Update on SRS experience

(Muon Atlas MicroMegas Activity – ATLAS R&D)

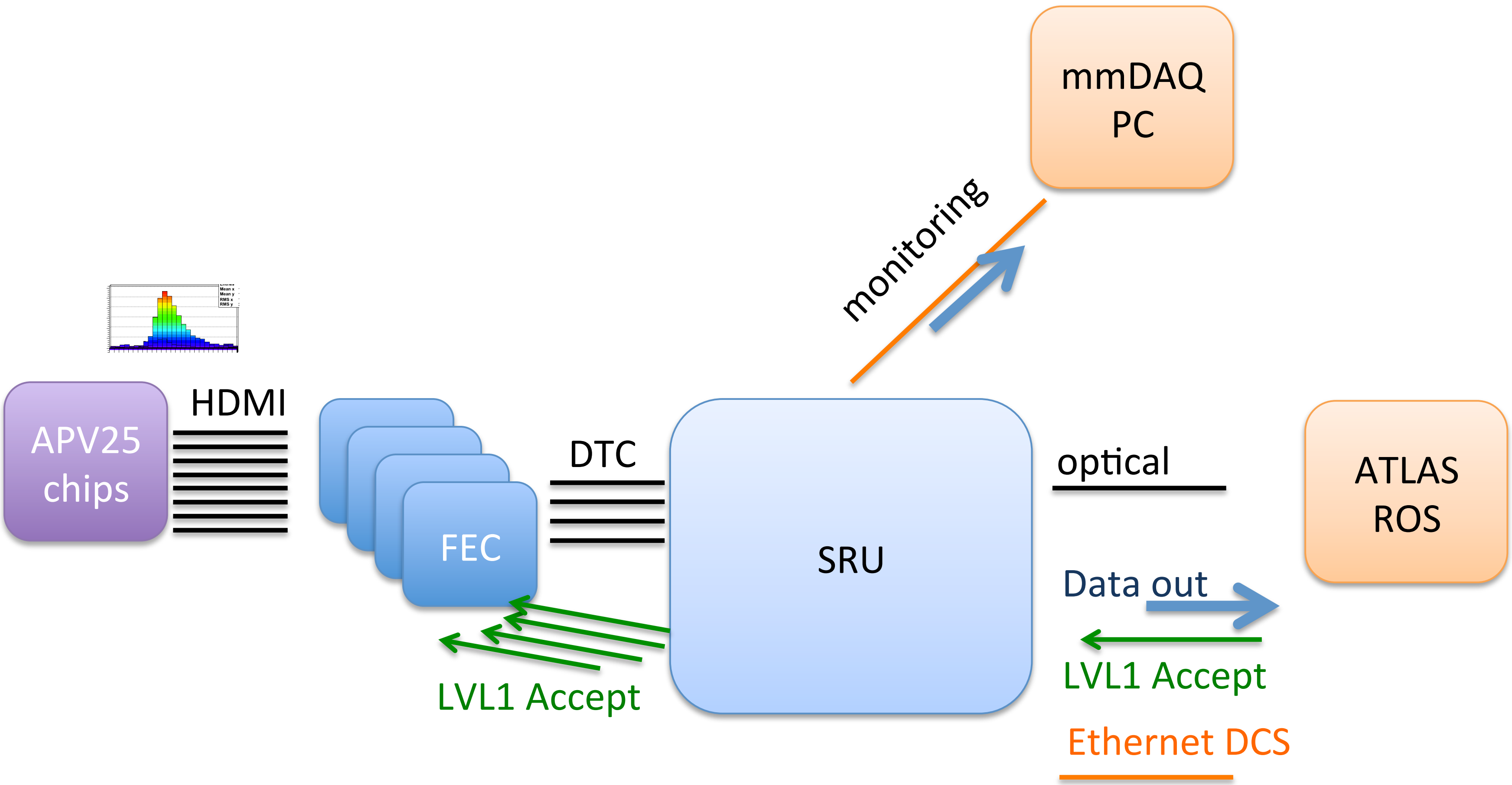
Marcin Byszewski

Micromegas in ATLAS @ 2012 winter shutdown

- Test chambers installed earlier this month:
  - MBTS (between inner tracker and the calorimeter)
  - Small wheel, side A

- SRS HW
  - All equipped with APV25 chips
  - 30 m HDMI cables
  - 3 FECs
  - DTC links to 1 SRU unit
  - Optical S-Link fibre to a ROS in USA15 cavern (100m)

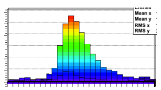
Integration into ATLAS DAQ

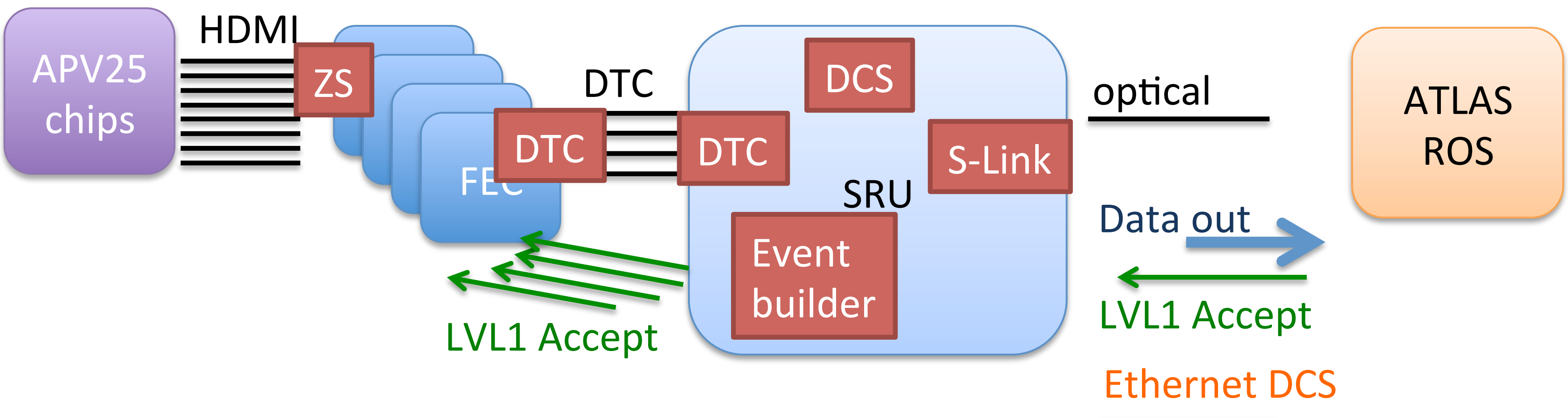


FPGA Firmware

FPGA firmware

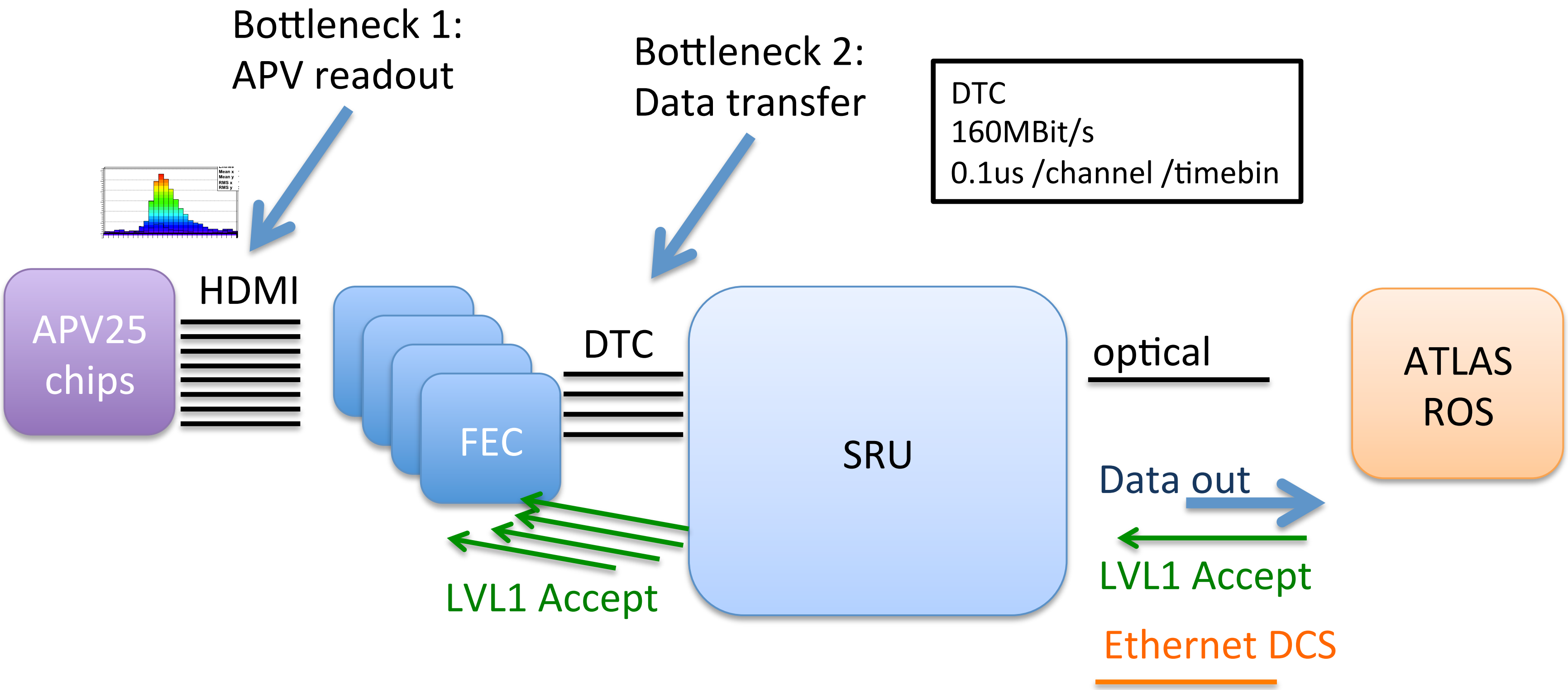
Raffaele Giordano
Vincenzo Izzo
Sorin Martoiu
Andre Zibell





SRS readout

APV ->FEC readout timing
APSP 4x70clocks = 7us / timebin
output 140bits*clock = 3.5 ns /timebin
1 time bins -> 10 us



SRS readout

Serve one in a few triggers

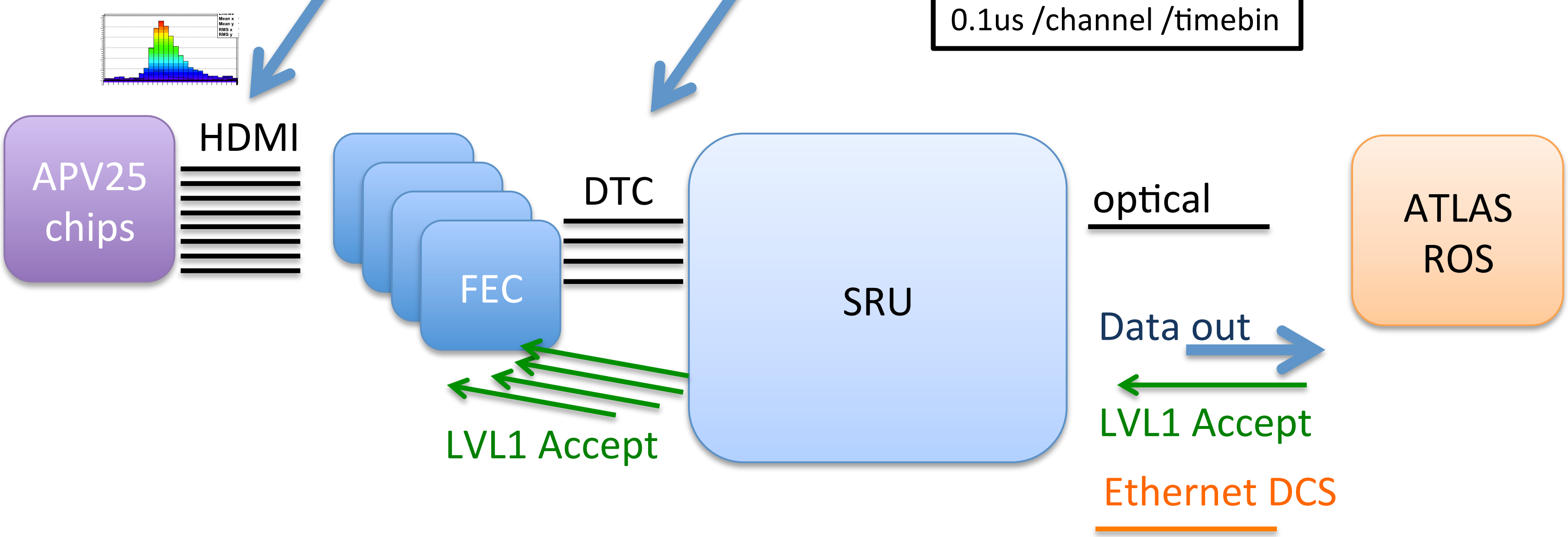
Bottleneck 1:
APV readout

Zero suppression
Peak finding

Bottleneck 2:
Data transfer

APV ->FEC readout timing
APSP 4x70clocks = 7us / timebin
output 140bits*clock = 3.5 ns /timebin
1 time bins -> 10 us

DTC
160MBit/s
0.1us /channel /timebin



FEC zero suppression

Two modes:

①

- Remove dark channels (# time bins)

②

- Remove dark channels
- Peak finding (q, time bin number)

- See S.Martoiu's talk

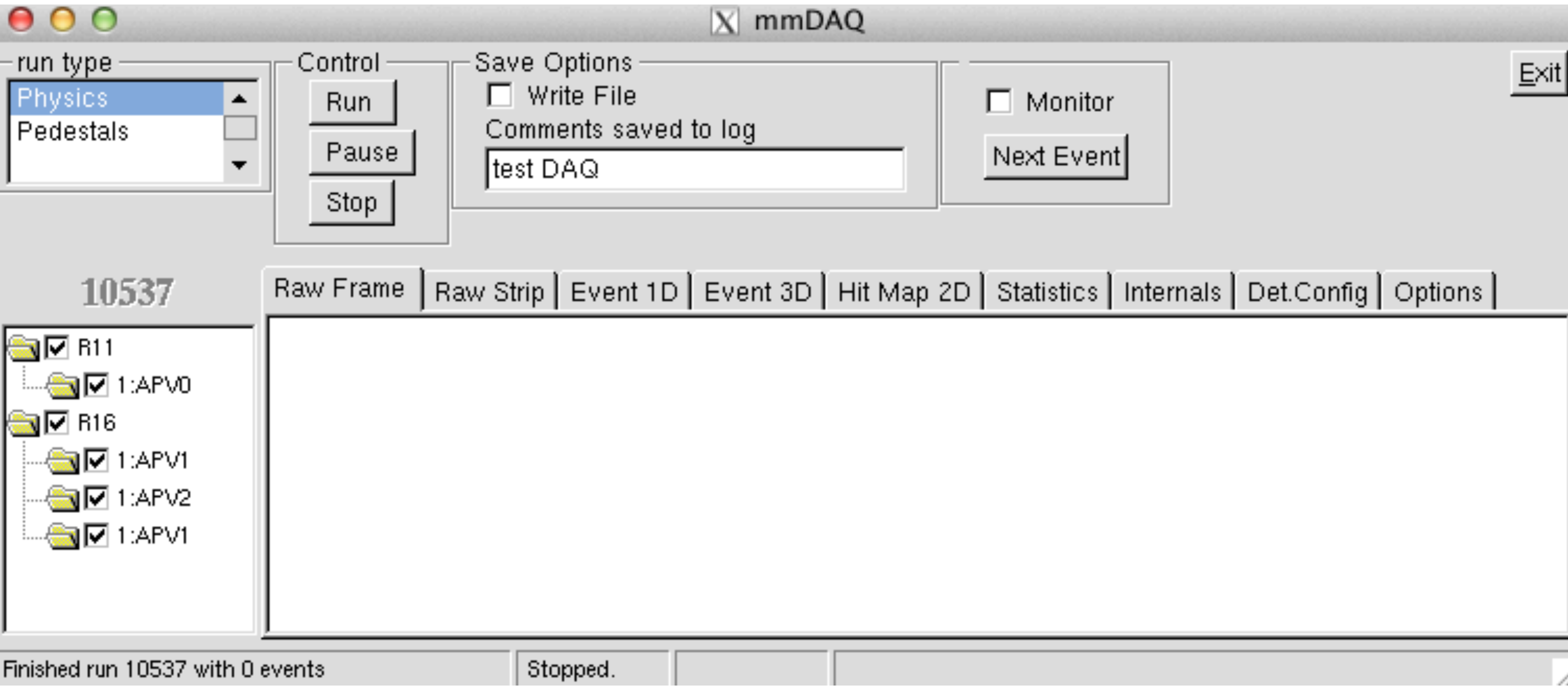
FEC zero suppression

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

  - Remotely controlled by DCS (by G. Iakovidis)

mmDAQ (see RD51 talk 14/4/2011)

- C++ multithreaded
- GUI online monitoring
- Online zero suppression
- Output to ROOT ntuples
- Simple data quality monitoring



mmDAQ

Input:

- DAQ parameters (IP, write path)
- Detector configuration (chambers, APV chips)
- Mapping (channel# → strip#)

Output:

- Root file: flat tree
- Event#, time (receive time)
- Address: (FEC, chip#, strip#)
- Data: ADC vector (vs. time bin numbers)

1/3 mmDAQ Config file

```
####################################################################
# network parameters of the DAQ
####################################################################
mmdaq.DAQIPAddress: 10.0.0.3
mmdaq.DAQIPPort: 6006

####################################################################
#data saving
####################################################################
mmdaq.LogFile:                 /data/2011/Munich_Nov/Munich_2011Nov.log
mmdaq.ChannelMapFile:          /data/2011/Munich_Nov/config/Munich.map
mmdaq.CrosstalkCorrectionFile: /data/2011/Munich_Nov/config/crosstalk.map
# save data to this directory:
mmdaq.WriteDataPath:           /data/2011/Munich_Nov/APV25/
# default value for zero suppression factor
mmdaq.ZeroTresholdFactor:      20.0
```

Log – a text file:
"7195", "05-Nov-2011", "16:55 ", "50483", "physics",
"Random trigger rate 200 Hz, HV=300/520, 100 nA, 20 cm boron, Dist +100 mm, ZSF=0.8"

2/3 mmDAQ SRS configuration

```
#################################################################
# FEC IP address, connection to APV Chips, and their id numbers (0-15)
#################################################################
mmdaq.FECs: FEC1
mmdaq.FEC1.Chips:  APV0, APV1, APV2, APV4, APV5
mmdaq.FEC1.IPAddress: 10.0.0.2

#################################################################
# translate names like APV0 to their id numbers
# usually you an leave this as it is
#################################################################
mmdaq.FEC1.Chips.APV0.Id: 0 (Cable 1, Master)
mmdaq.FEC1.Chips.APV1.Id: 1 (Cable 1, Slave)
mmdaq.FEC1.Chips.APV2.Id: 2
mmdaq.FEC1.Chips.APV3.Id: 3
```

3/3 mmDAQ chamber configuration

```
#################################################################
# CHAMBERS, list of names, their connection to APV chips
#################################################################
mmdaq.Chambers: R20, R21, MBT0_1, MBT0_2

mmdaq.Chamber.R21.Strips: X, Y

mmdaq.Chamber.R21.Strips.X.Angle: 90.0
mmdaq.Chamber.R21.Strips.X.Chips: APV4, APV5
mmdaq.Chamber.R21.Strips.X.Pitch: 0.5
mmdaq.Chamber.R21.Strips.X.Min: 1
mmdaq.Chamber.R21.Strips.X.Max: 180

mmdaq.Chamber.R21.Strips.Y.Angle: 0.0
mmdaq.Chamber.R21.Strips.Y.Chips: APV3
mmdaq.Chamber.R21.Strips.Y.Pitch: 1.0
mmdaq.Chamber.R21.Strips.Y.Min: 1
mmdaq.Chamber.R21.Strips.Y.Max: 90
```

Map file APV channel -> strip

#neutrobeam Munich 2011 November
#
#testbeam

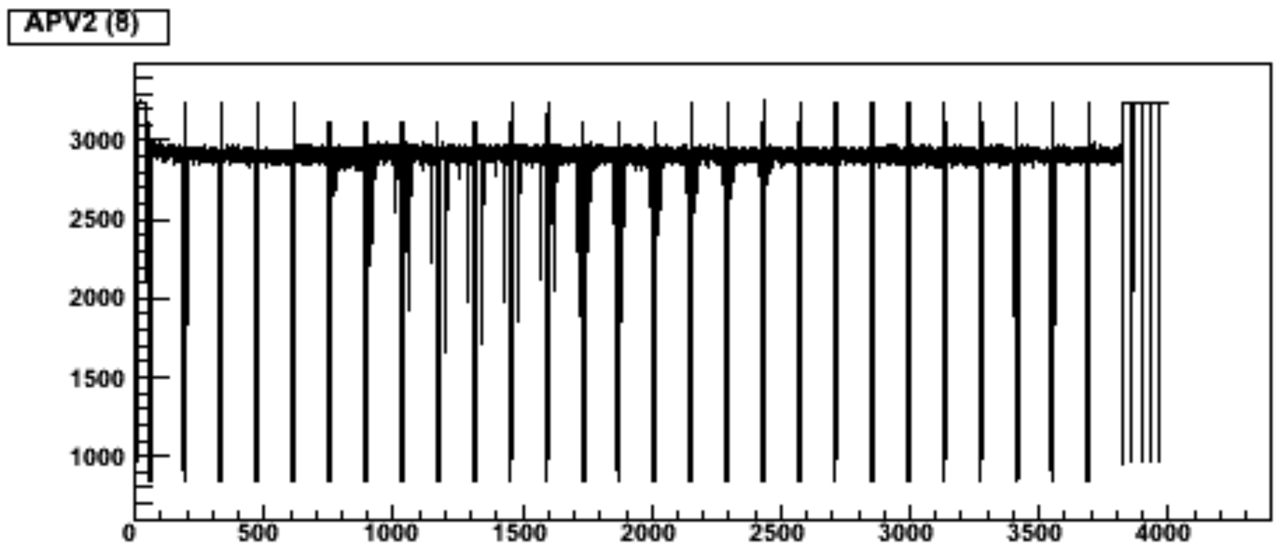
# dir	Y	Y	Y	U	X
APVid:	**-1**	**-1**	**-1**	**0**	**1**
Chamber:	**R18**	**R18**	**R18**	**R19G**	**R19G**
0	33	148	265	-1	4
1	32	147	264	-1	1
2	34	149	266	6	2
3	31	146	263	1	3
4	35	150	267	8	8
5	30	145	262	2	5
6	36	151	268	10	6
7	29	144	261	3	7
8	37	152	269	12	12
9	28	143	260	4	9
10	38	153	270	14	10
11	27	142	259	5	11
12	39	154	271	16	16
13	26	141	258	7	13
14	40	155	272	18	14
15	25	140	257	9	15
16	41	156	273	20	20
17	24	139	256	11	17
18	42	157	274	22	18

APV channel

mmDAQ

(see RD51 talk 14/4/2011)

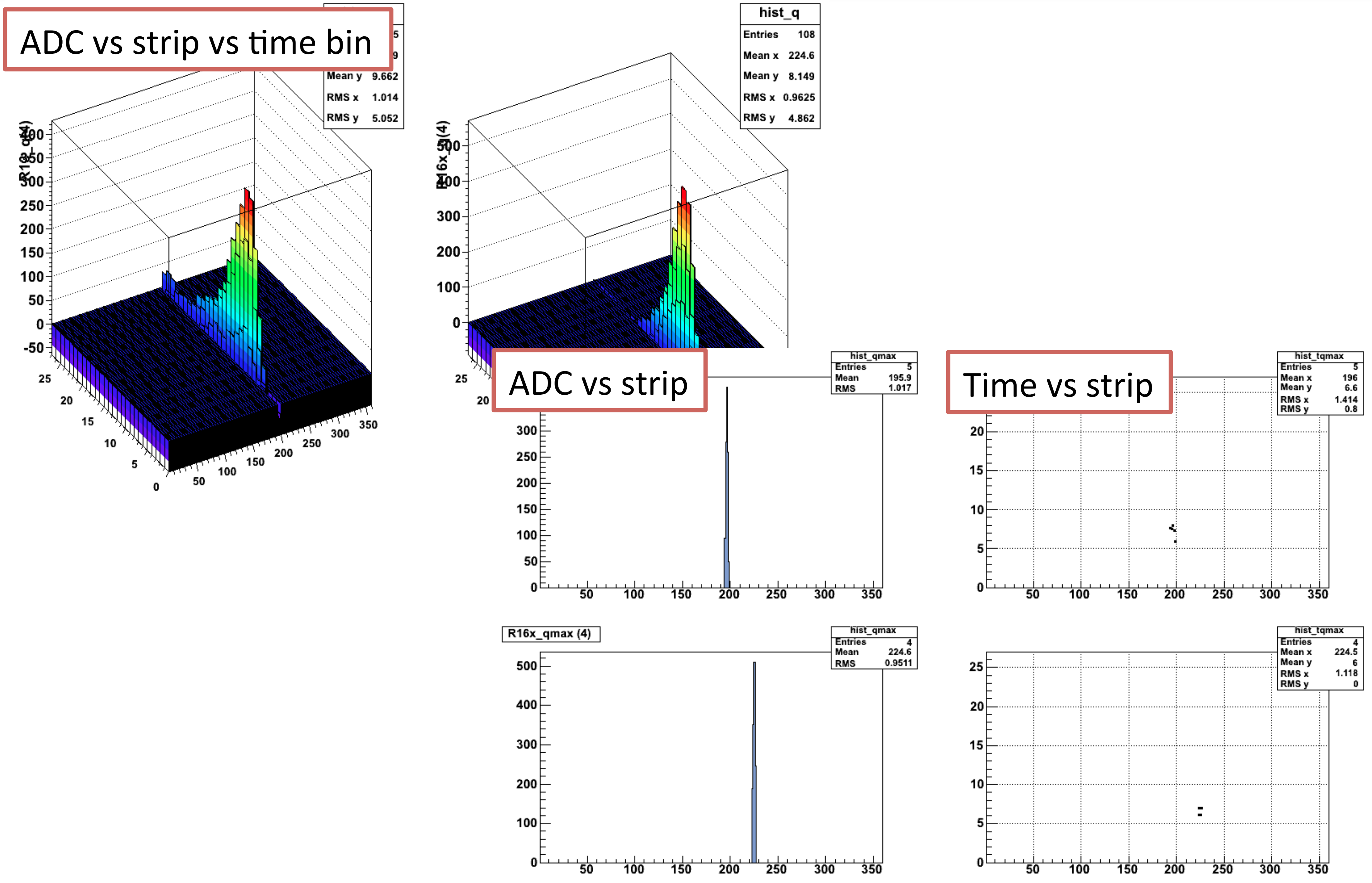
Raw UDP frame



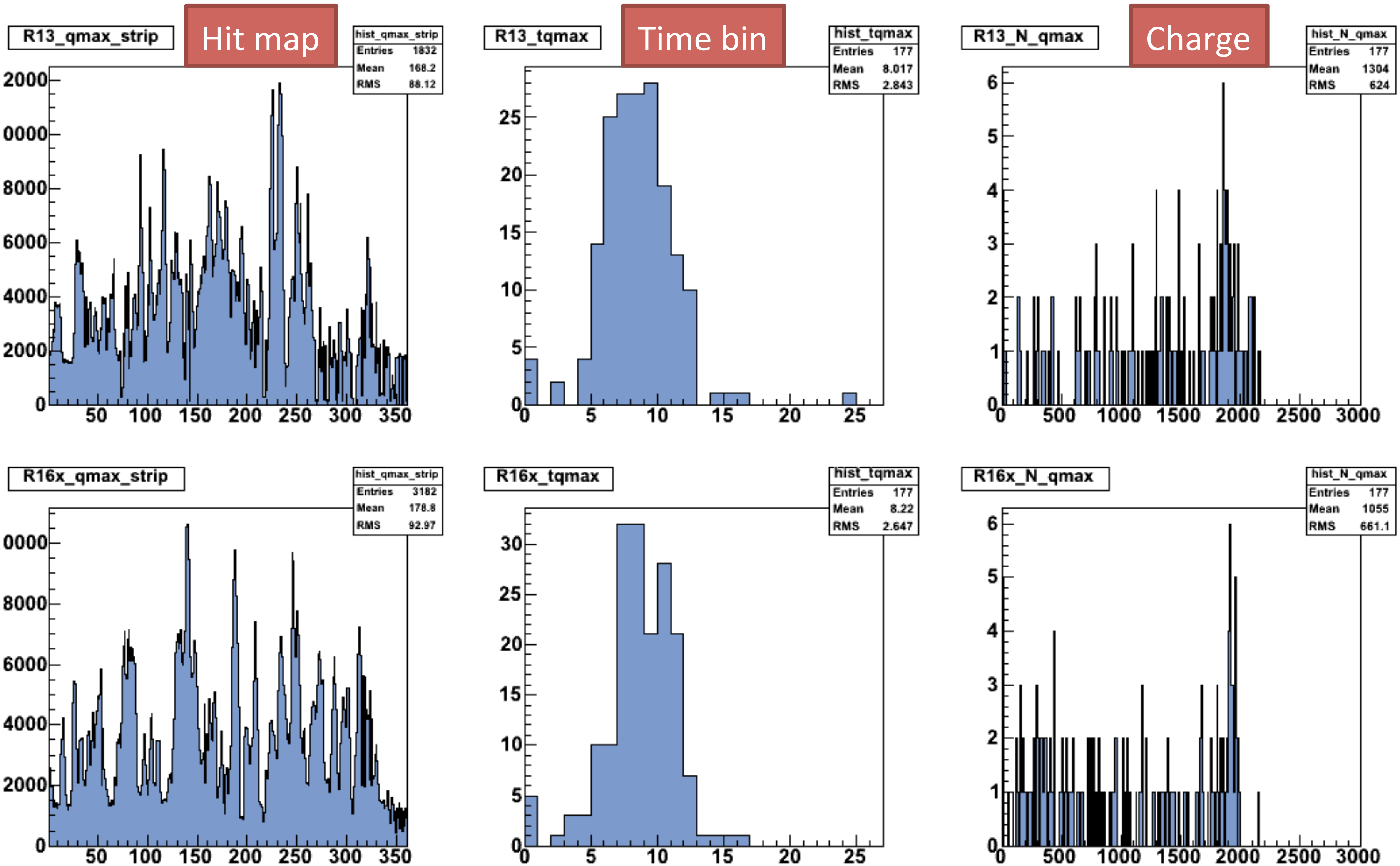
mmdaq checks data for obvious configuration errors

- Bad number of chips (compared to config)
- Bad number of time bins (between chips)

Single event display



mmDAQ



Hit map
Time bin
Charge
R13_qmax_strip
hist_qmax_strip
Entries 1832
Mean 168.2
RMS 88.12
R13_tqmax
hist_tqmax
Entries 177
Mean 8.017
RMS 2.843
R13_N_qmax
hist_N_qmax
Entries 177
Mean 1304
RMS 624
R16x_qmax_strip
hist_qmax_strip
Entries 3182
Mean 178.8
RMS 92.97
R16x_tqmax
hist_tqmax
Entries 177
Mean 8.22
RMS 2.647
R16x_N_qmax
hist_N_qmax
Entries 177
Mean 1055
RMS 661.1

Work in progress: mmDAQ2

- Redesigned mmDAQ

- Work in progress:
  - back-bone ready (network readout, basic Event Building)
  - UI and data output
  - EB filters

Characteristics:

- Multiple data sources: FECs, SRU (through ethernet)
- Different front-end HW side-by-side
- Online monitoring, possibility to define a user module

Modular design mmDAQ2

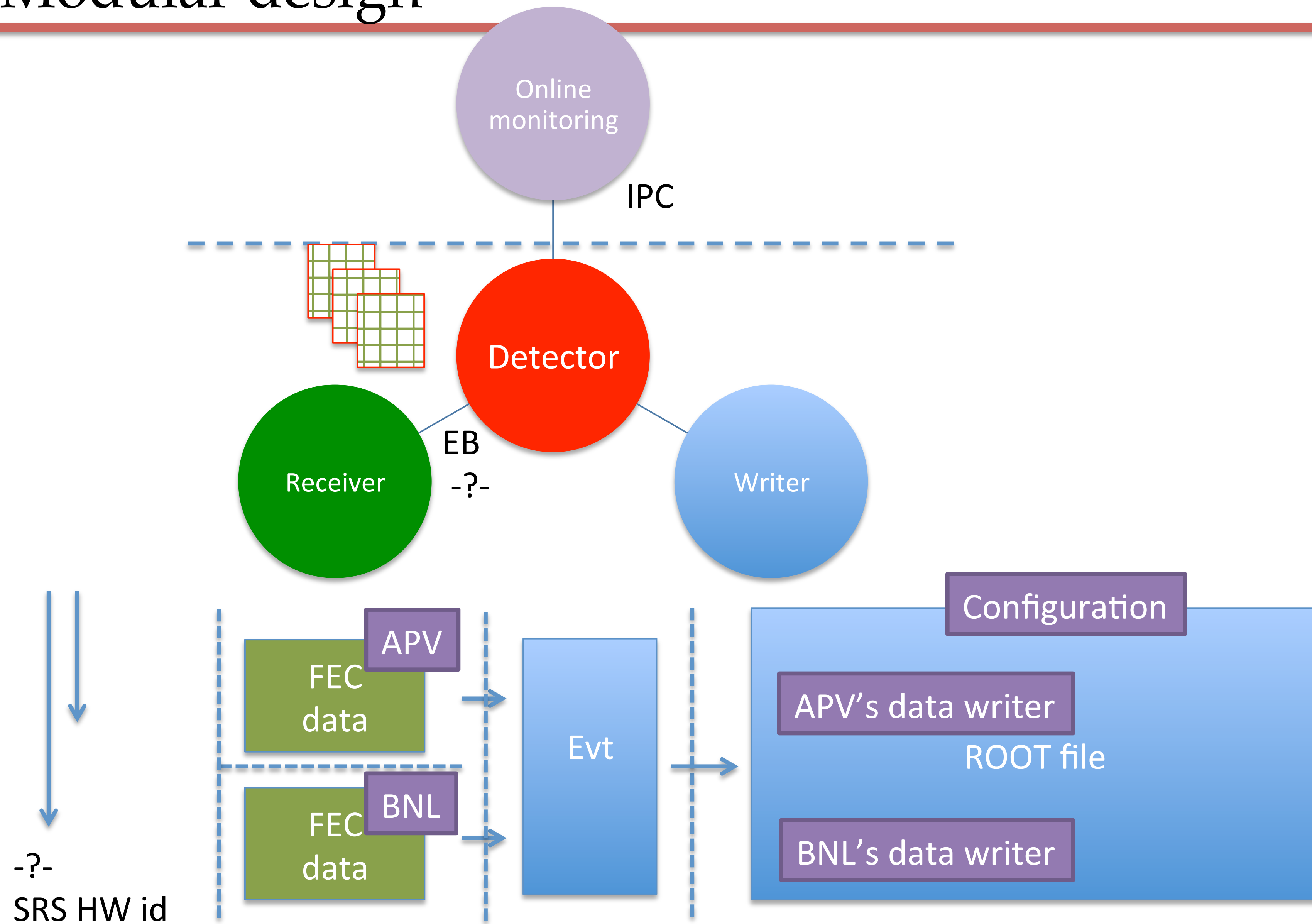
Modular:
adding new HW requires sub classing of 2 classes
Example for BNL's VMM1 chip
- Event class (defines data decoding) + Channel class (defines data format)

```
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

   };

}
```

Beetle:
SRS HW code, data format, error conditions ?

Modular design



Summary

- Integration with ATLAS DAQ is being finalized
- mmDAQ - functional
- mmDAQ2 - versatile version is being developed

Backup slides

FEC zero suppression (type I)

- 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
WORD16  FLAGS;  // ZS mode
WORD16    RESERVED;   // 16 bits reserved for future use
};

struct CHAN_INFO
{
BYTE        CHAN_ID;
BYTE        RESERVED;           // 8-bits reserved for future use
// 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
};
```

LVL1 handling scheme