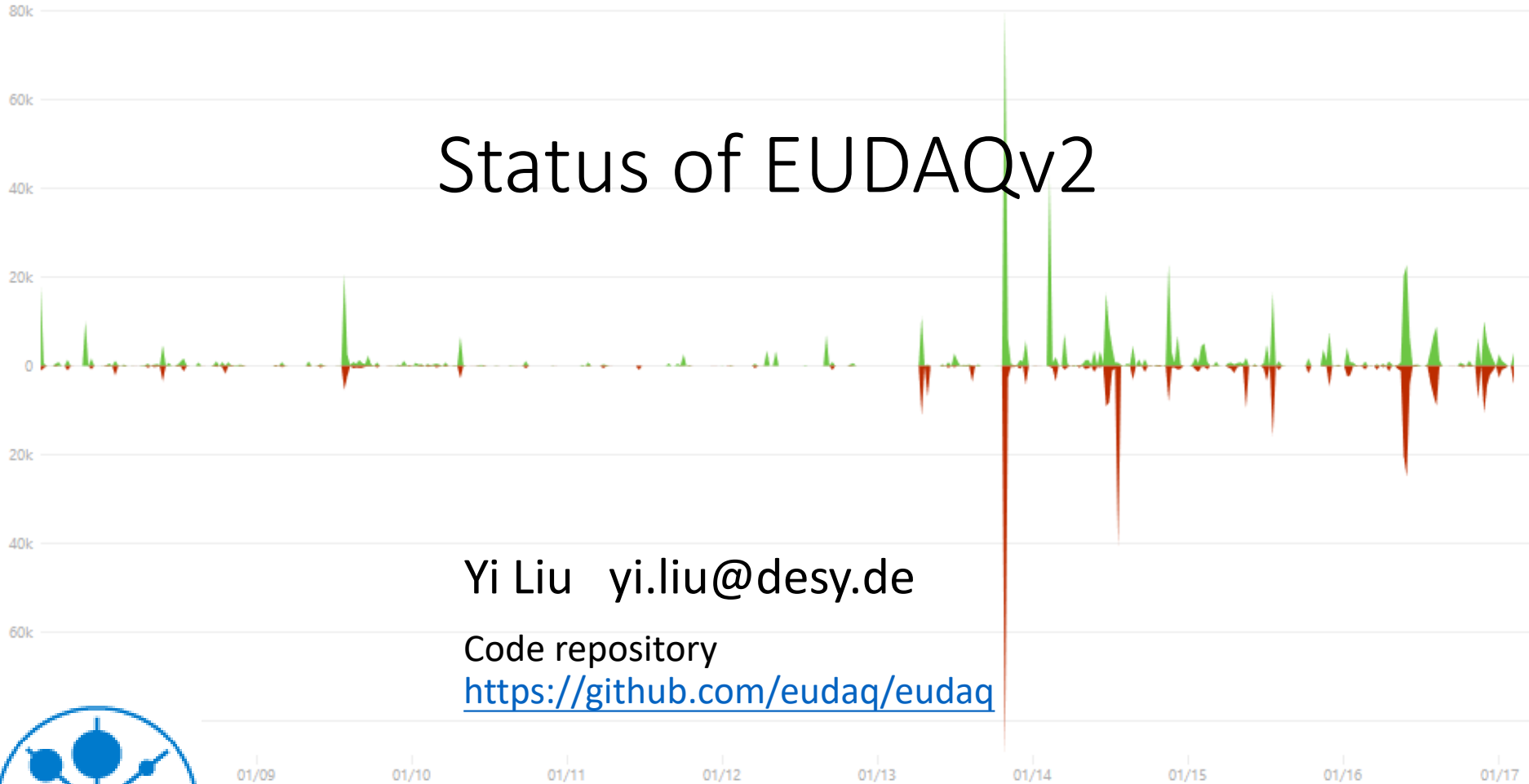


Status of EUDAQv2



Yi Liu yi.liu@desy.de

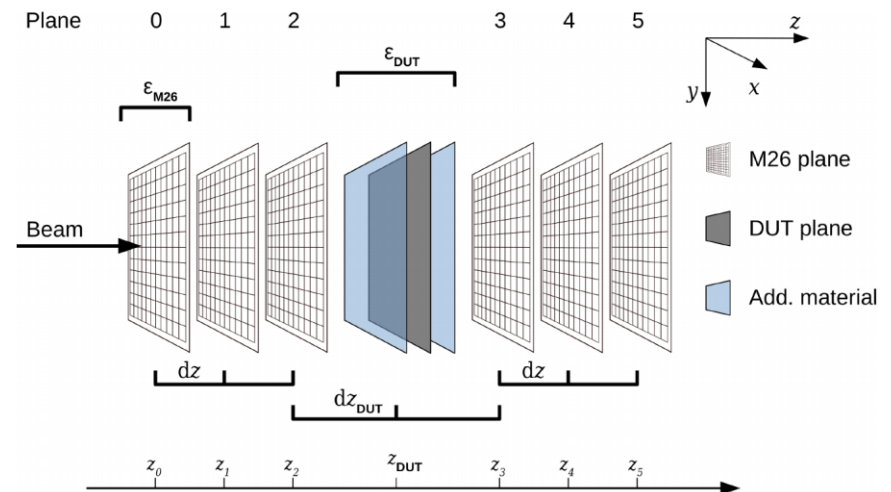
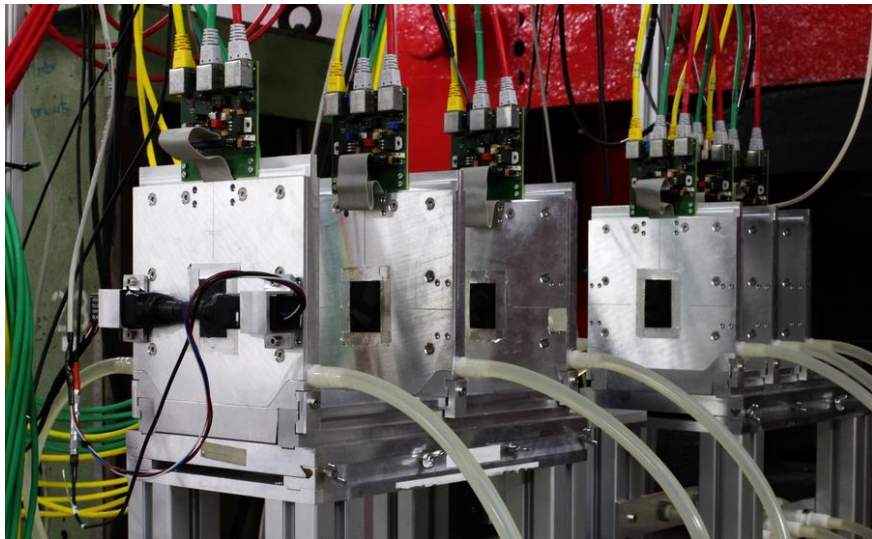
Code repository

<https://github.com/eudaq/eudaq>



History: EUDAQv1 on EUDET telescope

- EUDAQ is originally developed as a DAQ system for EUDET-type telescopes.
- Centralized controlling, logging.
- Interface for 3rd part users who do Testbeam under EUDET telescopes



Motivation: Extend its use case as common DAQ

- Key features to be a common DAQ

- Distributed data taking
- Central Control and configure interface.
- Data collector/builder and data converter
- GUI, Monitor
- Extendable
- Cross platform

In EUDAQ 1
√
√
√ *
√ *
√
√

- EUDAQ1 has almost all required key features to be a common DAQ, (*) except its data collector and Monitor was designed for EUDET hardware
- EUDAQ2 is a major version release. Let's take this chance to make a significant change of interface and improve to a nicer code.

Overview of EUDAQ components

EUDAQ2 consists of several components, which run online for data-taking or offline for data converting and quality analysis.

		EUDAQ2 Executable	Run mode	
EUDAQ	RunControl	Control all EUDAQ components	euRun, euCliRun	online
	Logger	Log the message	euLog, euCliLog	online
	Monitor	Display hit information online	OnlineMon(eudaq1)	Dual mode
	DataCollector	Merge the sub events.	euCliDataCollector	online
	Producer	Talk to device and send sub events	euCliProducer	online
	DataConverter	Convert data between different formats	euCliDataConverter	Dual mode

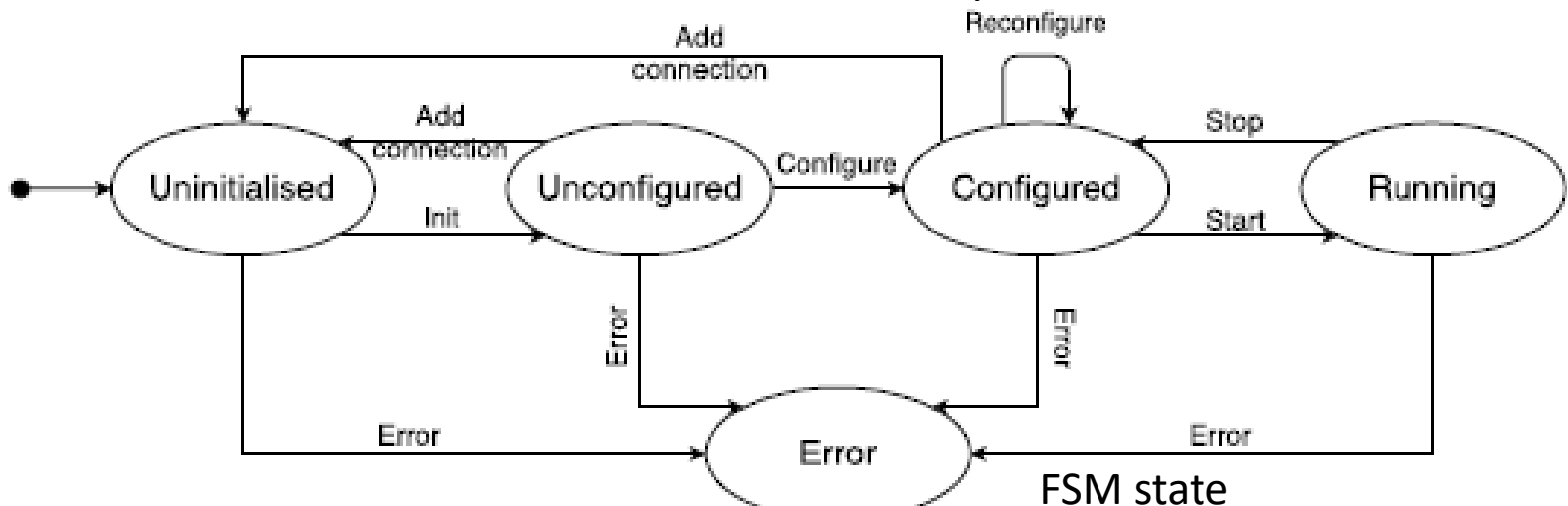
Component: RunControl

RunControl maintains a database about the address of clients and sends command to them.

- The Standard RunControl EUDAQ is enough for most user cases.

New in EUDAQ2

- QT GUI is decoupled from RunControl and EUDET-telescope
- User can reuse the GUI with their own RunControl without touch GUI code.
- Provide flexible to have dedicated RunControl to integrate with other DAQ system .
- The FSM states EUDAQ clients are checked by RunControl



Component: Producer

Producers are the binding part between a user DAQ and the central EUDAQ RunControl.

- The base Producer do all the common tasks for the derived Producer to simplify the integration.

New in EUDAQ2

- Unique launch executable application (euCliProducer)
- Runtime name
- FSM state is managed internally
- Configurable Data sending destination.

C++ Class UserProducer	functions deal with hardware device
	DoInitialise()
	DoConfigure()
	DoStartRun()
	DoStopRun()
	DoReset()
C++ Class Producer (base)	

Component: DataCollector

The Data Collector receives all the data streams from all the Producers, and combines them into a single stream.

- Capable to event data to different file formats by configuration.

New in EUDAQ2

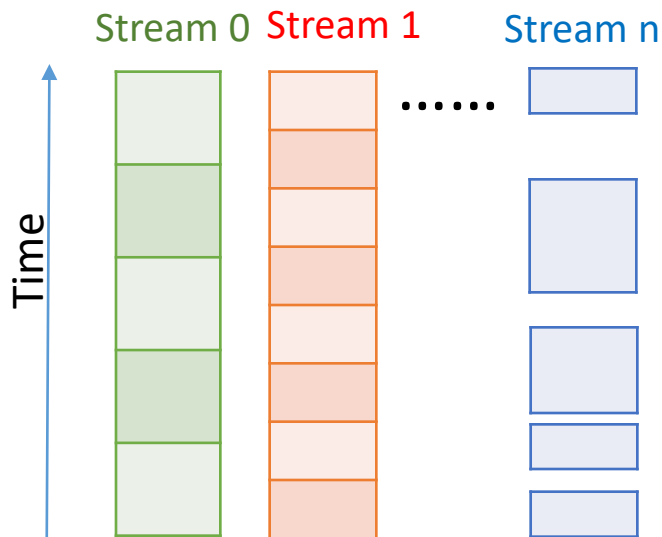
- Unique launch executable application (euCliDataCollector) and runtime naming.
- FSM state is managed internally.
- Event type independently.
- Multiple DataCollectors at one running setup

C++ Class	Cmd Functions	Data Functions
UserDataCollector	DoInitialise()	DoConnect
	DoConfigure()	DoDisconnect
	DoStartRun()	DoReceive
	DoStopRun()	
	DoReset()	

C++ Class
DataCollector
(base)

Component: DataCollector

Users are recommended to implement their DataCollector if there is a dedicated synchronization method to merge the data.



In generic case, sub events can arrive in any random time.

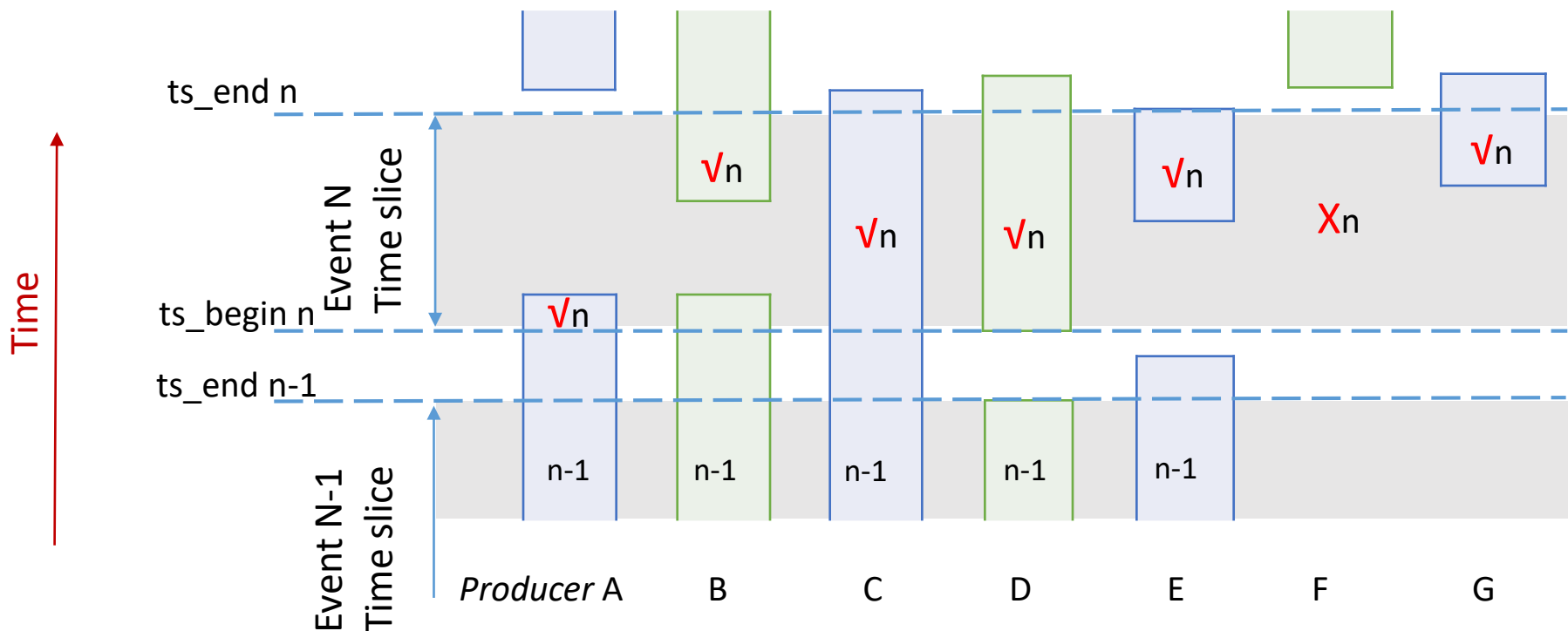


How to define a physical event?
How to define a valid trigger?
Which detector is the trigger device?

Generic DataCollectors for direct synchronization by timestamp and trigger number cases are provided. (next page)

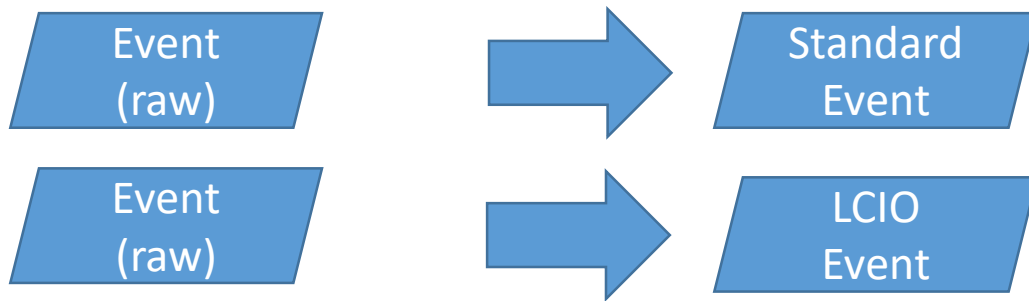
Example method for Event sync by Timestamp-pair

- Time length of incoming Event is flexible.
- Time slice of merged event is variable
- All Producers are equal to each others
- Empty incoming from Producer (F/G).

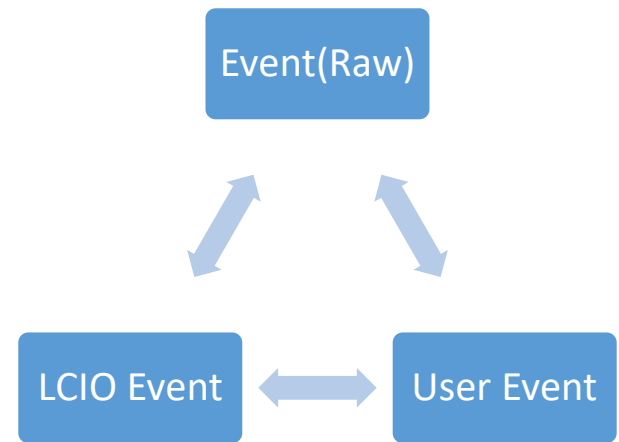


Component: DataConverter

- Modular designed for conversion between any kinds of data types.
- Conversion from EUDAQ Event to LCIO Event is native supported.
- DataConverter can be called online by DataCollector/Monitor if they want data in another format instead of EUDAQ native format.



User can extend the supported Event type of EUDAQ framework. (optional 2 ways converting)



Possible circle converting, if the correlated DataConverters among those event types are all implemented

Modular building / Runtime discovery

EUDAQ2 are build to 3 categories of libraries

Binary	Category	Source Code	Description
libeudaq_core	Core	main\lib\core	core library
libeudaq_lcio	Extension	main\lib\lcio	lcio extension library
libeudaq_std	Extension	main\lib\std	standard extension library
libeudaq_module_test	Module	main\module\test	test module library
libeudaq_module_example	Module	user\example\module	module library by example user











Table 2: Overview of EUDAQ libraries.

1. Core: the core library. It should be always build and installed.
2. Extension: optional features of EUDAQ (eg. support external data format).
Static linked core library.
3. Module: Dynamically discovered and loaded by EUDAQ core library at run-time.
user module.

The hardcode path dependence is removed.

Possible to distribute EUDAQ as binary package to End-User.

Documentation

		1. Remove all EUDET-type telescope paragraphs (framework only)	✓
+  Introduction		2. Restructure the order of chapter.	✓ drafted
+  Installation		3. Strip compilation/installation chapter.	✓ drafted
+  Run an Example Setup		4. Example Running setup.	✓ drafted
+  Integration with User Hardware		5. Introduce the base knowledge of Integration.	✓ drafted
+  Writing a Producer	}	6. Base example of user Integration.	○ Wip
+  Writing a Data Collector			
+  Writing a Data Converter			
 Writing a Monitor	}	7. Advanced Integration.	TODO
 Writing a RunControl			
+  Handling New Data Type			

Summary / Outlook

- API interface and core library are completed frozen.
- A successful run with CALICE/AHCAL hardware under testbeam
- Manual is on updating. (draft this month)
- Release with User Manual and Example codes. (alpha this month)
- Distribute by pre-compiled binary package (Ongoing by Andre Rummler)
- The EUDET telescopes in DESY will run EUDAQv2 by default (tested with Calice/AHCAL testbeam in Feb.)

TODO optional

- Support newTLU
- LCIO object serializing across applications (network, monitor)
- CMake configure file for user integration.
- Improve exception handling and threads safety
- XML config parser

(Nested blocks to describe the complex configuration case).

- Cascading DataCollector

(Event data from DataCollector to another DataCollector).

- Standalone Running

(Producer can run standalone without runcontrol/datacollector, and write data file to disk directly.)

- DataConverter and FileWriter working with ROOT TTree

Thanks to all the contributors of EUDAQ.

Thank you for your attention.

Definition of Event

Event maintains all common variables.

- Event Number is not always identical to Trigger Number. So both of EventNumber and Trigger Number are defined.
- The clock frequency which timestamp bases on.
- Timestamp is a pair of the clock points.
Timetamp_begin < Timesamp_end

Variable	Size/Type
Event Type	32Bits
EUDAQ version	32Bits
Event Flags	32Bits
Device Number	32Bits
Run Number	32Bits
Event Number	32Bits
Trigger Number	32Bits
Extend Word	32Bits
Clock numerator	32Bits
Clock denominator	32Bits
Timestamp Begin	64Bits
Timestamp End	64Bits
Description String	string
Tags	Pairs of strings