EDET 80k.
Firmware & software.

M.Polovykh, A.Vostrukhin

M.Hensel, M. Ibrahim, C. Koffmane, J.Ninkovic, T.Selle, J.Treis, A. Wassatsch

MPG Semiconductor Laboratory

EDET 80k.
Firmware & software.

Introduction

1.Global Overview. Front-end electronics, PL, PS, PC, Server plus user. Technologies.
2.Main tasks assigned to the back-end electronics:
- DAQ
- ASICs configuration
- Housekeeping: Power, monitoring
- Journaling

3.PL overview. Technologies, purpose, connection to FE-electronics. Main blocks.
- Clocking,
- Data source interfaces,
- DDR buffer,
- 10G sinks, signal integrity
- Configuration form PS

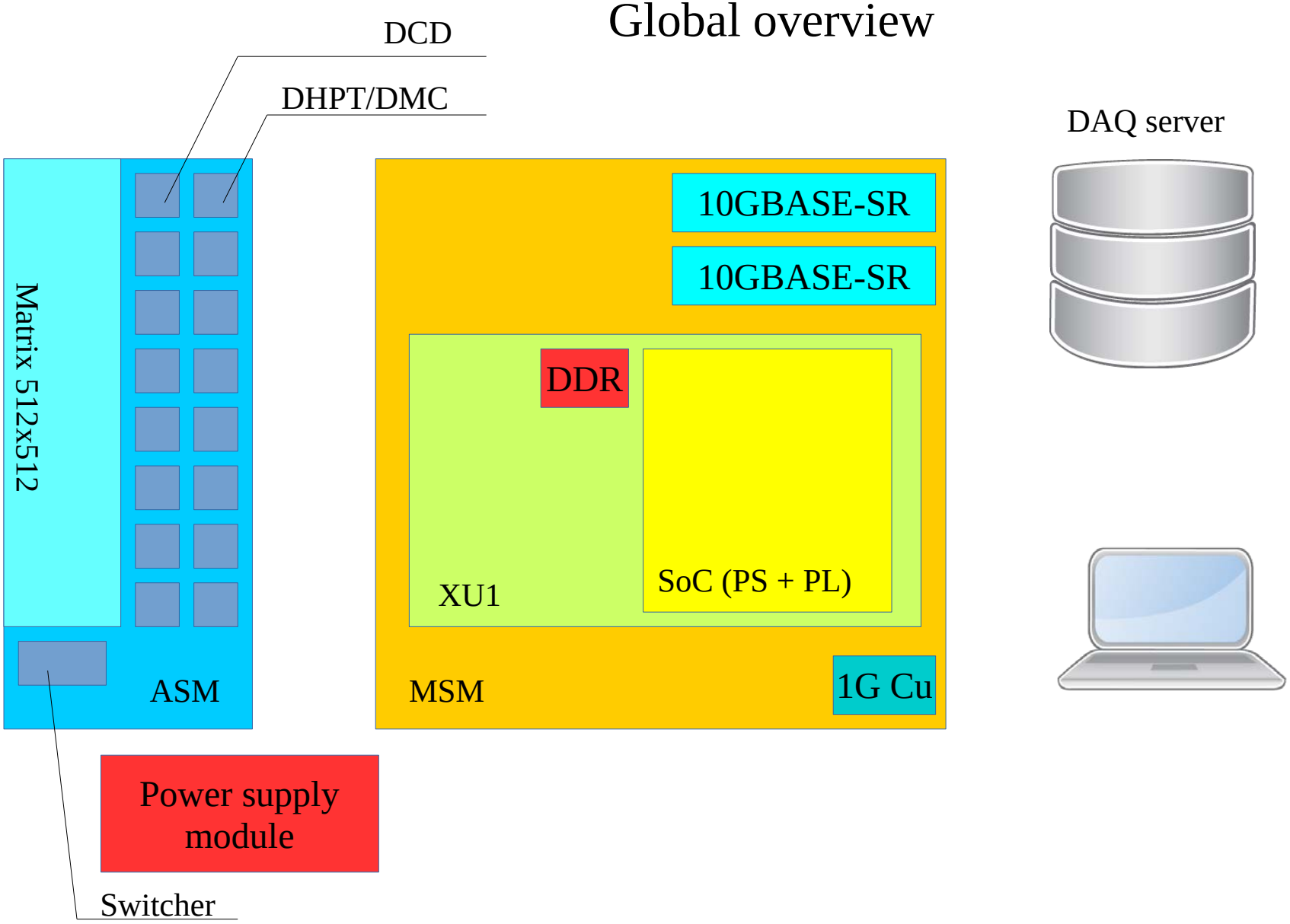
4.PS overview. Technologies, purpose, HW platform, main SW blocks.
5.Detailed PS overview: PMU, FSBL, U-Boot, Linux (Kernel, Device tree, File system), User applications.
- Boot process,
- User applications
- PL peripherals drivers, configuration,
- Low-level housekeeping (I2C subsystem, OMO, statistics gathering)
- JTAG based on OpenOCD and GPIO (low-level)
- Connection to PC software: ssh, sshfs, sockets
- Xserver providing GUI for ssh forwarding.

6.Desktop PC. Overview, main tasks.
- Boot over network support. DHCP, SFTP services.
- Current device state depiction. Includes housekeeping data and picture at full resolution but decimated framerate.
- Device state logging for failures backtracking.
- Hardware configuration user-friendly interface which wraps all the instruments to the one. It includes housekeeping, JTAG, PL peripherals and PL itself.

7.DAQ data server
8. Numbers. Test setup framerate.

EDET 80k.
Firmware & software.

Global overview



EDET 80k.
Firmware & software.

Main tasks

- Two units: PPM, PSP
- 14 adj. power sources
- Configurable I2C bus for control

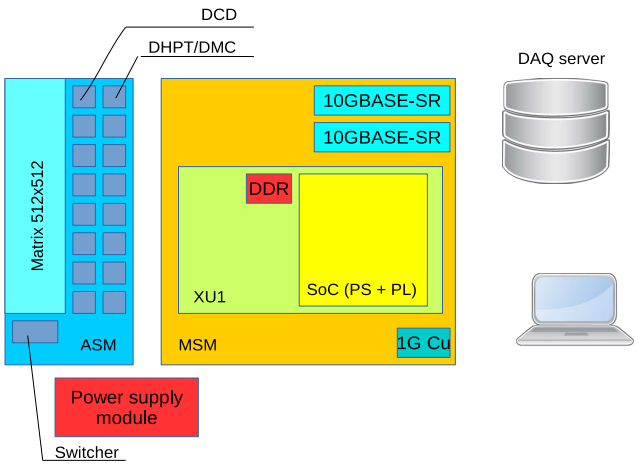
- 3+1 ASIC types
- More than 1k registers (per quadrant)
- Configurable JTAG chain

- Power converters currents and voltages read-out.
- On-line image displaying

- Power supply configuration,
- ASICs configuration,
- Online state monitoring
- Journaling
- DAQ

- DB based state logging
- Play saved data for past events analysis

- Acquire the data from FE-electronics
- Stitch the image (electrical to physical layout)
- XGBase-SR data sink
- Saving the data onto safe storage



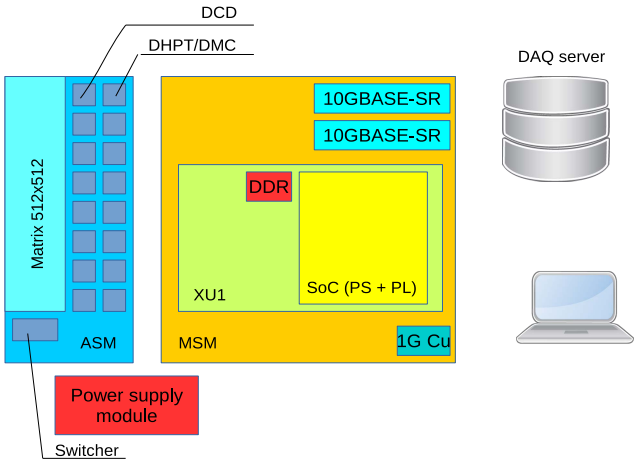
EDET 80k.
Firmware & software.

Technologies stack

Data storage server:

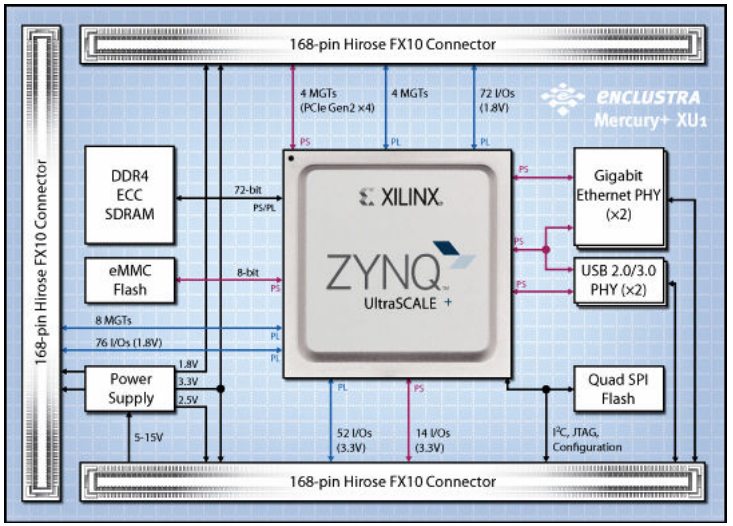
- RAID10 structure
- 2x12 HDD 15000RPM
- 1.2 GB/s write speed (1xDHPT generates ~160MB/s)
- 2x10GBASE-SR

OK for test setup.



Data receiving & ASICs configuration unit:

- High-end Xilinx SoC
  - PS: Cortex A53, Cortex R5 CPUs, MALI400 GPUs
  - PL: >250k Lcells, 12x16Gb/s transceivers
- 2GB DDR RAM



EDET 80k.
Firmware & software.

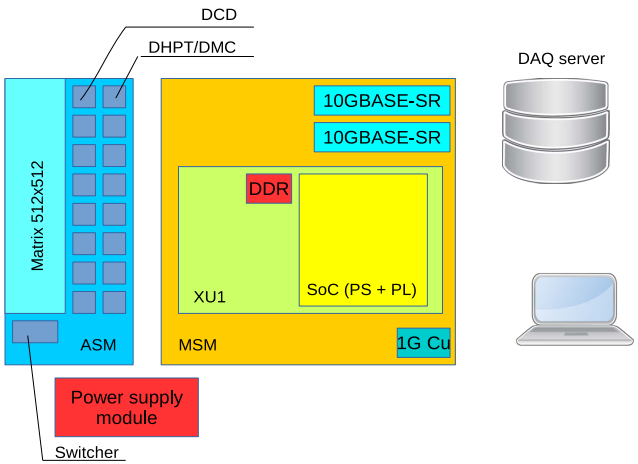
PL tasks overview

- Clocking (for test setup)
- ASICs digital IO interface
- Data receive and image stitch
- DDR RAM FIFO buffer
- UDP/IP/MAC/10GEth
- PL configuration on-the-fly update

- Internal clock management tile, configurable on the fly
- In final setup will be moved to the dedicated clock synthesizer IC. I2C control.

- Manage DHPT/DMC trigger/control line
- JTAG configuration interface
- Data received from DHPT/DMC remapping to form an image

- Images buffering in DDR

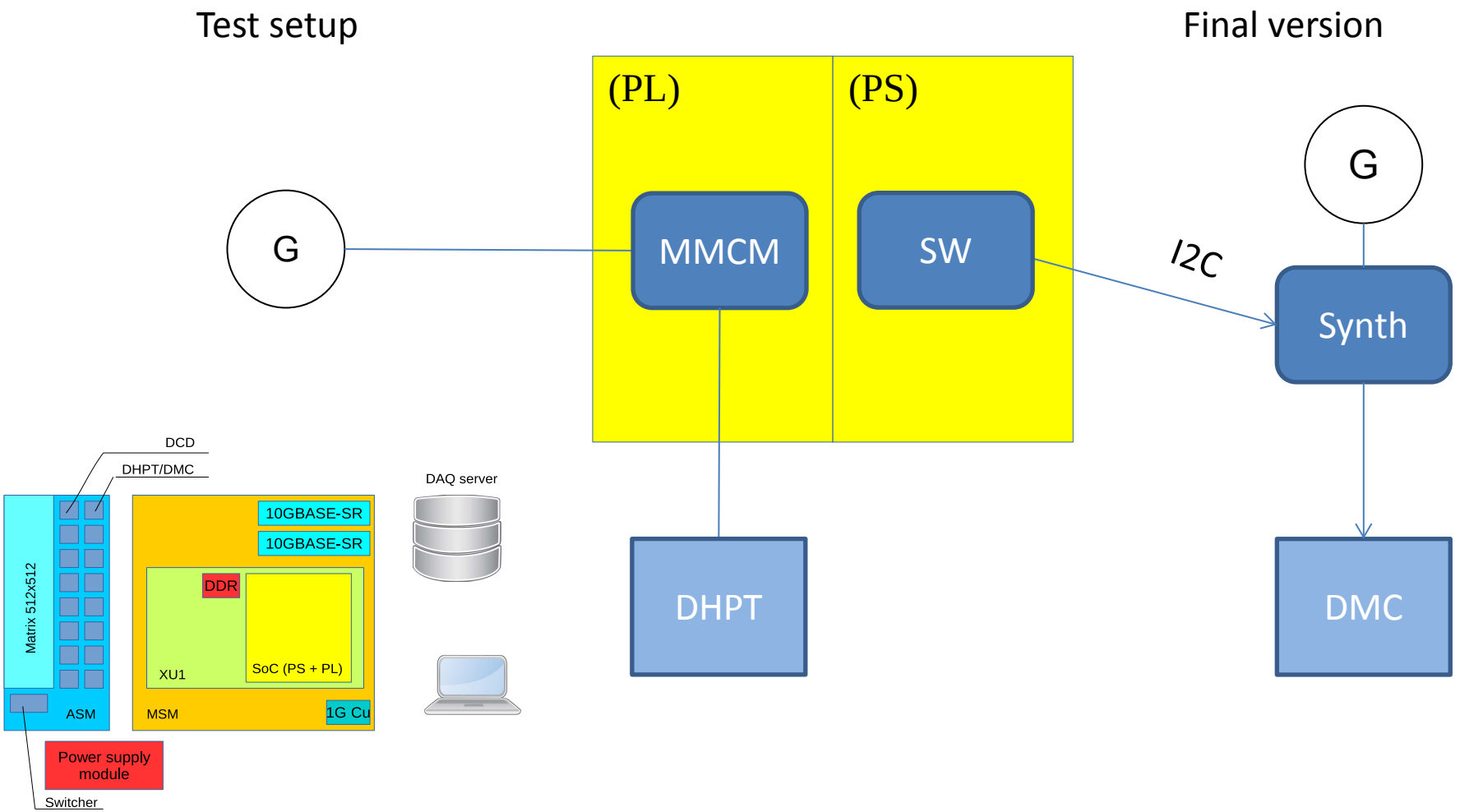


- DDR ring buffer read forms UDP payload
- Ethernet core converts UDP packet into XGMII which interfaces with GTH transceiver.

- PL reconfiguration on-the-fly..

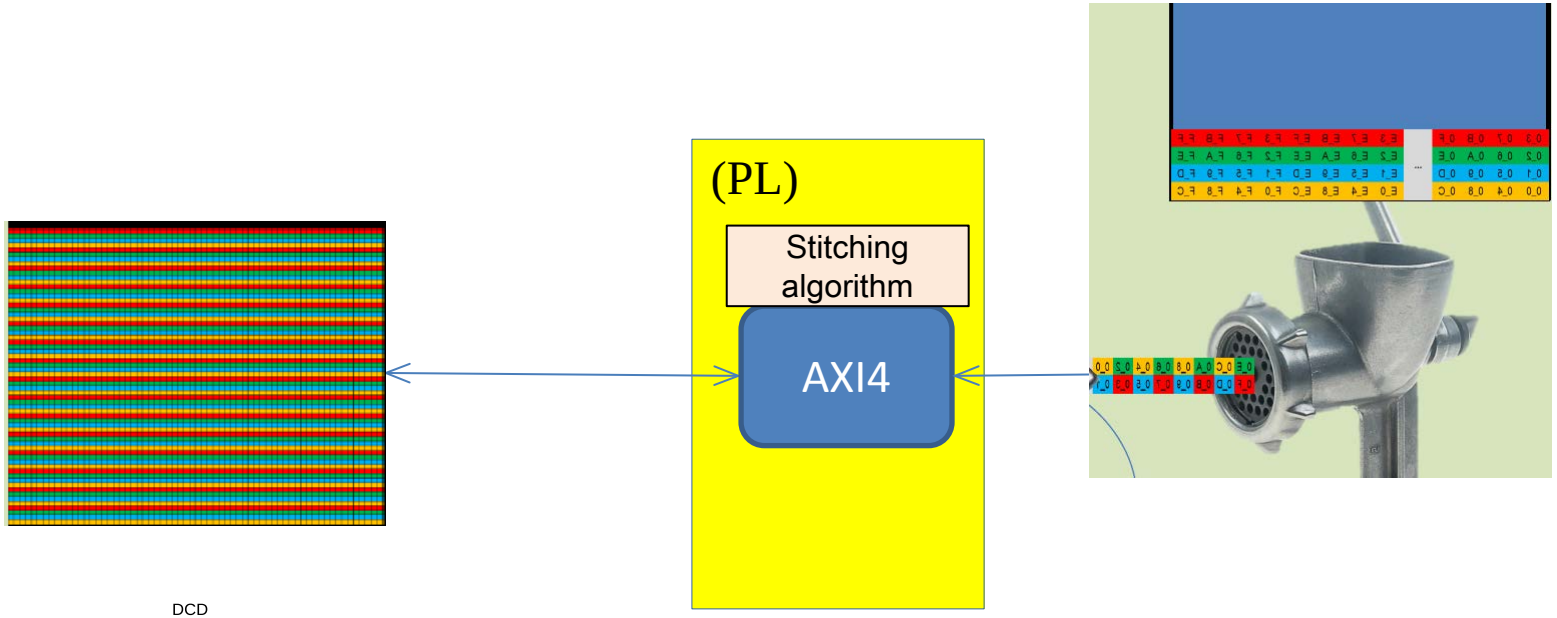
EDET 80k.
Firmware & software.

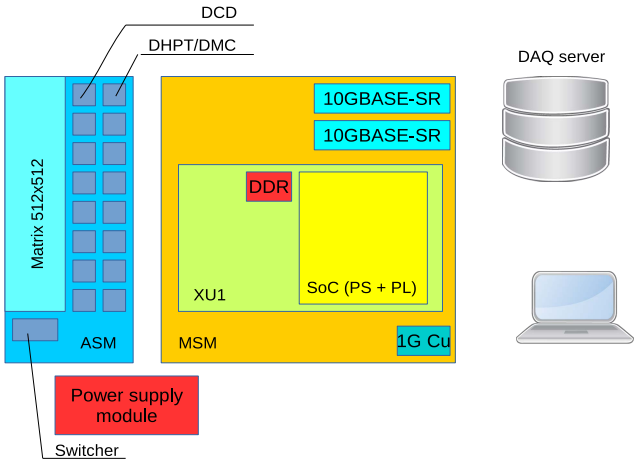
Clocking



EDET 80k.
Firmware & software.

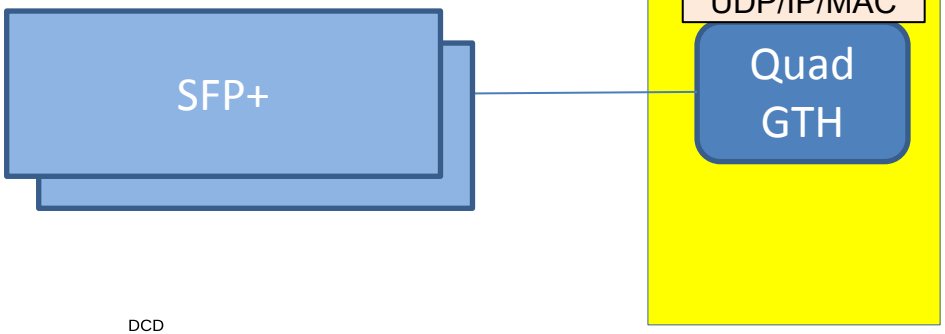
- Data buffer. Image stitching.

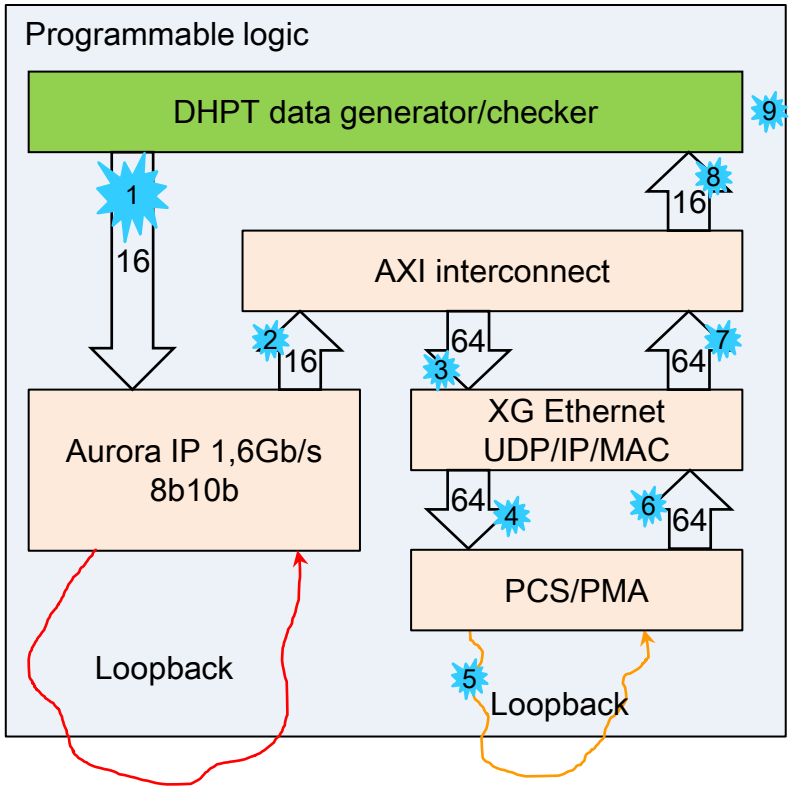


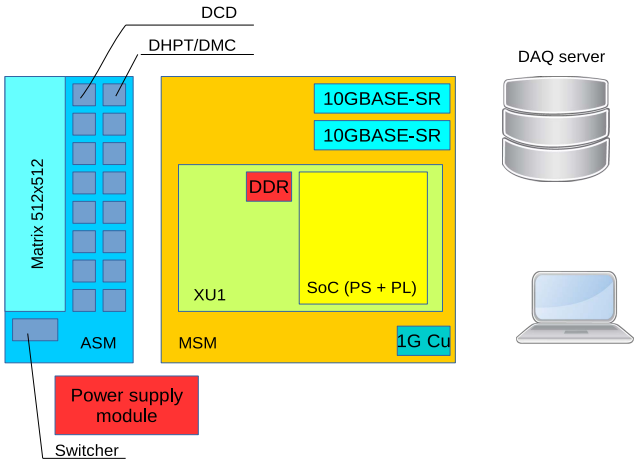


EDET 80k.
Firmware & software.

10G Ethernet



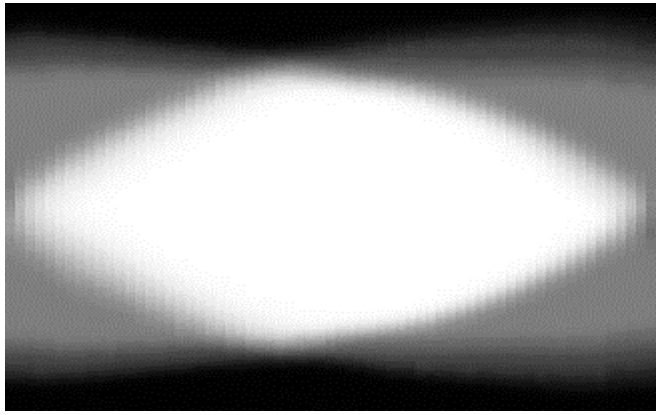




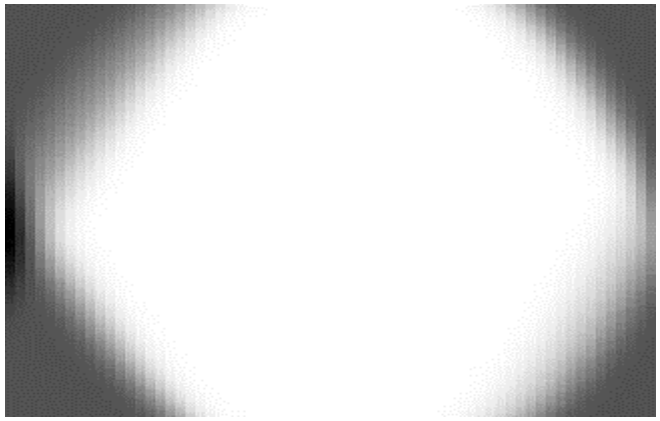
EDET 80k.
Firmware & software.

10G Ethernet and Aurora link. Signal integrity.

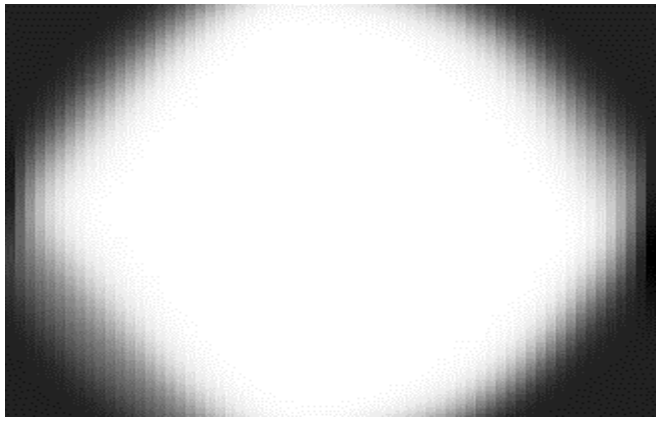
1. Local loopback, OA: 12769 (0%loss)



2. Default settings, OA: 2532 (20%loss)



4. Tx pre-emphasys, OA: 6822



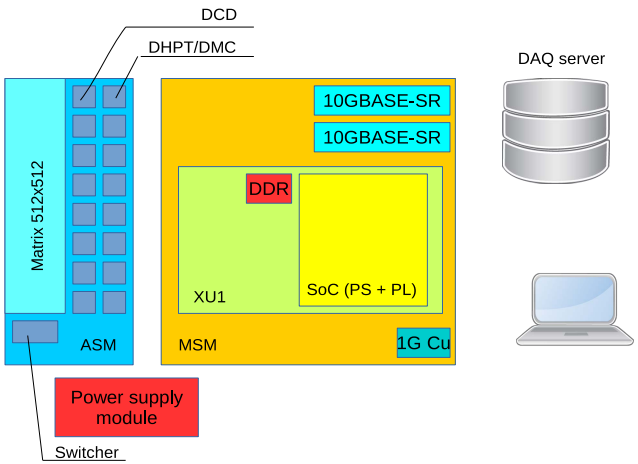
3. Rx equalizer disabled, OA: 4722

EDET 80k.
Firmware & software.

Breathing life into PS.

Boot steps:

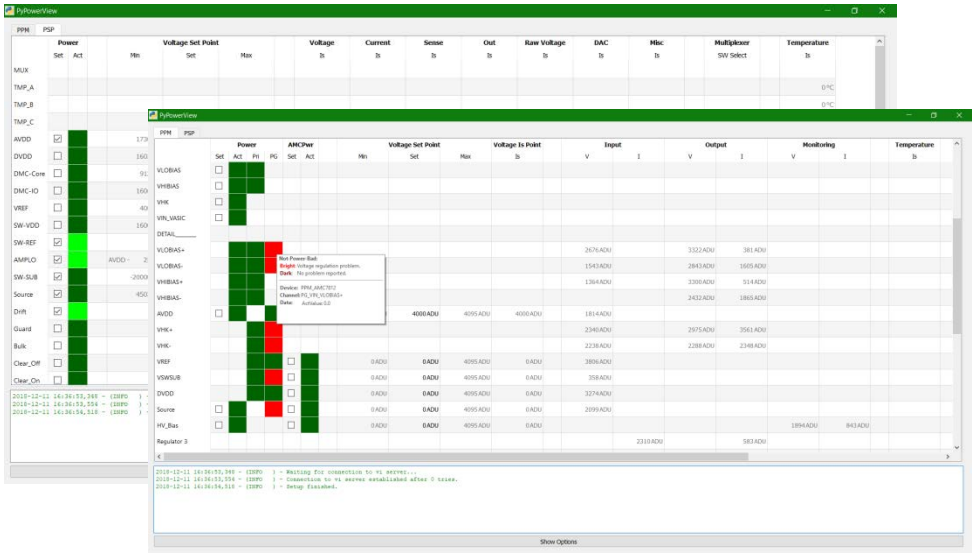
- Power management unit PMU (µBlaze, ROM based, housekeeping)
- CSU: security unit (skipping)
- FSBL (initialize DDR, load 2nd BL)
- 2nd BL (Provide all necessary functionality to load OS)
- Linux
  - Device tree
  - File system
  - Kernel
  - Initialize network interface
  - Load boot script from the server

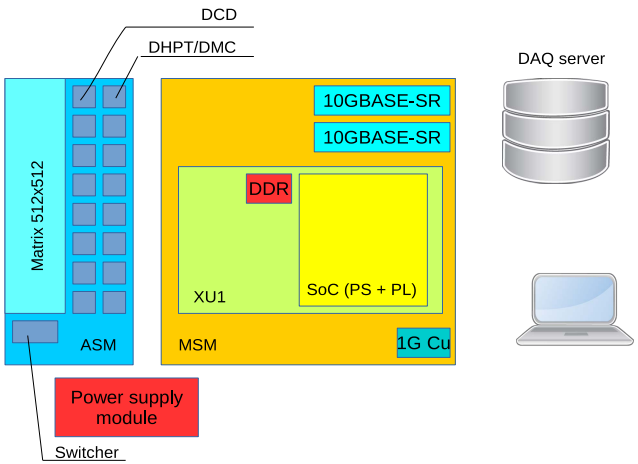


EDET 80k.
Firmware & software.

PS. User software.

- Python:
  - PSP & PPM configuration/monitoring
- C:
  - JTAG bus driver based on OpenOCD

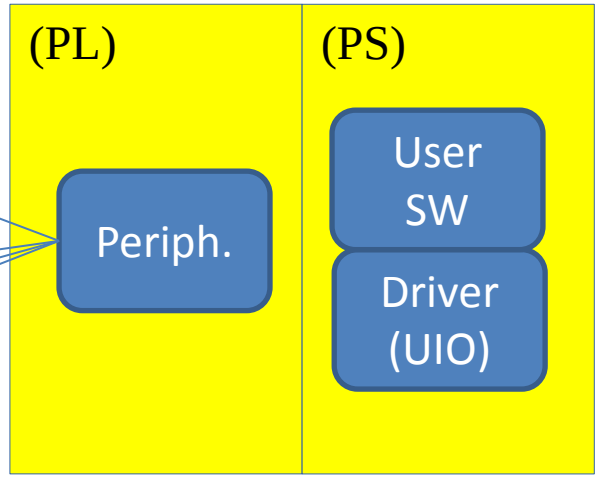


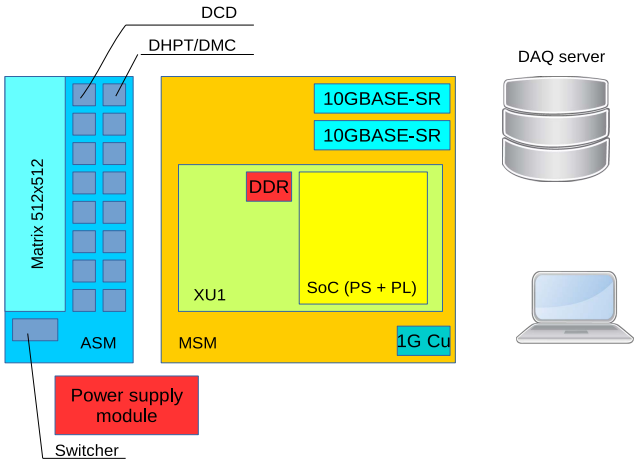


EDET 80k.
Firmware & software.

PS. Peripherals drivers.

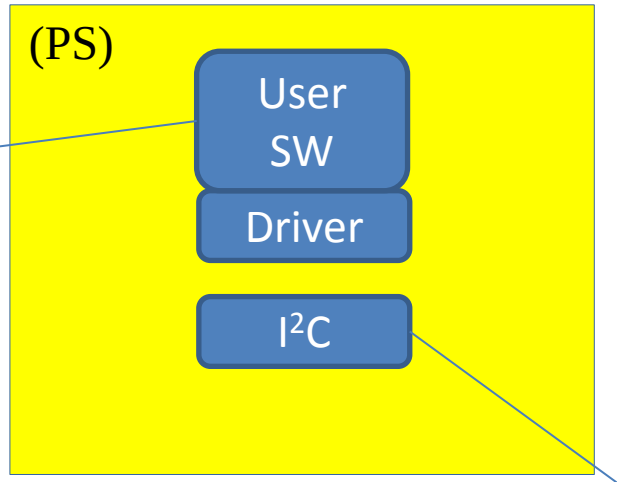
- 10G configuration (IFG, MAC, IP address, port N)
- GTH status & config
- MMCM config
- JTAG





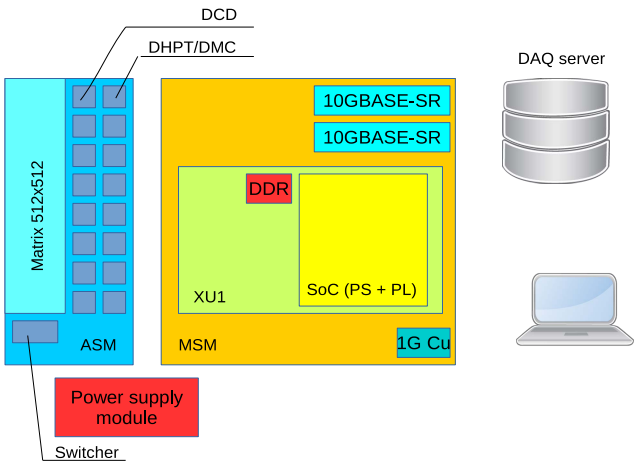
EDET 80k.
Firmware & software.

PS. Low-level housekeeping (I2C subsystem, statistics gathering)



Power-up sequence,
Emergency shutdown scenario

PSP & PPM



The values are calibrated

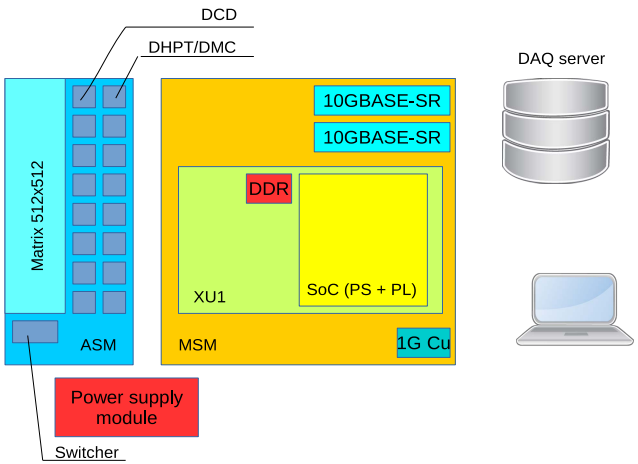
EDET 80k.
Firmware & software.

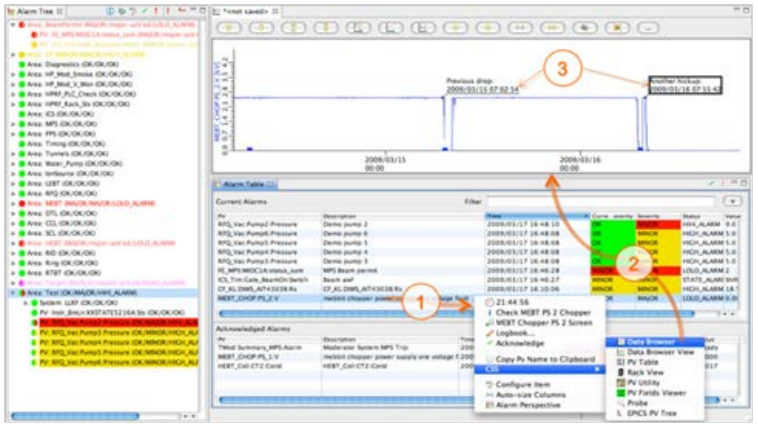
PS. Connection to PC: ssh, sshfs, telnet

EDET 80k.
Firmware & software.

PC. Overview, main tasks.

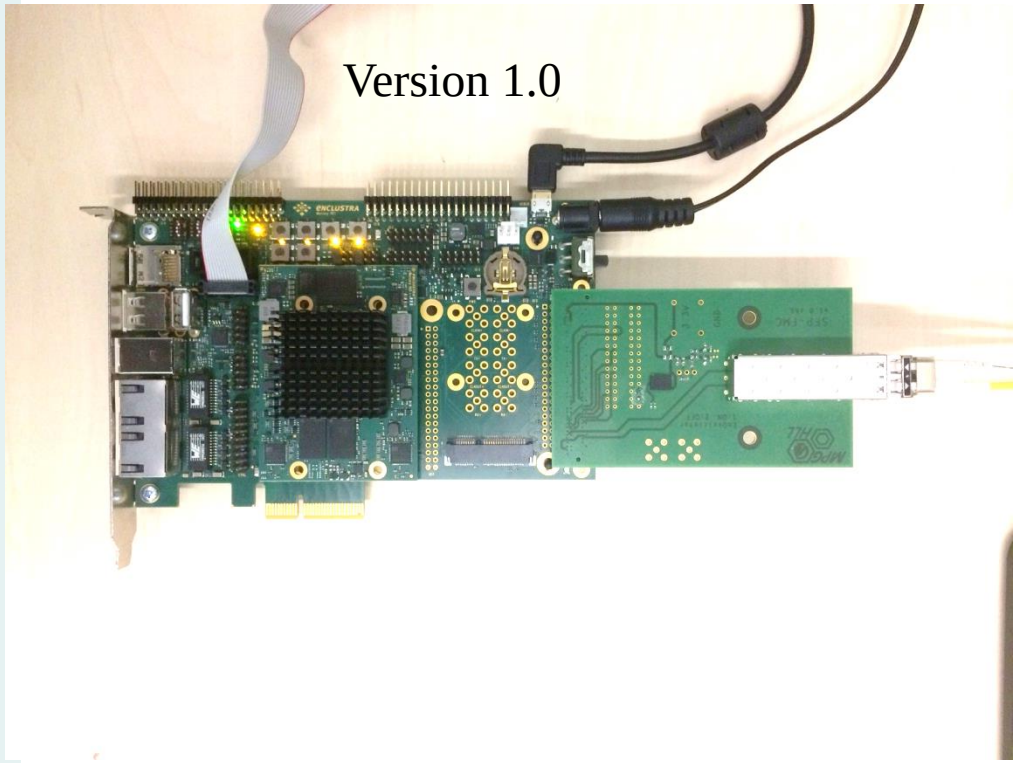
- Boot over network support. DHCP, TFTP services.
- Current device state depiction. Includes housekeeping data and picture at full resolution but decimated framerate.
- Device state logging for failures backtracking.
- Hardware configuration user-friendly interface which wraps all the instruments to the one. It includes housekeeping, JTAG, PL peripherals and PL itself.

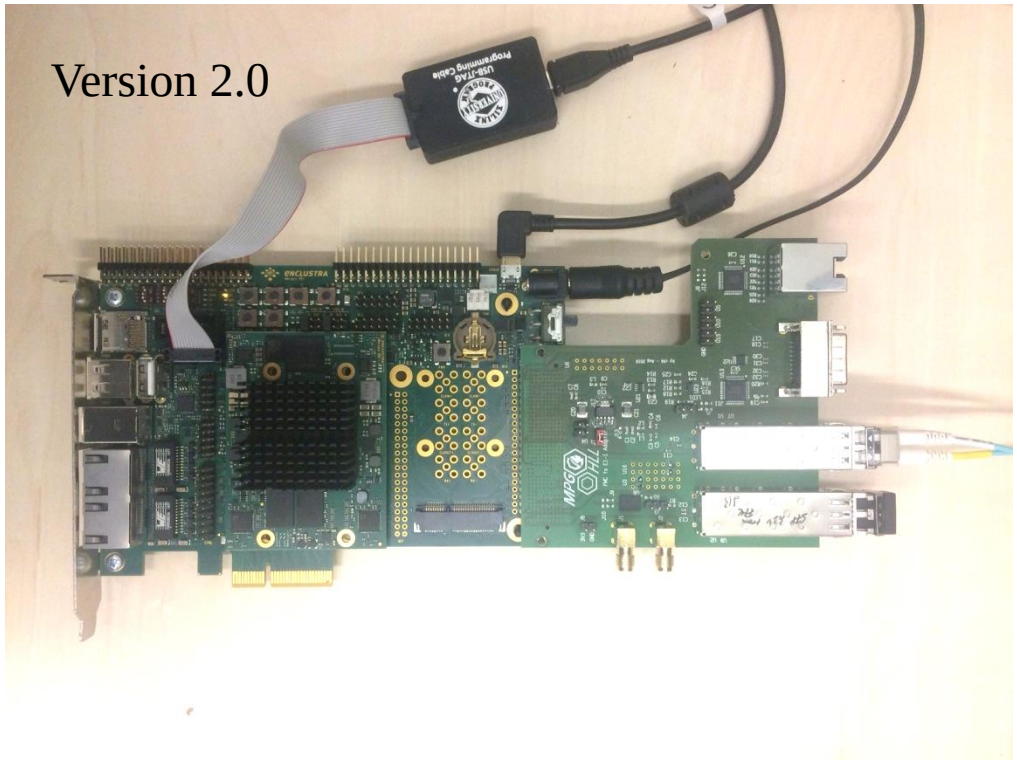




EDET 80k.
Firmware & software.

Test setup. Pictures.





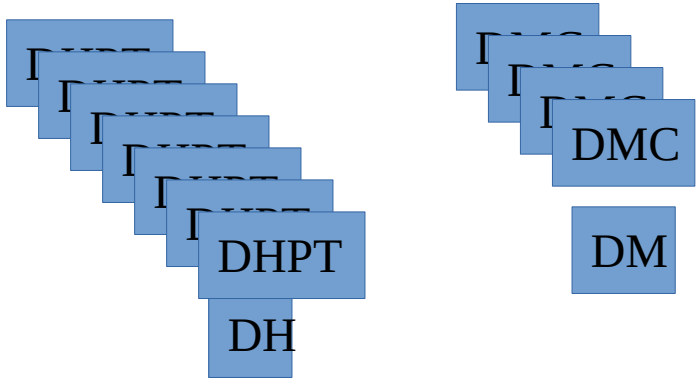
Version 3.0 is 2.0 + E1 hybrid (DHPT, DCD, SWb, 128x64 sensor)

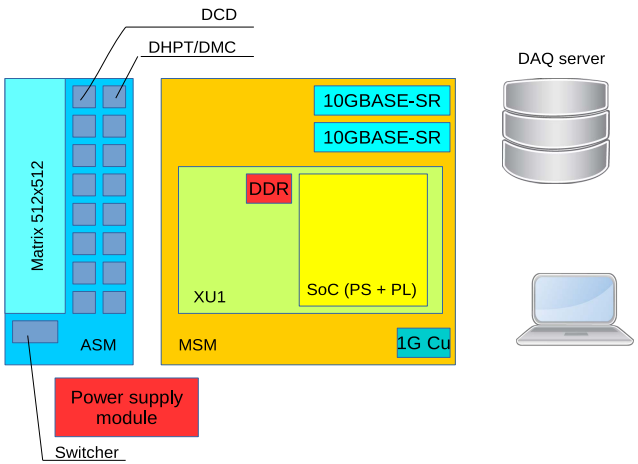
EDET 80k.
Firmware & software.

DAQ server.

Disks write speed: 1.2 GB/s

= 7,5 DHPT = 4.7 DMCs





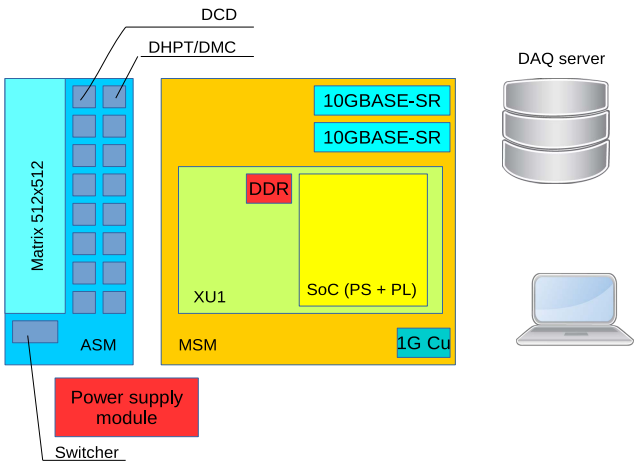
EDET 80k.
Firmware & software.

Numbers

Sensor size: 128x64 = 8192 pixels
GCK freq.: 78,125 MHz
Aurora link rate: 1,56 Gb/s
Aurora payload rate 1,25 Gb/s = 149 MB/s
DHPT main memory size = 128kB

$$\frac{Aurora\ payload\ rate}{Sensor\ size} = 19\,073 \text{ frames/s}$$

$$\frac{DHPT\ main\ memory\ size}{Sensor\ size} = 16 \text{ frames}$$



Even FoT (frames over time) distribution vs burst-mode of 16 frames?

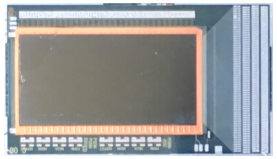
What we want to achieve?

Unlike Belle2 which is particle detector, main purpose for the EDet is TEM, so zero suppressed mode is not an option

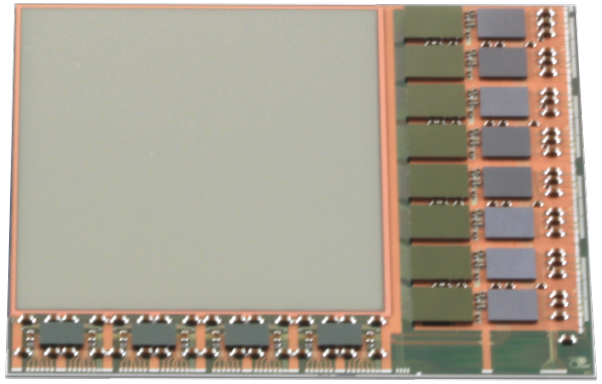
We are building data acquisition system to readout matrix in fullframe mode

Two sensor configurations*

– Prototype 32 lines x 256 channels sensor
– One DHPT
– One DCD
– One Switcher

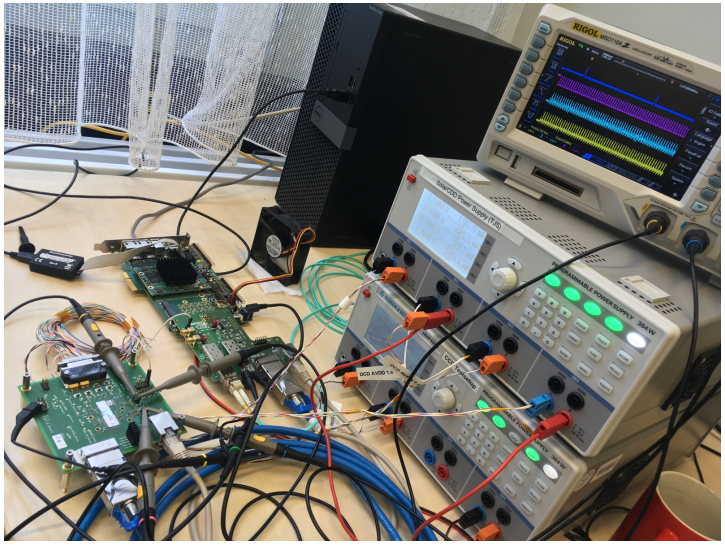


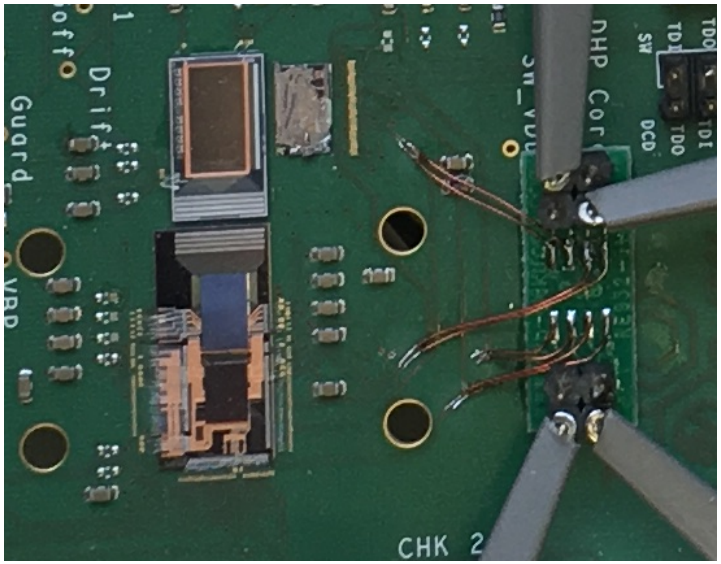
– Full quadrant 512x512 sensor
– 8x DHPTs
– 8x DCDs
– 4x Switchers



* FPA not shown for clarity. We benefit simple hardware multiplication from modular design

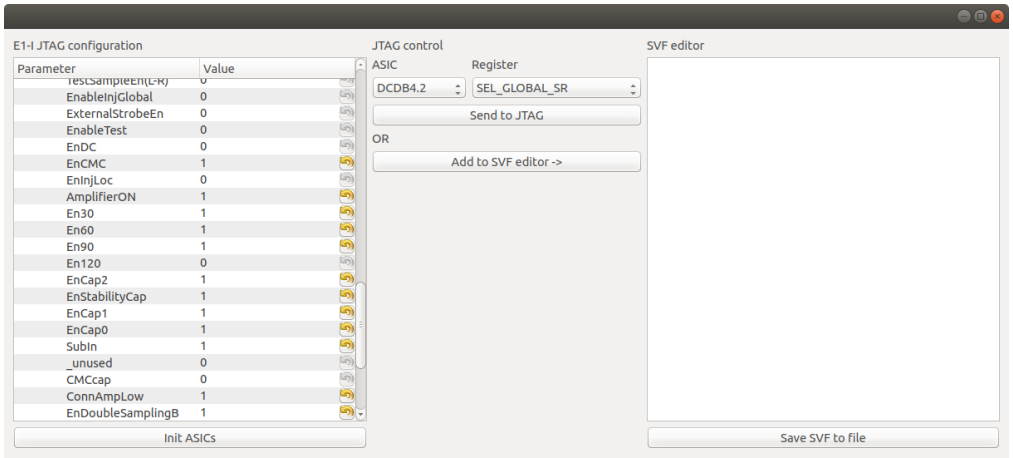
Current status





Test setup status:

- Uses E1-I Hybrid board with fully functional DCD and DHPT for electrical tests and debug
- Switcher and Sensor are non-functional and disconnected from DCD and DHPT
- Switcher sequence loaded to DHPT memory and monitored with oscilloscope
- Aurora link is up and running with eye diagram wide open
- DHPT and DCD controlled using register-level GUI App
- Data monitored in real-time through low-speed housekeeping channel: ~5fps



Important milestone: DCD data

En30 = 0	En30 = 1	En30 = 0	En30 = 1	En30 = 0	En30 = 1	En30 = 0	En30 = 1
En60 = 0	En60 = 0	En60 = 1	En60 = 1	En60 = 0	En60 = 0	En60 = 1	En60 = 1
En90 = 0	En90 = 0	En90 = 0	En90 = 0	En90 = 1	En90 = 1	En90 = 1	En90 = 1

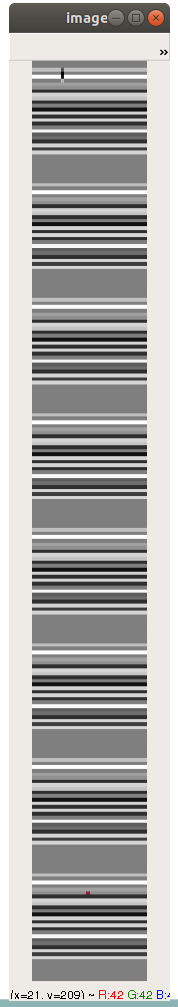
This is a ***live*** stream via real-time housekeeping channel

Pictures represent data with gain sweep low to high

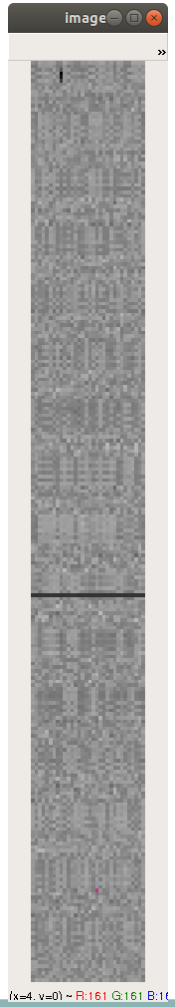
DCD data in electrical view: 32 rows x 256 ADC channels

Important milestone: DCD data

DCD test pattern



DCD data with high gain and one CMC bit enabled



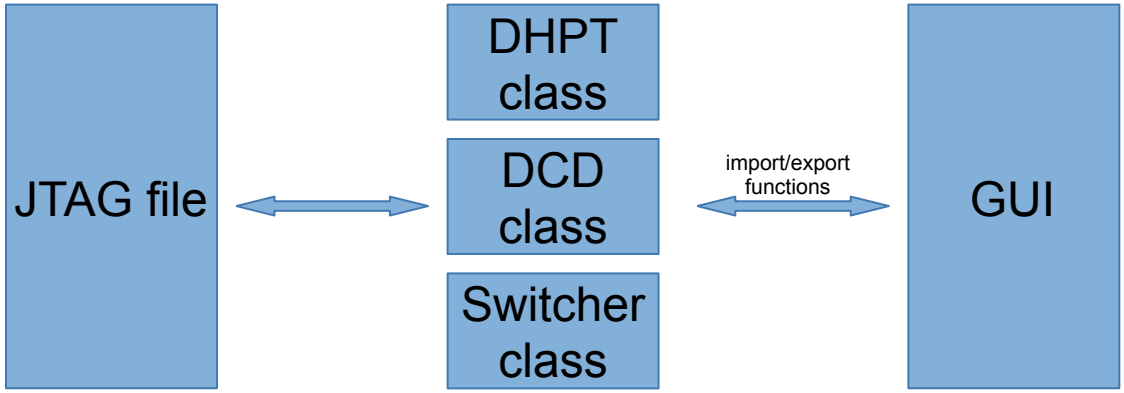
Data passed through complete DAQ chain to user Front-end

Prototype software scaling to ASM

JTAG control implemented using Python classes for each ASIC.

ASIC class has common methods to control registers, generate JTAG files and iterate with GUI

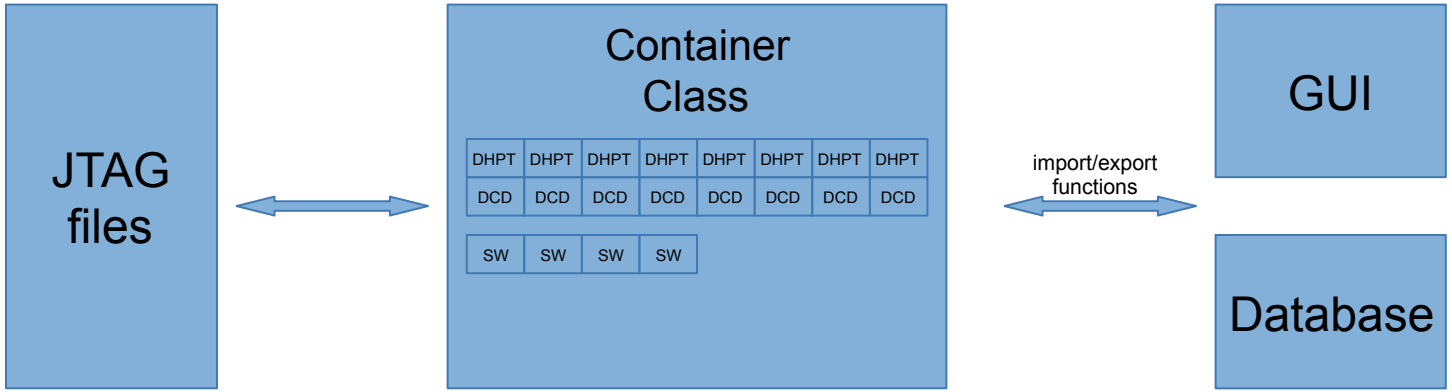
In addition to this, each particular class implements ASIC-specific operations, like memory uploads



Prototype software scaling to ASM

To scale this functionality to ASM assembly, container class is being developed to handle all instances of ASIC classes, maintain JTAG chains and provide uniform access to the ASICs.

Database to be implemented to support configuration management and operations logging.

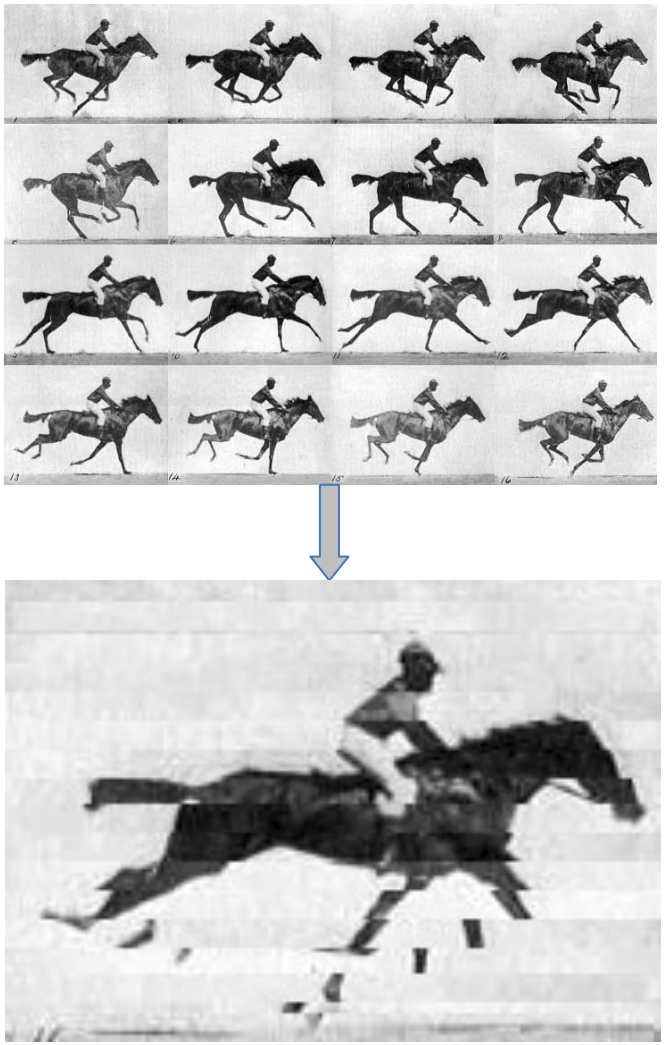


Like DMC physical layout and pinout provides drop-in replacement for DHPT, the same approach is used for software. This results in seamless DMC integration for ASMs to come by replacing DHPT class.

Todo list

- Head from electrical tests towards clean-room for physical measurements
- Develop user-friendly “you won’t be able to press wrong button” operator software
- Fork software and firmware to operate ASM assembly
- Development of DHPT FPA operating mode: Optimize DHPT settings for full frame readout of large quadrant device (see next slides)

DCD vs Aurora data rate



DHPT FPA operating mode:

DCD has 16x times higher data rate than Aurora link

This leads to the situation, where we have one frame from the chip made of 16 matrix scans, image overwritten 16 times per one Aurora transfer

Questions:

- DHPT FPA operating mode must avoid this situation
- Special sequence & settings required to "freeze" matrix
- Have 16x matrix exposure time to let Aurora send the data

For small matrix:

- Use 512 rows settings
- DHPT accumulates data from "dummy" 33-512 rows
- Aurora sends useful 1-32 rows

Not an option for 512x512 matrix: DHPT memory limited to 512 data rows

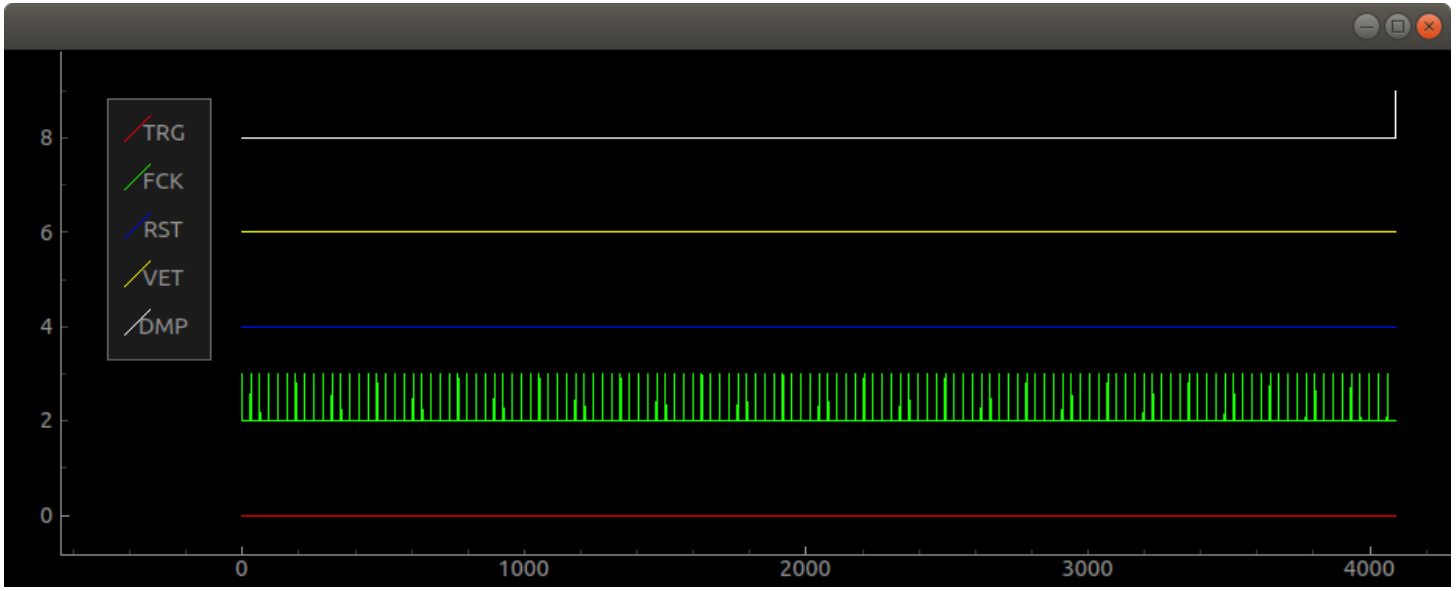
Input from DHPT experts required

Not an issue for DMC

DHPT trigger line

DHPT uses Trigger line to receive TRG, FCK, RST, VETO and MEM_DUMP commands

Sequence currently used at prototype setup:



- FCK every 32x8 GCK cycles
- MEM_DUMP once per 4096x8 GCK cycles (should be 512x8 for max frame rate, but ok for debug purposes)

Question:

- Do we need to use RST and TRG lines while in full frame readout mode?

EDET 80k.
Firmware & software.

Thank you.

Backup slides

DHPT settings

[CORE_REG]
reverse_bit_order = 0
double_out_bits = 0
how_long_align = 400
do_cc = 1
rand_gen_on = 0
trig_cont_en = 0 → 1
rand_gen_on_out_fifo = 0
offset_mem_active = 0
offset_des_dly = 8
offset_frame_sync_dly = 1534 → 5
offset_mem_disable = 0
Threshold = 5 → 255
cm_offset = 0
cm_use_overflow_bit = 0
cm_use_dcd_chan_mask = 0
cm_correction_en = 0
test_mode_en = 0
active_ped_mem = 1
pedestal_offset = 0
pedestal_subtraction = 1
disable_memory = 0
invert_input_values = 0
switch_sw_new_frame_veto = 0
mem_dump_start = 0
row2_sync_dcd_clk_dly = 3
frame_sync_proc_dly = 65
last_row_dump = 31
last_row = 31
latency = 0
chip_id = 0

[GLOBAL_REG]
iref_trimming = 8
idac_test = 0
idac_pll_ivco = 96
idac_pll_icp = 10
idac_pll_i50u = 40
idac_lvds_tx_iref = 150
idac_lvds_rx_iref = 100
idac_diode = 0
idac_dcd_rx_iref = 100
idac_cml_tx_ibiasdelay = 50
idac_cml_tx_biasd = 150
idac_cml_tx_bias = 50
ser_lfsr_rb = 0
pll_cml_dly_sel = 0
pll_cml_out_sel = 0
pll_des_clk_sel = 1
pll_ser_clk_sel = 0
pll_ser_clk_dly = 0
offset_dcd_dly^X = 7..8
dcd_rx_sdly^X = 10..11
dcd_clk_sdly = 0
dcd_clk_set30 = 1
dcd_clk_set12 = 1
dcd_clk_set06 = 1
frame_sync_dcd_dly = 0
dcd_row2_sync_dly = 0
dcd_cmos_clk_dly = 0
dcd_invert_tck_polarity = 0
dcd_invert_trst_polarity = 0
dcd_sync_en_frame_sync = 1
dcd_sync_en_row_sync = 1
dcd_sync_en_diff_clk = 1
dcd_sync_en_clk = 0
dcd_jtag_en_out = 1

[GLOBAL_REG]
tdo_sdly = 0
tdo_tx_set30 = 1
tdo_tx_set12 = 1
tdo_tx_set06 = 1
sw_frame_sdly = 0
sw_clk_sdly = 0
sw_gate_sdly = 0
sw_clear_sdly = 0
sw_tx_set30 = 1
sw_tx_set12 = 1
sw_tx_set06 = 1
sw_en_out = 1
offset_en_out = 1
top_ub_en_out = 0
pll_en_out = 0
tx_sel_clk = 0
pll_out_sel = 0
sw_new_frame_dly = 0
sw_gate_dly = 0
sw_clear_dly = 0
sw_clk_dly = 0
offset_dly^X_X = 0
row2_sync_dly = 0
frame_sync_dly = 0
ref_clk_div_dly = 0
global_reserved = 0

DCD settings

[SEL_GLOBAL_SR]
dummy = 0
dac0_ipaddout = 0
dac1_ifbpbias = 80
dac2_ipsource = 80
dac2a_ipsourcemiddle = 76
dac3_ipsource2 = 70
dac4_ipdel = 127
dac5_iinjpsignal = 0
dac6_ipdac = 0
dac7_itcp = 30
dac8_itcpl = 30
dac9_ipsourcecasc = 64
dac10_ifbncasc = 0
dac11_ifbref = 64
dac12_inmos = 120
dac13_itccasc = 0
dac14_vnsubin = 11
dac15_vnsubout = 0
dac16_vtcsfn = 60
dac17_vndel = 127
dac18_ipaddin = 0
dac19_refnwell = 64
dac20_iamppbias = 60
dac21_vpmos = 120
dac22_sigmirror0 = 0
dac23_sigmirror1 = 0
dac24_sigmirror2 = 0
enable_dcd_key = 10
enableneedletest = 0
enableshiftregtest = 0
qextra[12]_unused = 0
testload = 0
lvds_current = 1
lvds_threshold = 1

[SEL_GLOBAL_SR]
enresistivebiasgen = 2
vdc_pulldown = 0
vncm_pulldown = 1
band_gap_bias_gen = 0
npower_on_band_gap = 1
nen_mon_cap_filter = 1
q(11-8)_unused = 0
enablecmosbitclk = 0
amporadcglobal = 1
testinjen = 0
testsync = 0
testsampleen(l-r) = 0
enableinjglobal = 0
externalstrobeen = 0
enabletest = 0
endc = 0
encmc = 1
eninjloc = 0
amplifieron = 1
en30 = 1
en60 = 0
en90 = 0
en120 = 0
encap2 = 1
enstabilitycap = 1
encap1 = 1
encap0 = 1
subin = 1
_unused = 0
cmccap = 0
connamplow = 1
endoublesamplingb = 1
endoublesampling = 0

EDET 80k.
Firmware & software.

1. Using DHPT main memory for saving consecutive frames until no free space left and then disabling write operation until readout is finished (burst mode).

2. FSYNC resets write pointer. What happens with the pointer if no command is sent?

EDET 80k.
Firmware & software.

3. MEMDUMP resets read pointer. What does DHPT do after read pointer has already reached last_row_dump value but a new MEMDUMP command hasn’t arrived yet?(possibility to detect the state?)