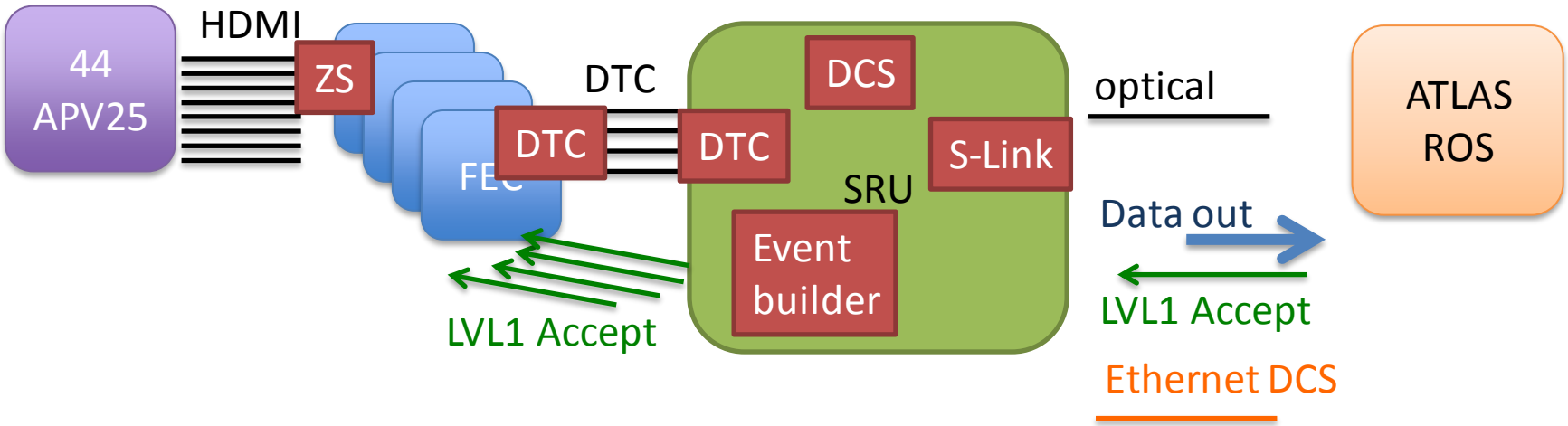


Integration: Challenge 2

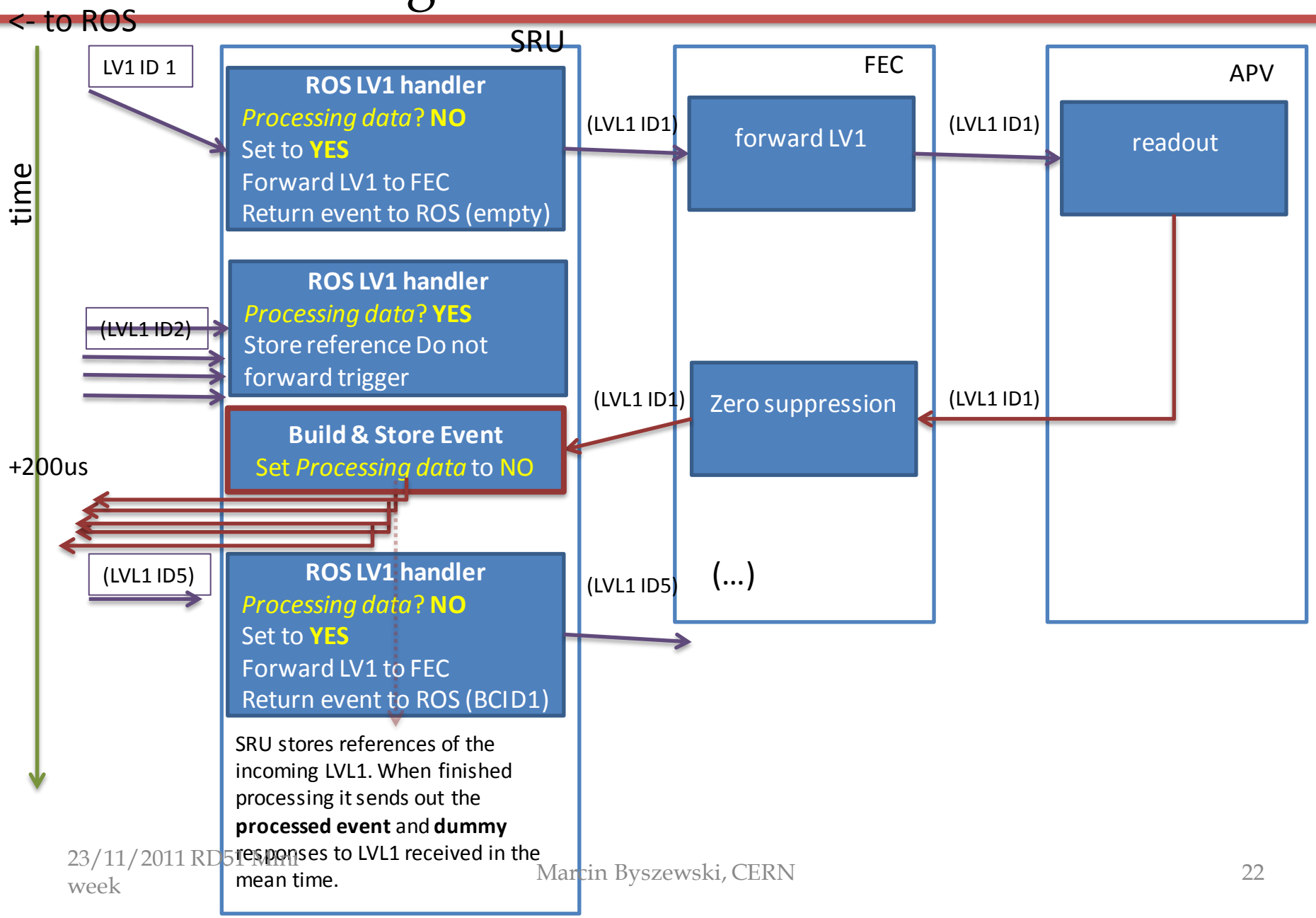
← Synchronous data flow @75kHz (13us)

APV ->FEC readout timing
APSP 4x70clocks = 7us / timebin
output 140bits*clock = 3500 ns /timebin
16 time bins -> 220us

Ok for 1 time sample, but we want to send 16 time samples.
→Do not provide data for every LVL1 Accept



LVL1 handling scheme



Summary

- mmDAQ makes users happy
- mmDAQ2 more versatile version is being developed
- Enormous effort in progress to prepare for ATLAS DAQ integration this winter

- Collaborators welcomed !

Backup slides