Status of the EMC Slow Control

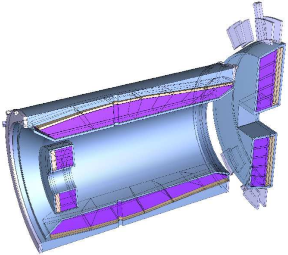
Florian Feldbauer

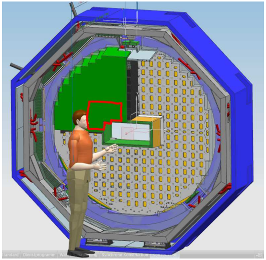
Experimentelle Hadronenphysik
Ruhr-Universität Bochum

XXXV. $\overline{\mathrm{P}}$ANDA Collaboration Meeting
December 3rd, 2010

$\overline{\text{P}}$ANDA Electromagnetic Calorimeter and Proto192

- Electromagnetic calorimeter (EMC) of the $\overline{\text{P}}$ANDA target spectrometer consists of $\sim$ 16000 PWO crystals
- Designed as barrel with 2 endcaps
- Cooled down to $-25\,^{\circ}$C to increase light yield of PWO by factor 4
- Proto192:
  - Prototype of the forward endcap of the EMC consisting of 216 PWO crystals
  - Allows tests of mounting, cooling, read-out electronics and slow control

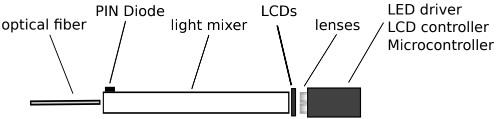


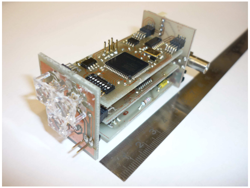


Overview of the Slow Control for EMC Proto192

- Monitoring temperature and humidity
  Temperature and Humidity Monitoring Board for $\overline{\text{P}}$ANDA (THM$\overline{\text{P}}$)
  Custom hardware with CAN interface
- Controlling LED pulser for monitoring radiation damages and transmittance of the PWO crystals
  Custom hardware with CAN interface
- Monitoring parameters of VME crate by Wiener via CAN interface
- Controlling power supplies:
  - Photodetectors: ISEG EHS 8 620p-F and EHS 8 210p-F modules with ECH238 controller with CAN interface
  - LED pulser: ISEG NHQ202M with RS232C interface
- Controlling chiller (LH47 and FP50 from Julabo) via RS232C interface
- Monitoring pressure in vacuum shield and endcap with hardware from Pfeiffer Vacuum via RS232C interface

LED Pulser

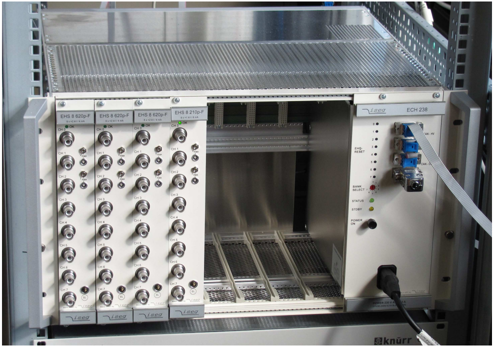




- Light pulser foreseen to check the proper operation of all EMC channels
- Radiation damages reduce the light transmittance of PWO
- With a light pulser the detection of radiation damages of the crystals and photodetectors is possible
- Requirements for the light pulser
  - Pulse form like PWO signal
  - Different colors (blue, green, red)
  - High light output
  - Intensity variation in a wide range (1:1000)
  - Small dimensions

ISEG ECH238 crate controller and modules

- EHS 8 620p-F: module with 8 channels
  Regulated 0 to 2 kV DC output, up to 4 mA
- EHS 8 210p-F: module with 8 channels
  Regulated 0 to 1 kV DC output, up to 8 mA
- 8 modules per crate and 8 crates per CAN-Bus
- Read out / set voltage and current, read out status of each channel



Chiller from Julabo

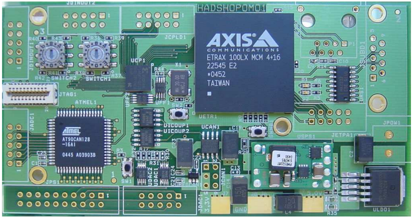
- LH47: 1300 W at $-20\,^{\circ}$C, 300 W at $-40\,^{\circ}$C
  FP50: 500 W at $-20\,^{\circ}$C, 160 W at $-40\,^{\circ}$C
- Cooling fluid: Methyl water (1:1)
  Considering Perfluorohexane (C_6F_{14})
- Controllable via RS232C interface
- Read out all parameters
  Set temperature and pump stage

CAN Bus Interface

- Used the I-7565 USB/CAN Converter for CAN-Bus
- Works with devices with little communication load (e.g. VME crate)
- Insufficient for devices with high data exchange like EHS HV-modules from ISEG
- No buffer for USB transmission → loss of sent data

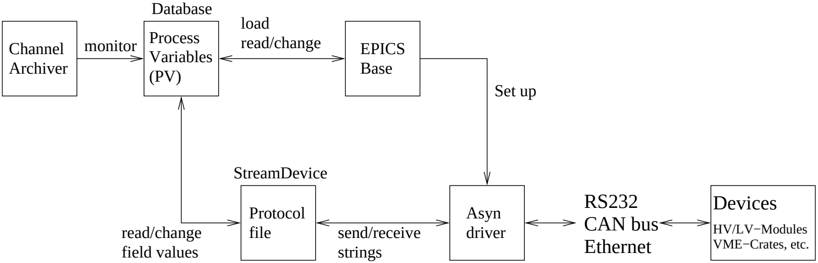
CAN Bus Interface

- Now using HADControl from HADES group (M.Traxler)

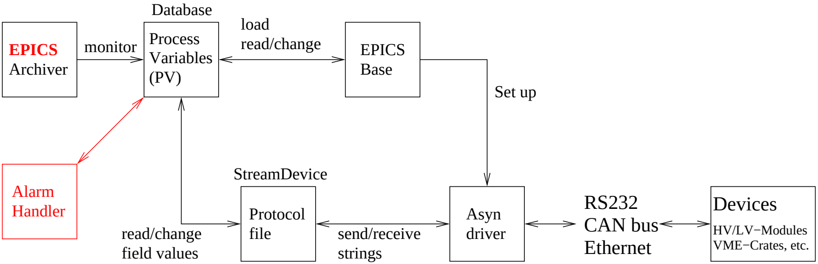


- ETRAX 100LX embedded CPU running EPICS
- Microcontroller AT90CAN128 with CAN interface connected via serial interface
- Command for sending messages via CAN:
  `SEND ID IDMSK RTR DLC D0 D1 D2 ...`

EPICS Overview



EPICS Overview



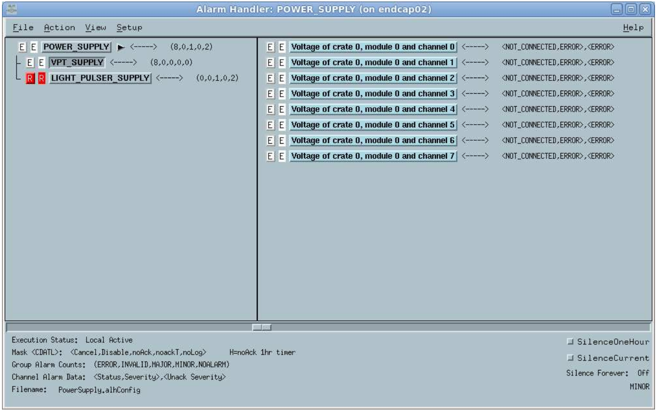
EPICSArchiver

- Using EPICSArchiver 1.0.2
- Stores values of PV in MySQL-database
- Configuration in MySQL
- Monitor a new PV → `pvarch add_pv MyPV.VAL`
  or list of PVs: `pvarch add_pvfile PVList.txt`

Alarm Handler

- Interactive graphical application that monitors EPICS database alarm states
- Monitored PVs are arragned in trees
- Allows logging alarms

Alarm Handler



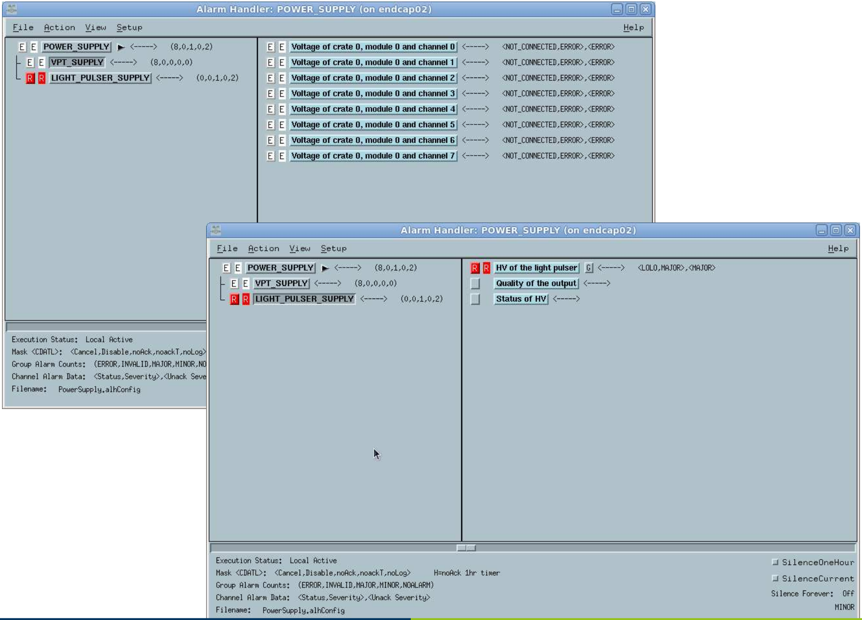
```
E E POWER_SUPPLY ▶ <-----> (8,0,1,0,2)
├ E E VPT_SUPPLY <-----> (8,0,0,0,0)
└ R R LIGHT_PULSER_SUPPLY <-----> (0,0,1,0,2)
```

```
E E Voltage of crate 0, module 0 and channel 0 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 1 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 2 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 3 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 4 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 5 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 6 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 7 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
```

```
Execution Status:  Local Active
Mask <CDATL>:  <Cancel,Disable,noAck,noackT,noLog>      H=noAck 1hr timer
Group Alarm Counts:  (ERROR,INVALID,MAJOR,MINOR,NOALARM)
Channel Alarm Data:  <Status,Severity>,<Unack Severity>
Filename:   PowerSupply.alhConfig
```

```
□ SilenceOneHour
□ SilenceCurrent
Silence Forever:  Off
MINOR
```

Alarm Handler



```
Alarm Handler: POWER_SUPPLY (on endcap02)

E E POWER_SUPPLY ▶ <-----> (8,0,1,0,2)
  E E VPT_SUPPLY <-----> (8,0,0,0,0)
  R R LIGHT_PULSER_SUPPLY <-----> (0,0,1,0,2)

E E Voltage of crate 0, module 0 and channel 0 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 1 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 2 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 3 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 4 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 5 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 6 <-----> <NOT_CONNECTED,ERROR>,<ERROR>
E E Voltage of crate 0, module 0 and channel 7 <-----> <NOT_CONNECTED,ERROR>,<ERROR>

Execution Status:  Local Active
Mask <CDATL>:  <Cancel,Disable,noAck,noackT,noLog>
Group Alarm Counts:  (ERROR,INVALID,MAJOR,MINOR,NO
Channel Alarm Data:  <Status,Severity>,<Unack Seve
Filename:  PowerSupply.alhConfig
```

```
Alarm Handler: POWER_SUPPLY (on endcap02)

E E POWER_SUPPLY ▶ <-----> (8,0,1,0,2)
  E E VPT_SUPPLY <-----> (8,0,0,0,0)
  R R LIGHT_PULSER_SUPPLY <-----> (0,0,1,0,2)

R R HV of the light pulser G <-----> <LOLO,MAJOR>,<MAJOR>
□   Quality of the output <----->
□   Status of HV <----->

Execution Status:  Local Active                                          □ SilenceOneHour
Mask <CDATL>:  <Cancel,Disable,noAck,noackT,noLog>    H=noAck 1hr timer
Group Alarm Counts:  (ERROR,INVALID,MAJOR,MINOR,NOALARM)                 □ SilenceCurrent
Channel Alarm Data:  <Status,Severity>,<Unack Severity>                  Silence Forever:  Off
Filename:  PowerSupply.alhConfig                                                     MINOR
```

Alarm Handler

- Write config file for each device: (example LED Pulser)

```
GROUP NULL LIGHT_PULSER_SUPPLY
CHANNEL LIGHT_PULSER_SUPPLY Readvoltage1
$ALIAS HV of the light pulser
$GUIDANCE
Voltage above 700 V can damage the light pulser.
$END
CHANNEL LIGHT_PULSER_SUPPLY QualOfOutput1
$ALIAS Quality of the output
```

- Write global config file to include individual config files

Subroutine Records

- ISEG ECH238 HV module
  - Uses 32-bit float (IEEE-754) values as input/output values
  - HADControl Send Command:
    `SEND ID IDMSK RTR DLC D0 D1 D2 ...`
  - Split float value in 4 bytes in hex representation and join 4 bytes in hex representation to float, respectively

    Set voltage of channel 0 to 1200 V
    ⇒ `SEND 200 0 0 7 41 0 0 44 96 0 0`
- Solved problem using subroutine records
- Used to call C initialization routine and recurring scan routine
- No device support ⇒ further record is needed for communication with StreamDevice
- 12 input fields (INPA-INPL)

Subroutine Records

C routine for splitting float value in 4 bytes:

```
union FloatIntconv{
  float floatVal;
  int   intVal;
}floatToInt;

static long ISEGsplit(subRecord *prec) {
  floatToInt.floatVal = (float)prec->val;
  prec->a = ( floatToInt.intVal & 0xFF000000 ) >> 24;
  prec->b = ( floatToInt.intVal & 0x00FF0000 ) >> 16;
  prec->c = ( floatToInt.intVal & 0x0000FF00 ) >> 8;
  prec->d = ( floatToInt.intVal & 0x000000FF );

  return 0;
}
```

Subroutine Records

```
record (sub, "SetVolt") {
  field (INAM, "ISEGinit")
  field (SNAM, "ISEGsplit")
  field (FLNK, "SetVoltCom")
}

record (calcout, "SetVoltCom") {
  field (INPA, "SetVolt.A")
  field (INPB, "SetVolt.B")
  field (INPC, "SetVolt.C")
  field (INPD, "SetVolt.D")
  field (OUT,  "@ECH238.proto SetVolt hadcon")
}
```

Conlusion and Outlook

Conclusion

- Slow Control for PROTO192 complete
- Changed archiver (⇒ EpicsArchiver)
- Included Alarm Handler in EPICS software

Conlusion and Outlook

Conclusion

- Slow Control for PROTO192 complete
- Changed archiver (⇒ EpicsArchiver)
- Included Alarm Handler in EPICS software

Outlook

- Up to now only tested each application for itself
  ⇒ Test all devices together
- Writing GUIs for each device
  Do we want to use CSS for building our GUIs?
  Possible alternatives: EpicsQT, GTK+, MEDM, ...