cil, missusb, ddl and friends...

Alexey Filin,
OKA, IHEP, Protvino, September 29, 2011

Prologue

- The presentation describes some not presented before software components used to test/run/control OKA DAQ & trigger electronics, decode stored data, monitor beam

Cil (1)

- Crate interface library (cil) was developed in C to simplify access to read-out, front-end and trigger electronics
- There are 5 abstract classes:
  - cil_actl (abstract controller)
  - cil_abctl (abstract CAMAC/SUMMA branch ctl, inherits cil_actl)
  - cil_amctl (abstract MISS ctl, inherits cil_actl)
  - cil_ahiface (abstract host interface)
  - cil_ariface (abstract read-out interface)
- And 9 real derived classes to access hardware:
  - cil_bit3 (PCI host interface, VME ctl), cil_pq/cil_iq (PCI/ISA host interface, Q-bus ctl)
  - cil_v02 (VME device, SUMMA branch ctl), cil_cbd8210 (VME device, CAMAC branch ctl), cil_le20 (SUMMA ctl, Q-bus ctl)
  - cil_le51, cil_le83 (Q-bus device, MISS ctl)
  - cil_mu (MISS USB read-out interface)

Cil (2)

- 2 classes to abstract access to registers in crate electronics (can be mmaped or read/write accessed, depends on driver):
  - cil_register16 – access to 16 bit registers
  - cil_register32 – access to 32 bit registers
- Access to crate electronics is implemented with classes described above and is transparent. Real classes derived from the same parent are fully interchangeable (taking into account hw compatibility):
  - Any SUMMA module can be accessed with any branch controller
  - Any MISS module can be accessed with any MISS controller
  - Any MISS controller can be accessed with any Q-bus controller.
- Python bindings and aux modules are provided for the classes:
  - BranchController, BranchModule, Hiface, MISSController, MISSModule, Register16, Register32, Riface (abstract)
  - Bit3, CBD8210, ISAQbus, LE51, LE83, PCIQbus, V02 (real)

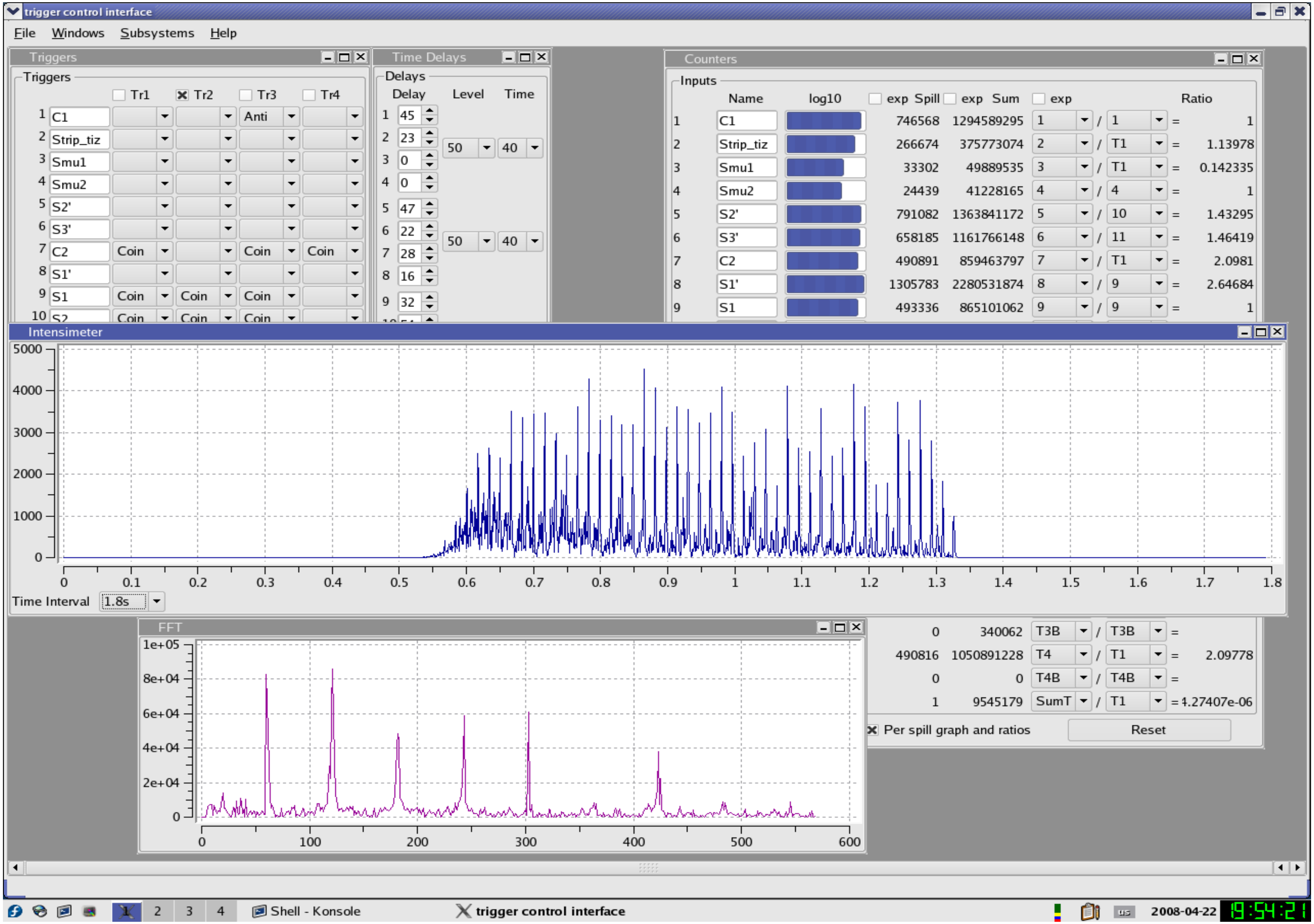
Cil (3)

- Nearly all SUMMA/CAMAC/MISS electronics modules used by OKA and managed with a computer interface are provided with support python modules:
  - CAMAC: Commutator, GAMSMotor, Generator, LEDGenerator, MOR, UNISI
  - SUMMA: D135, F133, IMT, LE57, LE75, LE90M, LE94, R8, TH, TL2_50
  - MISS: LE71, LE85
  - USB: MU
- Python interface speeds up significantly tests of crate electronics in comparison with C interface. Python interpretator can be used to operate with crate electronics interactively
- Scripts to work with crate electronics were developed with cil:
  - Scripts to work with separate modules: generator_ctl.py, le71_ctl.py, le83_ctl.py, le85_ctl.py, le85_ctl.py, le94_ctl.py, led_generator_ctl.py, mor_ctl.py, mu_ctl.py
  - Scripts to work with subsystems: qdc_ctl.sh, trig_ctl.sh

trigctl (1)

- A program to manage trigger electronics trigctl was written in Python with cil and used successfully in 7 runs for 4 years:
  - Web interface provides access to stored scalers, intensimeter, trigger solutions, delays, thresholds with web-browser
  - Intensity, spectrum (with FFT) and intensity distribution histogram from trigctl are used by beam shifts to control beam parameters
  - MySQL database is used to store history of control information
  - All info is passed to DAQ readout process each spill with shared memory and stored into files with real data for off-line analysis
  - Directory tree used to store scalers, intensimeter, spectrum in files should be replaced with database back-end. There were problems with file system due to enormous number of files (~10^6)
  - Trigctl should be ported to Qt4 (Qt3 is used now) to run it on Fedora 14

trigctl (2)



Triggers

		□ Tr1	☒ Tr2	□ Tr3	□ Tr4
1	C1			Anti	
2	Strip_tiz				
3	Smu1				
4	Smu2				
5	S2'				
6	S3'				
7	C2	Coin		Coin	Coin
8	S1'				
9	S1	Coin	Coin	Coin	
10	S2	Coin	Coin	Coin	

Delays

	Delay	Level	Time
1	45		
2	23	50	40
3	0		
4	0		
5	47		
6	22	50	40
7	28		
8	16		
9	32		

Inputs

	Name	log10	□ exp Spill	□ exp Sum	□ exp				Ratio
1	C1		746568	1294589295	1	/	1	=	1
2	Strip_tiz		266674	375773074	2	/	T1	=	1.13978
3	Smu1		33302	49889535	3	/	T1	=	0.142335
4	Smu2		24439	41228165	4	/	4	=	1
5	S2'		791082	1363841172	5	/	10	=	1.43295
6	S3'		658185	1161766148	6	/	11	=	1.46419
7	C2		490891	859463797	7	/	T1	=	2.0981
8	S1'		1305783	2280531874	8	/	9	=	2.64684
9	S1		493336	865101062	9	/	9	=	1
			0	340062	T3B	/	T3B	=	
			490816	1050891228	T4	/	T1	=	2.09778
			0	0	T4B	/	T4B	=	
			1	9545179	SumT	/	T1	=	4.27407e-06

☒ Per spill graph and ratios   Reset

Time Interval 1.8s

2008-04-22 19:54:21

missusb

- LE94 uses CYPRESS EZ-USB SX2 high-speed USB interface device (CY7C68001) to transfer data to front-end host with one bulk-in endpoint
- Missusb provides:
    - Synchronous (with read) data input
    - Asynchronous (with ioctl) data input from some devices simultaneously (work mode in DAQ)
    - Run time configuration of device sync transfers (timeout, buffer size) and async transfers (memory allocator type, chunk size, number of chunks)
    - Open by one process only to prevent multi-process access to the same device
    - Thread safe device access (read/configuration)
    - Linux device name with USB DeviceID to identify device and crates attached to it. USB is a hot-plug bus, unique static device id is a must for DAQ reconfiguration

Ddl (1)

- Basic structure of ddl is 5 level tree of decoders mapping given configuration of DAQ hardware and software (levels are ordered from tree root to leaves):
  - Event builder host (superevent)
  - Front-end host (event)
  - Equipment
  - Read-out module
  - Front-end electronics module
- Configuration can be performed with hard-coding, configurator with interface to Kurshetsov database and configurator with interface to DAQ front-end database [to be done]
- Each event type (physics, LED, PED) is configured with personal decoder tree

Ddl (2)

- Ddl is based on zero-copy approach (http://en.wikipedia.org/wiki/Zero-copy):
    - "Zero-copy" describes computer operations in which the CPU does not perform the task of copying data from one memory area to another.
    - Zero-copy protocols are especially important for high-speed networks in which the capacity of a network link approaches or exceeds the CPU's processing capacity. In such a case the CPU spends nearly all of its time copying transferred data, and thus becomes a bottleneck which limits the communication rate to below the link's capacity.
- Decoding is performed in two steps repeated recursively:
    - Decoder processes data block and stores pointers to found header, hits, trailer and error words (specific for each decoder, some can be absent)
    - Decoder identifies subdecoders and passes data blocks to the subdecoders for decoding

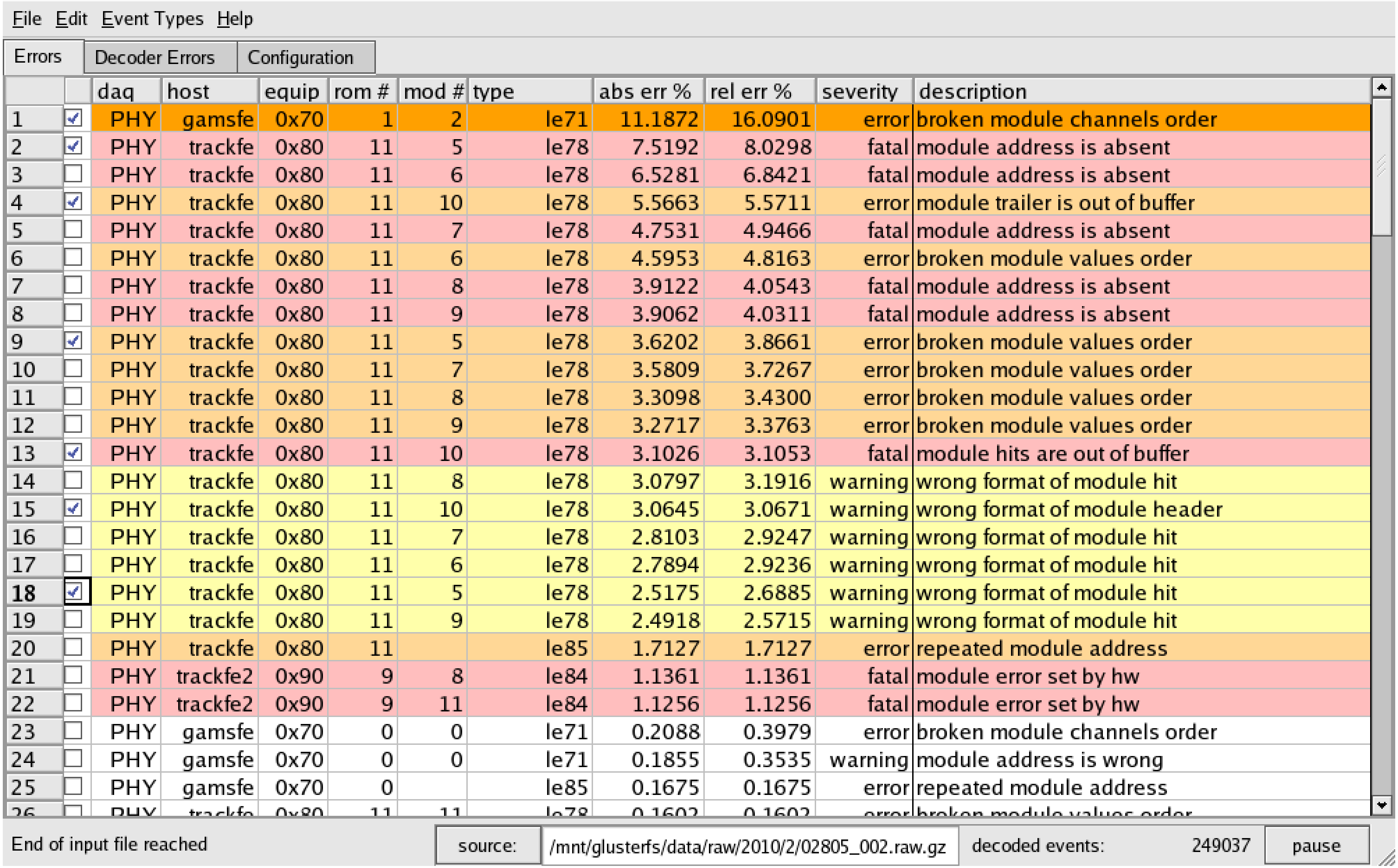
Ddl (3)

- There are 5 abstract decoder classes:
  - ddl_adec (base parent for all decoders)
  - ddl_amod, ddl_arom, ddl_autorom, ddl_sdrrom
- And 19 real decoder classes:
  - 3 upper levels: ddl_daq, ddl_host, ddl_equip
  - Read-out modules: ddl_le83, ddl_le85, ddl_ledbufrom, ddl_pedbufrom, ddl_rawbufrom,
  - Front-end modules: ddl_le69, ddl_le71, ddl_le71led, ddl_le71_ped, ddl_le76, ddl_le78, ddl_le79, ddl_le84, ddl_le84nt, ddl_le95, ddl_trigctl
- Each decoder contains mask of found errors and array of pointers to words in decoded buffer where errors were found:
  - Smart repairing of data for found errors can be implemented
  - Format analyser for data tests and electronics development can be implemented to simplify investigation of hex dumps

Dem (1)

- Decoding error monitor (dem) was developed to control operation of DAQ electronics on-line and provide full error statistics off-line. Dem relies deeply on ddl and influenced ddl design. Dem provides:
  - List of decoders sorted by number of errors, each record provides:
    - Decoder id (DAQ name, host name, equipment id, read-out id, front-end id)
    - Decoder type
    - Absolute & relative percent of error events for given error severity
    - Error severity, error description
  - List of all decoding errors for each decoder each record provides:
    - Number, absolute & relative percent of error events for each decoder error
  - Size of data decoded by each decoder:
    - Number, percent of events
    - Accumulated, minimum, average, maximum data length
  - Way to show/hide errors for any decoder (and event type) and its decoder sub-tree

Dem (2)



		daq	host	equip	rom #	mod #	type	abs err %	rel err %	severity	description
1	☑	PHY	gamsfe	0x70	1	2	le71	11.1872	16.0901	error	broken module channels order
2	☑	PHY	trackfe	0x80	11	5	le78	7.5192	8.0298	fatal	module address is absent
3	□	PHY	trackfe	0x80	11	6	le78	6.5281	6.8421	fatal	module address is absent
4	☑	PHY	trackfe	0x80	11	10	le78	5.5663	5.5711	error	module trailer is out of buffer
5	□	PHY	trackfe	0x80	11	7	le78	4.7531	4.9466	fatal	module address is absent
6	□	PHY	trackfe	0x80	11	6	le78	4.5953	4.8163	error	broken module values order
7	□	PHY	trackfe	0x80	11	8	le78	3.9122	4.0543	fatal	module address is absent
8	□	PHY	trackfe	0x80	11	9	le78	3.9062	4.0311	fatal	module address is absent
9	☑	PHY	trackfe	0x80	11	5	le78	3.6202	3.8661	error	broken module values order
10	□	PHY	trackfe	0x80	11	7	le78	3.5809	3.7267	error	broken module values order
11	□	PHY	trackfe	0x80	11	8	le78	3.3098	3.4300	error	broken module values order
12	□	PHY	trackfe	0x80	11	9	le78	3.2717	3.3763	error	broken module values order
13	☑	PHY	trackfe	0x80	11	10	le78	3.1026	3.1053	fatal	module hits are out of buffer
14	□	PHY	trackfe	0x80	11	8	le78	3.0797	3.1916	warning	wrong format of module hit
15	☑	PHY	trackfe	0x80	11	10	le78	3.0645	3.0671	warning	wrong format of module header
16	□	PHY	trackfe	0x80	11	7	le78	2.8103	2.9247	warning	wrong format of module hit
17	□	PHY	trackfe	0x80	11	6	le78	2.7894	2.9236	warning	wrong format of module hit
18	☑	PHY	trackfe	0x80	11	5	le78	2.5175	2.6885	warning	wrong format of module hit
19	□	PHY	trackfe	0x80	11	9	le78	2.4918	2.5715	warning	wrong format of module hit
20	□	PHY	trackfe	0x80	11		le85	1.7127	1.7127	error	repeated module address
21	□	PHY	trackfe2	0x90	9	8	le84	1.1361	1.1361	fatal	module error set by hw
22	□	PHY	trackfe2	0x90	9	11	le84	1.1256	1.1256	fatal	module error set by hw
23	□	PHY	gamsfe	0x70	0	0	le71	0.2088	0.3979	error	broken module channels order
24	□	PHY	gamsfe	0x70	0	0	le71	0.1855	0.3535	warning	module address is wrong
25	□	PHY	gamsfe	0x70	0		le85	0.1675	0.1675	error	repeated module address
26	□	PHY	trackfe	0x80	11	11	le78	0.1602	0.1602	error	broken module values order

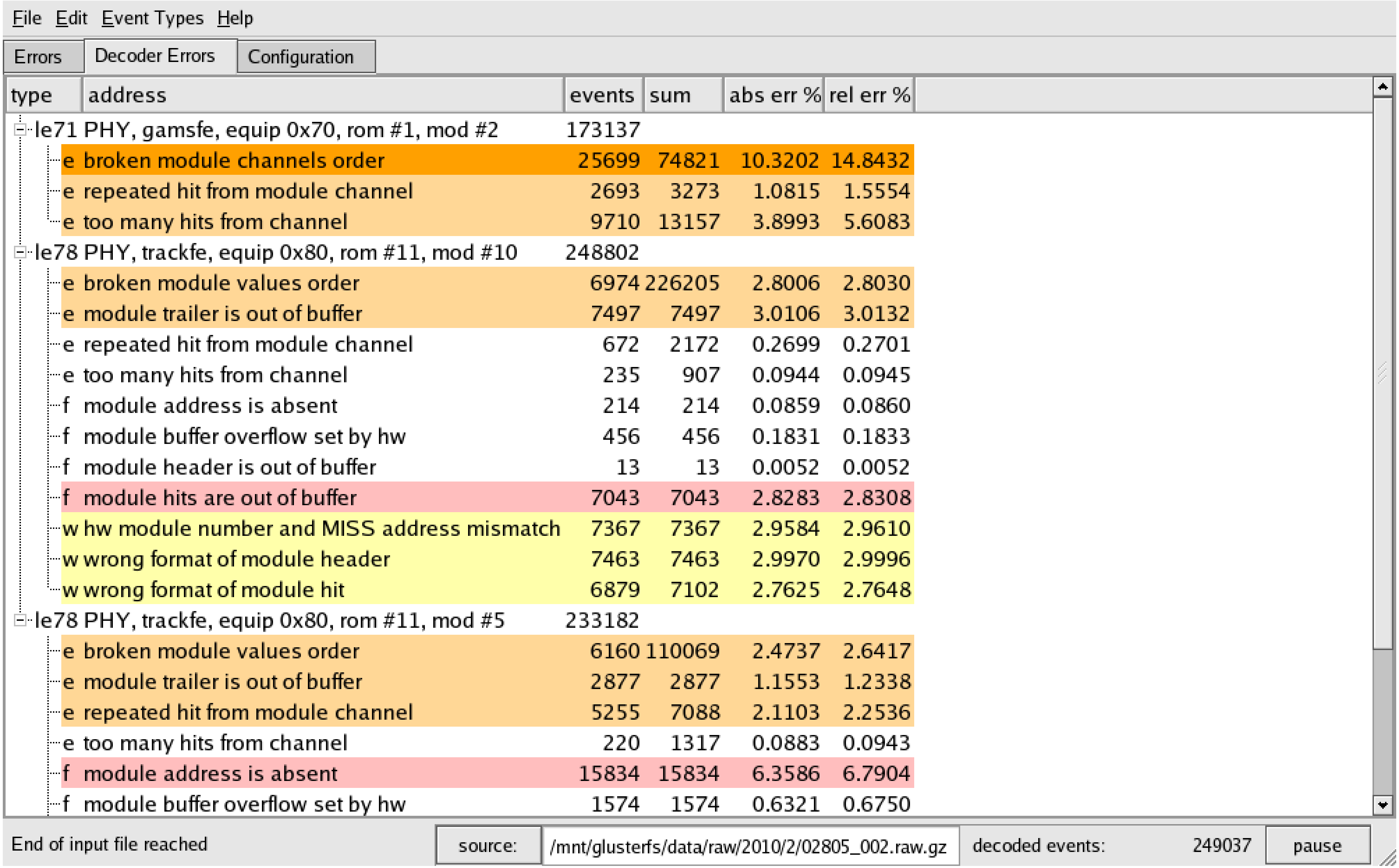
End of input file reached

source: /mnt/glusterfs/data/raw/2010/2/02805_002.raw.gz

decoded events: 249037

pause

Dem (3)



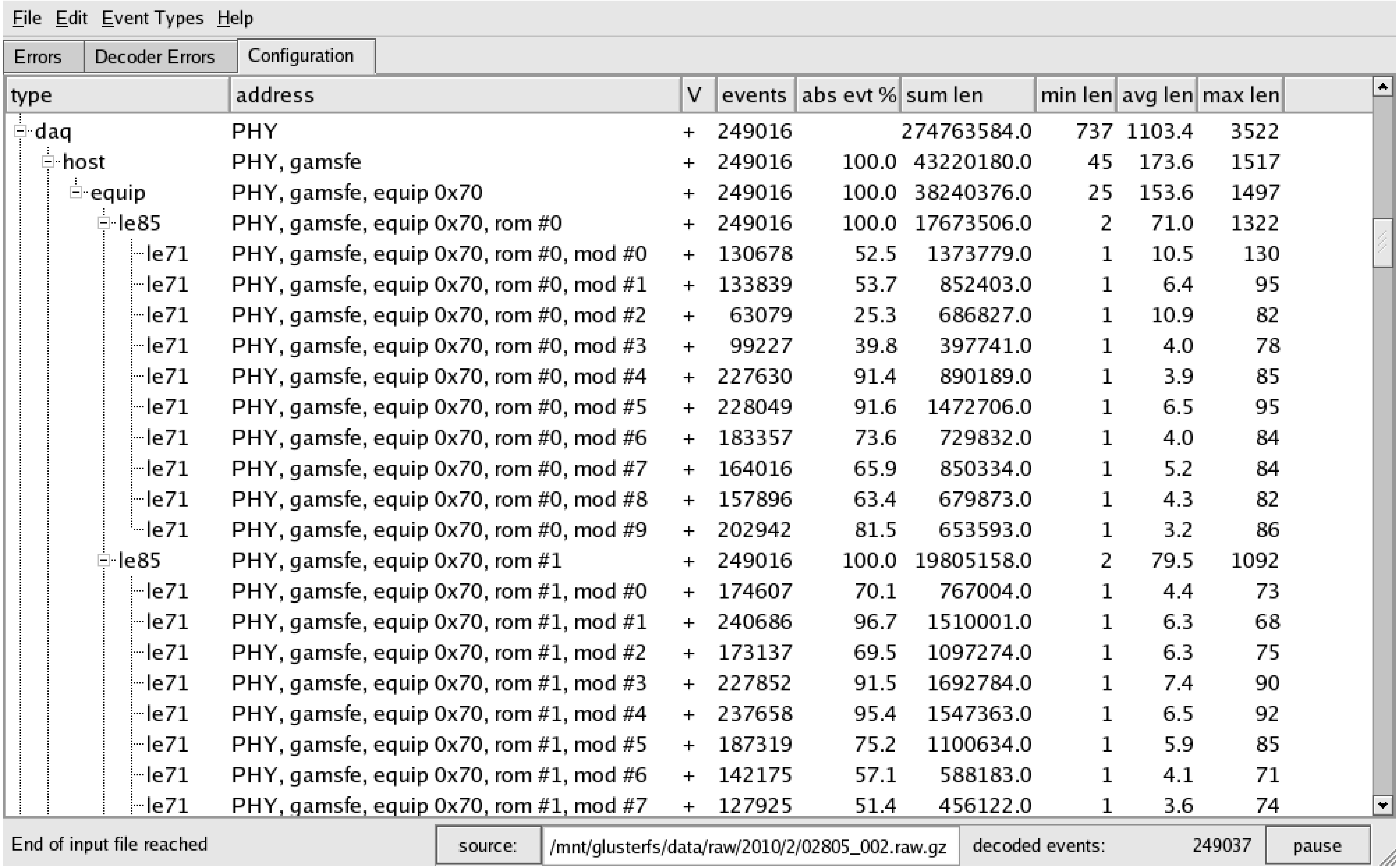
type	address	events	sum	abs err %	rel err %
le71	PHY, gamsfe, equip 0x70, rom #1, mod #2	173137			
e	broken module channels order	25699	74821	10.3202	14.8432
e	repeated hit from module channel	2693	3273	1.0815	1.5554
e	too many hits from channel	9710	13157	3.8993	5.6083
le78	PHY, trackfe, equip 0x80, rom #11, mod #10	248802			
e	broken module values order	6974	226205	2.8006	2.8030
e	module trailer is out of buffer	7497	7497	3.0106	3.0132
e	repeated hit from module channel	672	2172	0.2699	0.2701
e	too many hits from channel	235	907	0.0944	0.0945
f	module address is absent	214	214	0.0859	0.0860
f	module buffer overflow set by hw	456	456	0.1831	0.1833
f	module header is out of buffer	13	13	0.0052	0.0052
f	module hits are out of buffer	7043	7043	2.8283	2.8308
w	hw module number and MISS address mismatch	7367	7367	2.9584	2.9610
w	wrong format of module header	7463	7463	2.9970	2.9996
w	wrong format of module hit	6879	7102	2.7625	2.7648
le78	PHY, trackfe, equip 0x80, rom #11, mod #5	233182			
e	broken module values order	6160	110069	2.4737	2.6417
e	module trailer is out of buffer	2877	2877	1.1553	1.2338
e	repeated hit from module channel	5255	7088	2.1103	2.2536
e	too many hits from channel	220	1317	0.0883	0.0943
f	module address is absent	15834	15834	6.3586	6.7904
f	module buffer overflow set by hw	1574	1574	0.6321	0.6750

End of input file reached

source: /mnt/glusterfs/data/raw/2010/2/02805_002.raw.gz

decoded events: 249037

Dem (4)



type	address	V	events	abs evt %	sum len	min len	avg len	max len
daq	PHY	+	249016		274763584.0	737	1103.4	3522
host	PHY, gamsfe	+	249016	100.0	43220180.0	45	173.6	1517
equip	PHY, gamsfe, equip 0x70	+	249016	100.0	38240376.0	25	153.6	1497
le85	PHY, gamsfe, equip 0x70, rom #0	+	249016	100.0	17673506.0	2	71.0	1322
le71	PHY, gamsfe, equip 0x70, rom #0, mod #0	+	130678	52.5	1373779.0	1	10.5	130
le71	PHY, gamsfe, equip 0x70, rom #0, mod #1	+	133839	53.7	852403.0	1	6.4	95
le71	PHY, gamsfe, equip 0x70, rom #0, mod #2	+	63079	25.3	686827.0	1	10.9	82
le71	PHY, gamsfe, equip 0x70, rom #0, mod #3	+	99227	39.8	397741.0	1	4.0	78
le71	PHY, gamsfe, equip 0x70, rom #0, mod #4	+	227630	91.4	890189.0	1	3.9	85
le71	PHY, gamsfe, equip 0x70, rom #0, mod #5	+	228049	91.6	1472706.0	1	6.5	95
le71	PHY, gamsfe, equip 0x70, rom #0, mod #6	+	183357	73.6	729832.0	1	4.0	84
le71	PHY, gamsfe, equip 0x70, rom #0, mod #7	+	164016	65.9	850334.0	1	5.2	84
le71	PHY, gamsfe, equip 0x70, rom #0, mod #8	+	157896	63.4	679873.0	1	4.3	82
le71	PHY, gamsfe, equip 0x70, rom #0, mod #9	+	202942	81.5	653593.0	1	3.2	86
le85	PHY, gamsfe, equip 0x70, rom #1	+	249016	100.0	19805158.0	2	79.5	1092
le71	PHY, gamsfe, equip 0x70, rom #1, mod #0	+	174607	70.1	767004.0	1	4.4	73
le71	PHY, gamsfe, equip 0x70, rom #1, mod #1	+	240686	96.7	1510001.0	1	6.3	68
le71	PHY, gamsfe, equip 0x70, rom #1, mod #2	+	173137	69.5	1097274.0	1	6.3	75
le71	PHY, gamsfe, equip 0x70, rom #1, mod #3	+	227852	91.5	1692784.0	1	7.4	90
le71	PHY, gamsfe, equip 0x70, rom #1, mod #4	+	237658	95.4	1547363.0	1	6.5	92
le71	PHY, gamsfe, equip 0x70, rom #1, mod #5	+	187319	75.2	1100634.0	1	5.9	85
le71	PHY, gamsfe, equip 0x70, rom #1, mod #6	+	142175	57.1	588183.0	1	4.1	71
le71	PHY, gamsfe, equip 0x70, rom #1, mod #7	+	127925	51.4	456122.0	1	3.6	74

End of input file reached

source: /mnt/glusterfs/data/raw/2010/2/02805_002.raw.gz

decoded events: 249037

Epilogue

- Some of described sw components could not be developed without sw by other developers:
  - VME device driver for Bit3 Model 617/618/620 by Enomoto Sanshiro (KEK, Japan)
  - PCI-Qbus driver by Oleg Solovianov (IHEP, Protvino, Russia)
  - Python binding generation: gccxml by Kitware, Inc. and ctypeslib by Thomas Heller
- The presentation can be get by http://www.oka.ihep.ru/Members/filin/files/cil_missusb_ddl_2011sep29.pdf/download