



Elettra Sincrotrone Trieste

The ESS WS ACQ SYS ICS integration PDR-2



what is a WS

WS: an electro-mechanical device which measures the transverse beam density profile in a particle accelerator by means of a moving thin wire (CERN)

WS input: signal generated by a moving wire (wire: primary sensing element)

WS output: transverse beam density profile

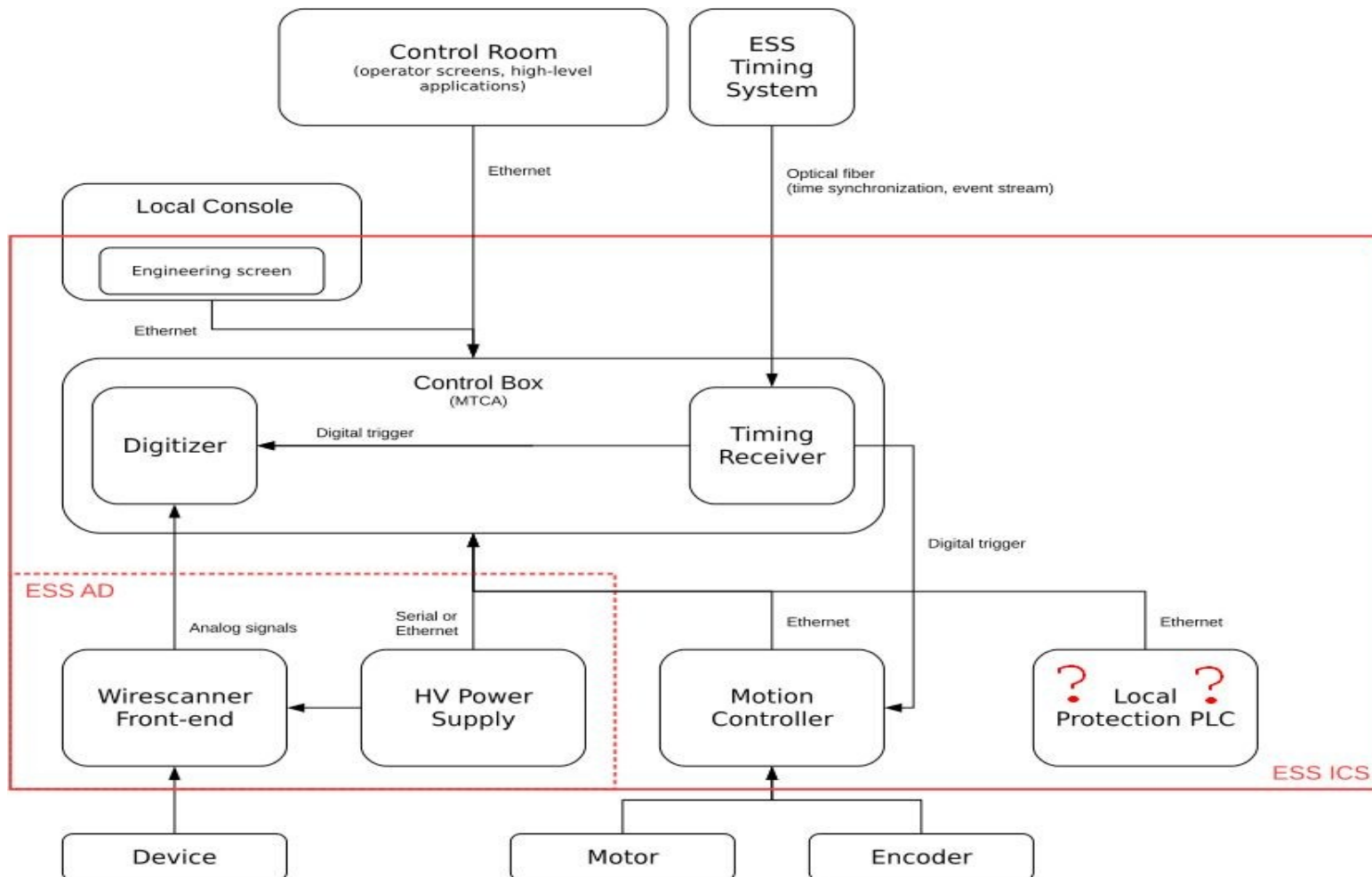
→ beam density profile = $f(\text{wire position, time, signal processing, setup parameters, ...})$

main SW goal

to implement $f()$ by the identification, development and connection of suited functional blocks that can be built by code

Conceptual SW architecture

WS HW description suggests a two layers client/server SW architecture





Two layers SW architecture

The high level layer (HLL - client side) is in charge of human interaction activities (control panel (OPI), Engineering Screen) and, if required or considered worthwhile, should host complex tasks (i.e. scripts, macro, cascaded panels, ...) that are not strictly related to IOC operation

The low level layer (LLL - server side) is in charge of all kind of HW access (device driver) and the logical functionality strictly related to the implementation of the WS transfer function

Ancillary low level tasks: protection and servicing (internal access server (optional))

Constrains

EPICS: IOC concept, several difficulties and constrains about code structure and development to implement complex tasks. Whenever possible or worthwhile all complex tasks should be shifted from LLL (EPICS) to HLL (CSS-boy)

CSS-boy: CSS BOY is an Operator Interface (OPI) development and runtime environment. An OPI is a general GUI but with extra facilities to connect to your live data directly (source: github.com/ControlSystemStudio/cs-studio/wiki/BOY). All HLL activities like control panels (operator panel, Engineering Screen) will be built under it. Thanks to its scripting capabilities, complex tasks not directly involved in IOC activities could be hosted in OPIs.

Expected SW functions – raw estimation

Motion control: scan “trajectory” (step/fly mode) synchronized with the timing system

Data acquisition synchronized with the timing system

Elaboration of the acquired raw data accordingly to specified algorithms

Data presentation

Data archiviatio

Local access server (optional)

Auto and procedure driven test

Data exchange (communication protocols – CA, ethercat, ...)

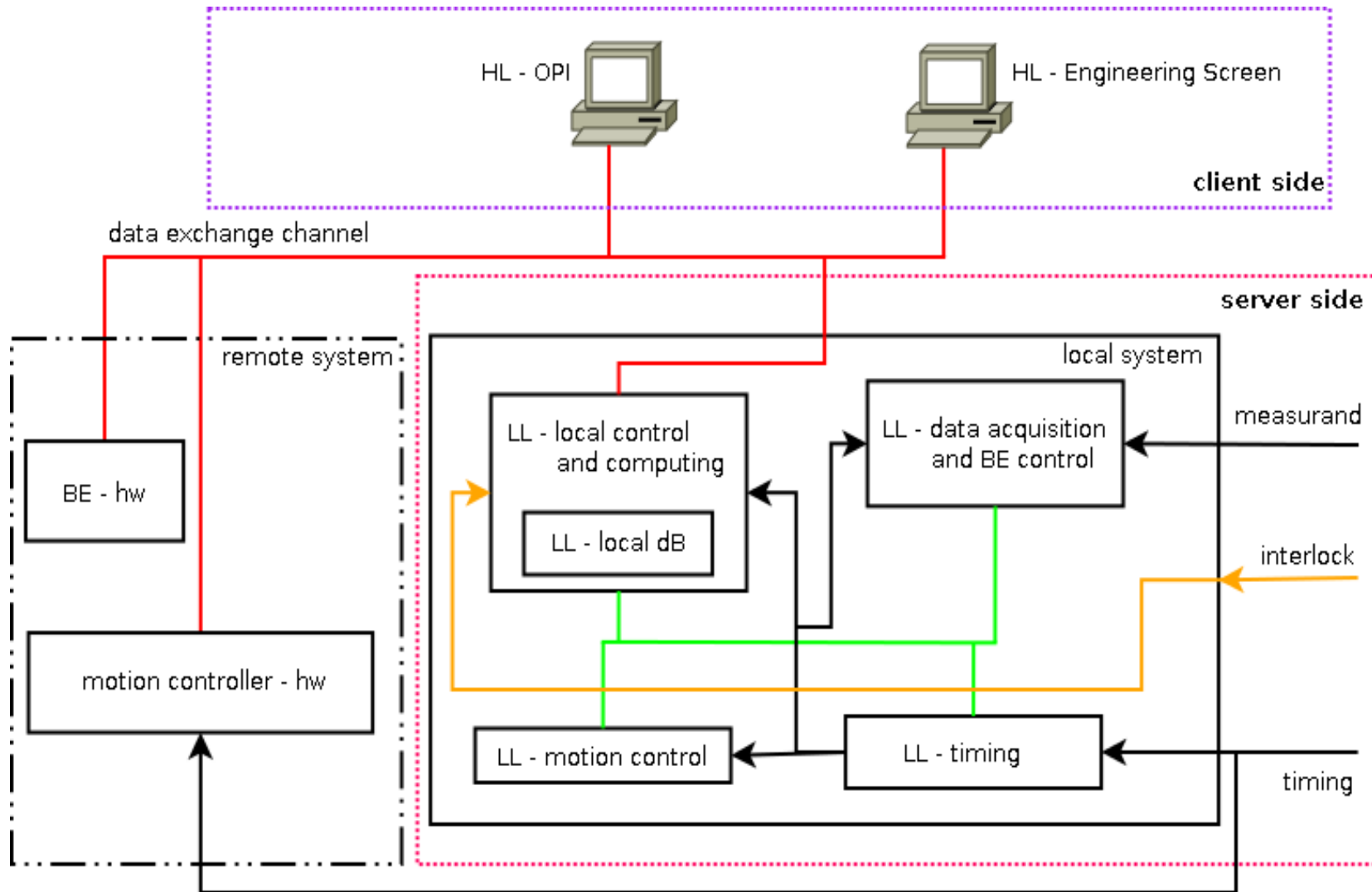
State management (normal operation, alarms, ...)

State machine (sequencer)

Interlocks

HLL: HMI + complex tasks (if worthwhile)

Putting all together: tasks/functions



Thanks to B.C. and H.K.

!!: IN, OUT and calc are referred to the IOC “SW” point of view

ADC RAW acquisition:

devHost:RawCurrentData – array, 400 elements - IN

devHost:CurrentData – array, 400 elements - calc

FE:

devHost:HighVoltagePSvoltage – scalar - OUT

devHost:HighVoltagePS – boolean/array of boolean - OUT

devHost:WireStatus (wire check) – boolean - IN

devHost:Gain (to be precisely defined) – scalar - OUT

algorithm:

devHost:StartOfPulse – scalar (pulse marker) - IN

devHost:EndOfPulse – scalar (pulse marker) - IN

devHost:RMSoverPulse – scalar (calculated RMS) - OUT

acq trigger related - background subtraction and acq delay:

devHost:AcquisitionStartDelay – scalar - IN

devHost:PreTriggerSamples – scalar - IN

devHost:PostPulseSamples – scalar - IN

scan setup:

Original: scan direction, split in two PVs:

devHost:ScanDirectionH – boolean – I/O

devHost:ScanDirectionV – boolean – I/O

Original: type of scan, split in two PVs:

devHost:TypeOfScanFly – boolean – I/O

devHost:TypeOfScanSbS – boolean – I/O

Scan operation (motion control):

devHost:ScanStartPosition – scalar - I/O

devHost:ScanEndPosition – scalar – I/O

devHost:ScanSteps – scalar – I/O

devHost:AbortScan – boolean – I/O

Enable/Disable operation:

devHost:BeamPermit – boolean – IN

?? maybe the BeamShape PV is not present in the document by ESS ?? ... it doesn't matter, it should be defined as an array containing as many values as the steps performed during the scan trajectory

Extra PVs:

devHost:EnableDisable – boolean – IN

a lot of FE settings

temporary debug Pvs

Examples:

```
record(bo, "$(user):TypeOfScanSbS") {  
    field(DESC, "TypeOfScan_StepByStep")  
    field(ZNAM, "Off")  
    field(ONAM, "On") }
```

```
record(ao, "$(user):RMSoverPulse") {  
    field(DESC, "RMSoverPulse")  
    field(SCAN, "1 second")  
    field(PREC, "4") }
```

Example: RMS routine

```
static int training_asubRMS(aSubRecord *precord) {  
    float RMS, *a;  
    unsigned long nelements; a;  
    int i;  
    a = (float *)precord->a;  
    nelements = precord->nea;  
    RMS = 0;  
    for (i = 0; i < nelements; i++) {  
        RMS += (a[i]*a[i]); }  
    printf("RMS: element %d\n",i);  
    RMS /= nelements;  
    RMS = sqrt(RMS);  
    *(float *)precord->vala = RMS;  
    printf("RMS: %f\n",RMS);  
    return 0; }
```

Example: RMS routine record

```
record(aSub, "$(user):do_RMS") {  
  field(SCAN, "1 second")  
  field(SNAM, "training_asubRMS")  
  field(FTA, "FLOAT")  
  field(NOA, "400")  
  field(INPA, "$(user):CSdata1")  
  field(FTVA, "FLOAT")  
  field(NOVA, "1")  
  field(OUTA, "$(user):RMSoverPulse")  
}
```

At a given wire position, the wire generates 2 Analog Input signals (measurand) that are acquired by 2 ADCs:

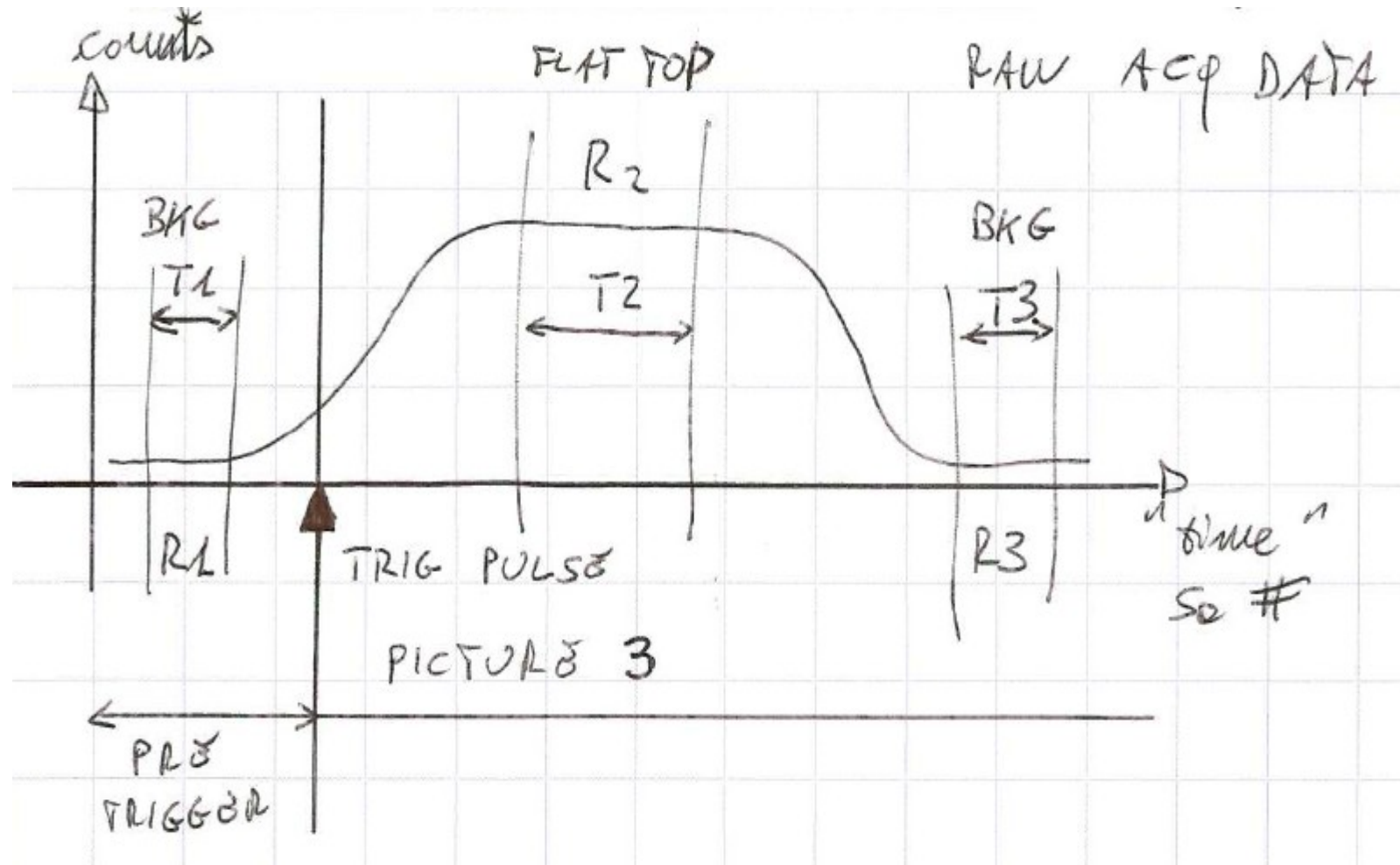
- low gain wide range raw current (LGWB);
- high gain small range raw current (HGNCB);

How to acquire data:

- FE gain should be preliminary set up at a given “good” value
- If the HGNCB branch doesn't saturate (RAW data don't contain any 0xFFFF value) its data are extracted from its buffer to be processed, but...
- ...if it saturates the data acquired by LGWB are used instead

Pulse shape (estimated)

Estimated shape of the pulse acquired by ADC, RAW data



Due to the lack of a real hardware to be connected to our HW pulse generator, an ADC raw data simulator has been developed.

It is still under development but it is able to generate pulses that look like the expected ones.

In the future it could be incorporated in the WS SW as an internal testing tool.

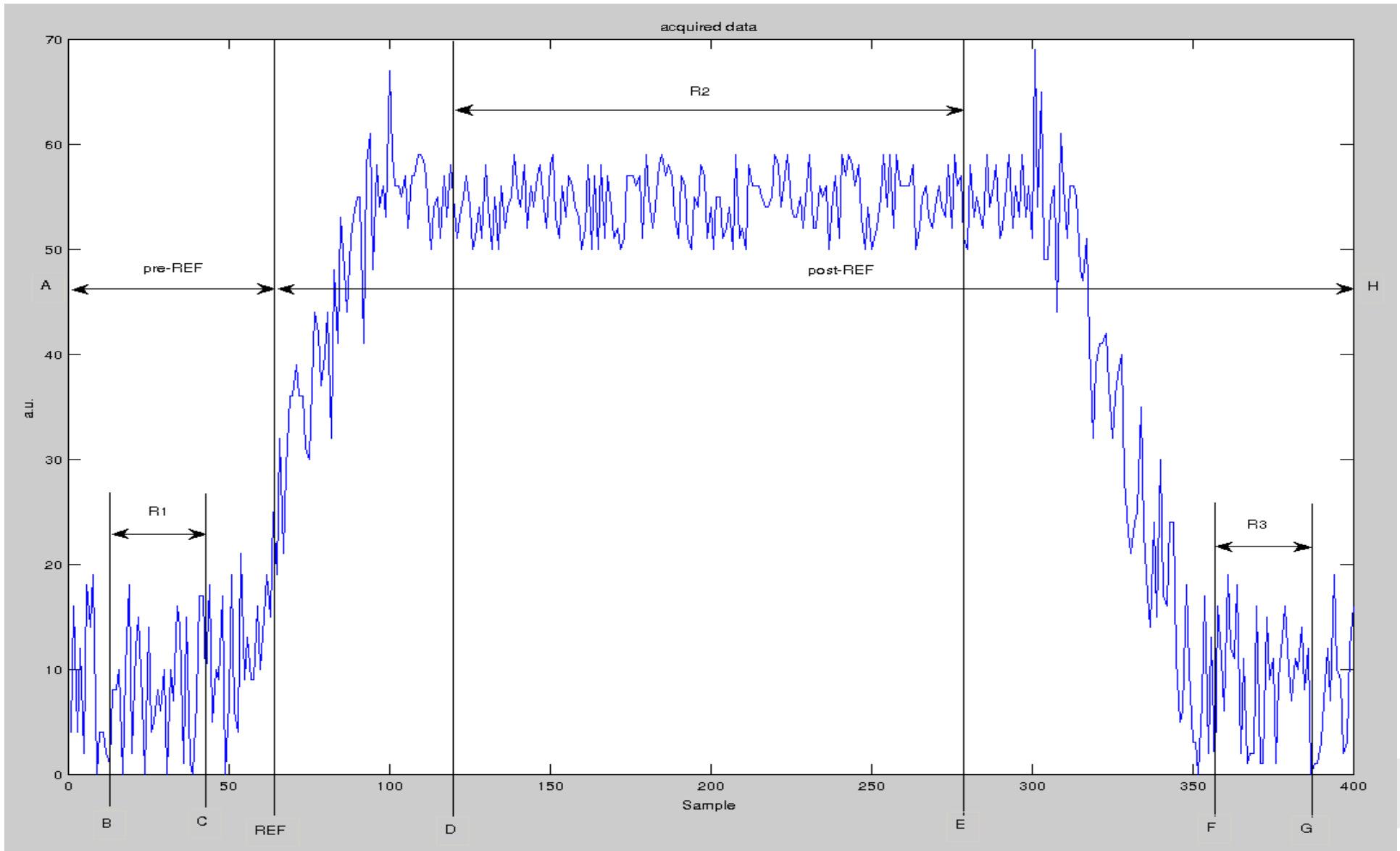
The simulator generates a pulse similar to the one expected with 1Hz repetition rate.

```
caget devHost:Cpdata2
```

```
devHost:CPdata2 400 19 6 9 1 11 4 17 17 16 19 10 1 19 14 3  
18 6 13 14 8 4 3 4 18 9 3 15 13 11 10 2 10 9 4 4 12 8 1 9 4 12  
11 18 11 17 1 2 4 6 16 4 11 13 11 5 5 9 23 14 23 ...
```



Pulse shape (simulated) – RAW data



Parameters defined for the processing of ADC RAW data

R1: pre-trigger samples (PV: PreTriggerSamples) – bkg sub

R2: flat-top, defined as the D-E segment

R3: post-pulse samples (PV: PostPulseSamples) – bkg sub

D: start of pulse (PV: StartOfPulse)

E: end of pulse (PV: EndOfPulse)

(0,0): origin of axis, it is delayed by “acquisition start delay” samples (PV: AcquisitionStartDelay) with respect to trigger signal distributed by the timing system of the accelerator

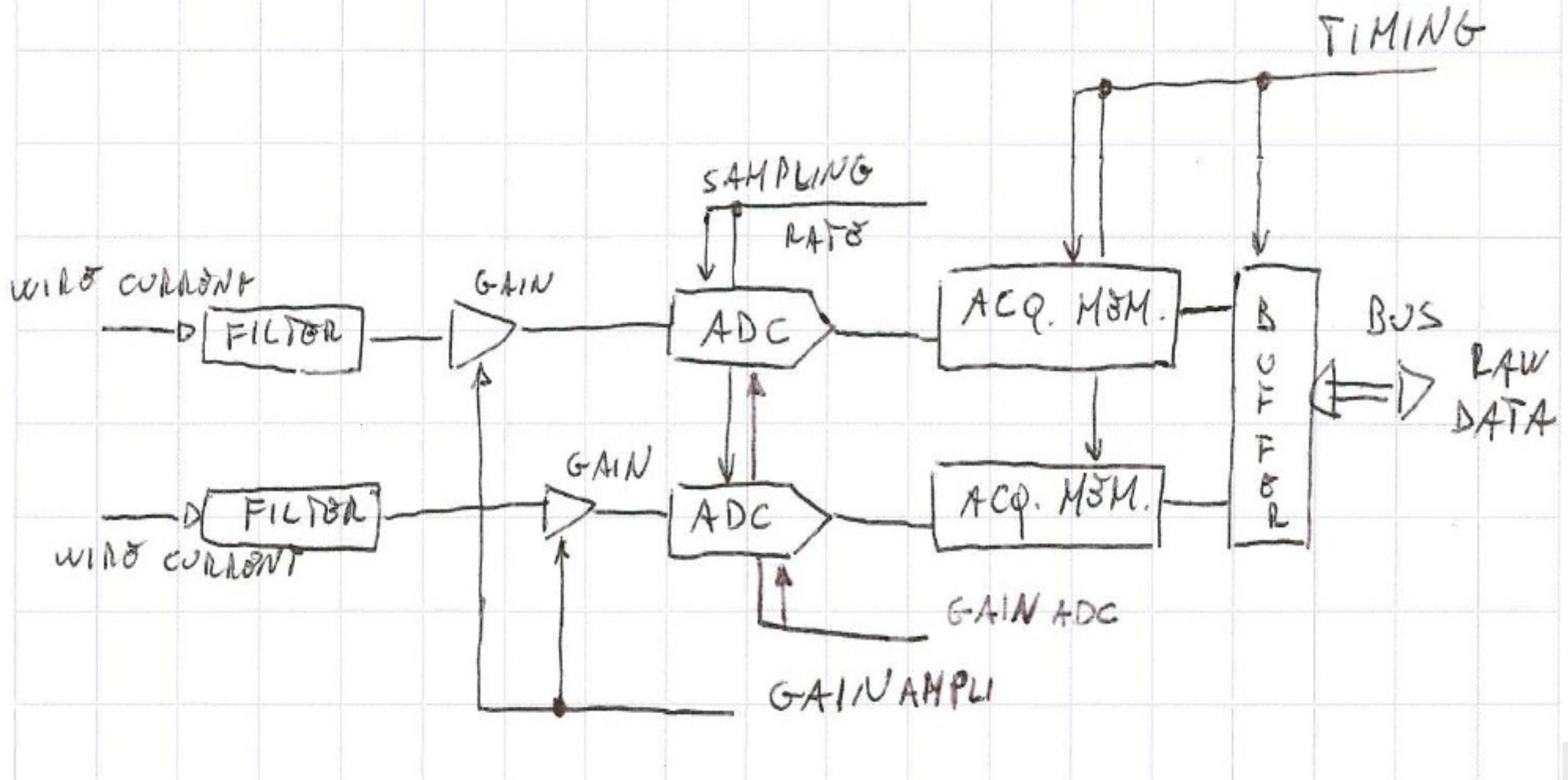
!!: to correctly compute the profile of the wire current the RAW data must be processed accordingly to the setup of the FE (gains, ...)

!!: this kind of processing has to be applied to each set of acquired samples, i.e. must be known the position of the wire inside the vacuum chamber



FE overview (just a model)

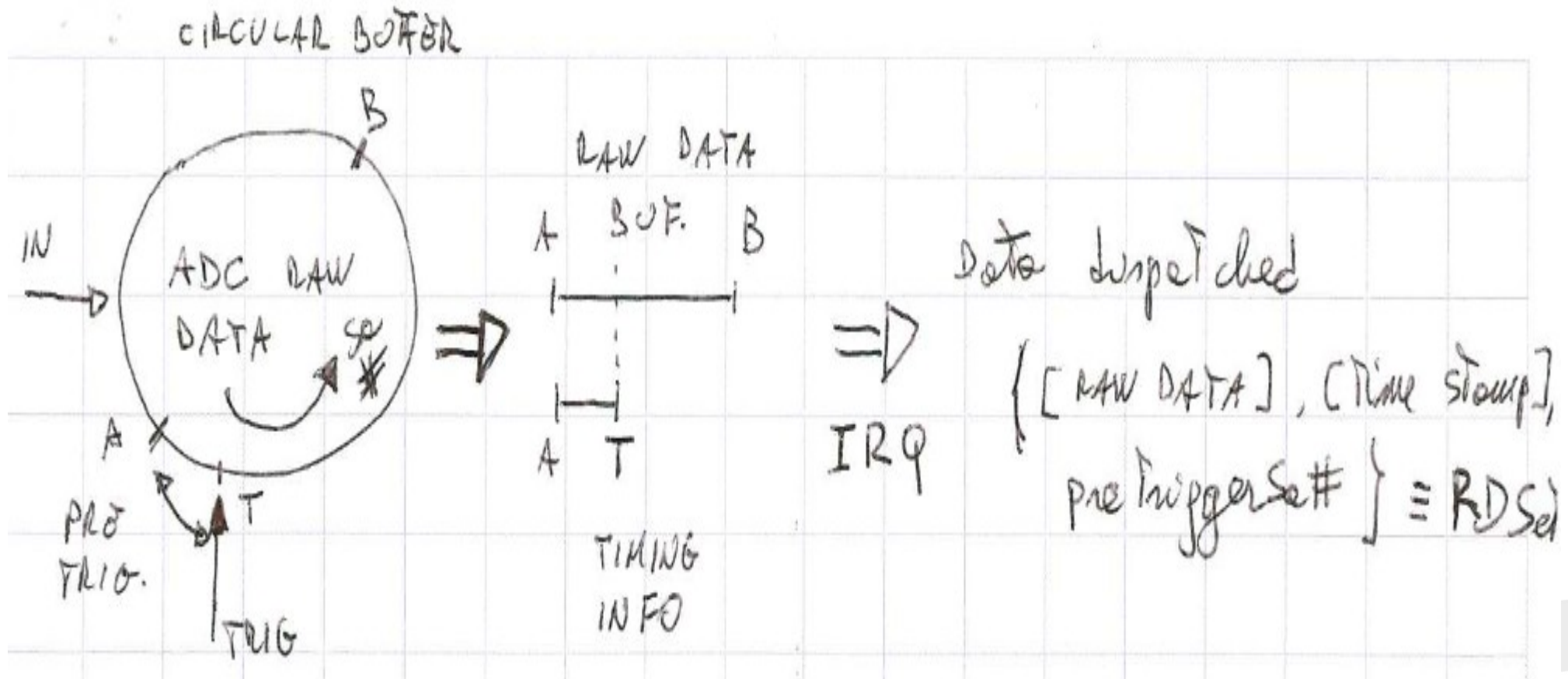
There are still some “gains” that must be converted in PVs





Acquisition vs timing/trigger

It has not yet defined the way to synchronize the acquired data with the motion of the wire and how to time stamp them





Engineering panel - preliminary

Status Flags

PS	MOTORS
<input checked="" type="checkbox"/> AFE 1	<input checked="" type="checkbox"/> H axis
<input checked="" type="checkbox"/> AFE 2	<input checked="" type="checkbox"/> V axis
<input checked="" type="checkbox"/> AFE 3	
<input checked="" type="checkbox"/> BEAM permit	<input checked="" type="checkbox"/> WIRE wire integrity

scan setup

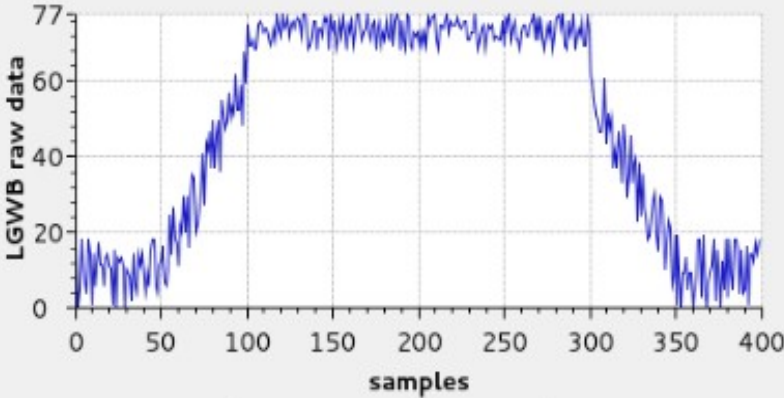
scan direction
 H axis
 V axis

type of scan
 on the fly
 step by step

scan position
 2 start
 0 end
 1 steps

ABORT SCAN

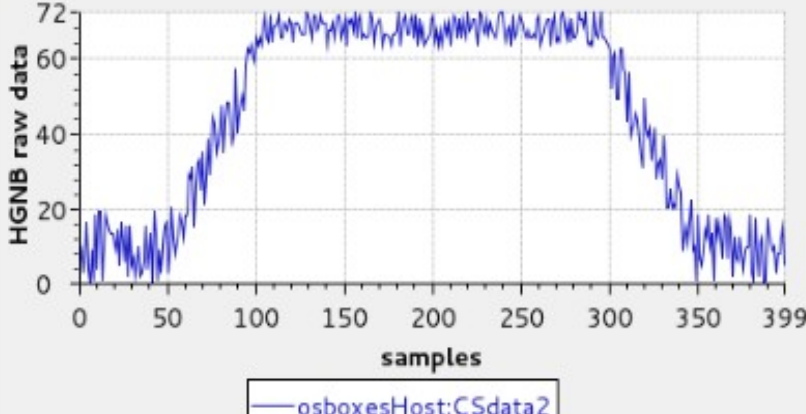
ENABLE



LGWB raw data

samples


— osboxesHost:CSdata1



HGWB raw data


samples


— osboxesHost:CSdata2



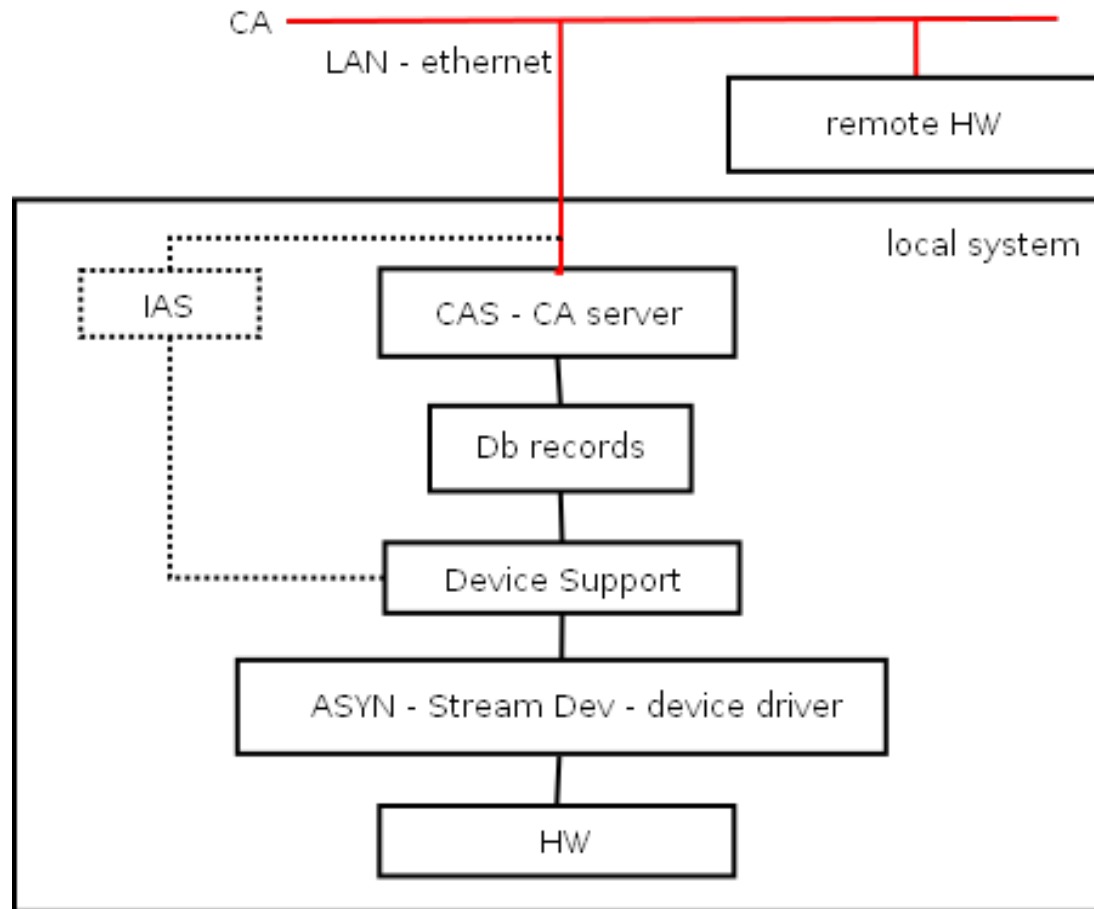
RMS over pulse
44.7613

mean over pulse
39.2375





Protocol file has been added in order to exchange messages with remote FE



Since the PDR-1 meeting a lot of work has been done, so now the the global picture describing the SW is more clear than in the past;

Due to the lack of a complete hardware set, the development of a wire scanner simulator has been started in order to develop the HL application (OPI);

A lot of details have to be still investigated, depending on:

- HW issues (e.g. motion controller driving, uTCA crates and boards, ...);
- EPICS device support issues;
- Low level drivers issues (EPICS device support);
- OPI and Engineering screen definitions;
- ...