

1 Description of the setup

The current TWR daq system, which is displayed in figure 1 is a modified and enlarged version of the DAQ built in the Poleseason 2001/2002.

The DAQ consists of a master VME crate, several slave VME crates and a NIM crate holding the trigger logic. A VME-VME bridge establishes the connection between master and slave crates. One complete link consists of a master and a slave bridge module and an optical cable with two fibres.

Each slave VME crate holds 16 Transient Waveform Recorders TWR, one GPS-latch GP-SAMD and a VME converter board. The bridge module in the slave crate is acting as VME-master and controls the data flow in this crate.

The various parts and the readout process will be described on the following pages.

1.1 Transient Waveform Recorder

The Transient Waveform Recorder TWR build by SIS ¹⁾ is the heart of the recording system. It is a 12 bit flash adc sampling continuously the incoming signal at a rate of about 100 MHz in a ring buffer with a length of 1024 samples covering a total time window of 10.24 μ sec. After every external trigger pulse (event-trigger), which serves as a COMMON STOP TRIGGER the TWR switches to the next ring buffer and starts sampling for the next event. Every TWR has 8 channels and a total voltage range of 5V. In order to sample the PM peaks in the right way, the range of the TWR is adjusted to 1V to -4V.

For the optical channels of AMANDA the fast output of the ORBs and the ORMs are chosen as input signal, for the electrical channels the fast output of the SWAMPS is used. Every TWR has two memorybanks containing each 128 events. The TWRs are running in *Autobankswitchmode*, which reduces the deadtime of the system by separating the readout and the data-taking process. After starting the sampling, the first memory bank is filled. While reading out the first memory bank, the second memorybank is filled. As long as the readout time for one memory bank is not longer than the time for 128 triggers, the data-taking and the readout process are independent.

The TWR is equipped with a timestamp counter running with clock speed or with a down-scaled clock speed, which enables it to measure the time between events. With downscaling the timestamp clock speed, the sample clock speed is not affected. This feature of the TWR enables the TWR daq to find any synchronization errors between the TWRs by comparing the time differences between same events in different TWRs.

Since the number of waveforms with a PM peak is rather small it is essential while reading out the TWRs to detect these valid waveforms and store this information in a dedicated register. During the readout process only valid waveforms are read out. Every channel has an independent programmable threshold. A waveform is valid, if at least one of the 1024 values is below the threshold. Thus the thresholds can be conformed to the individual noise behaviour of each OM.

¹⁾Struck Innovative Systems - Hamburg

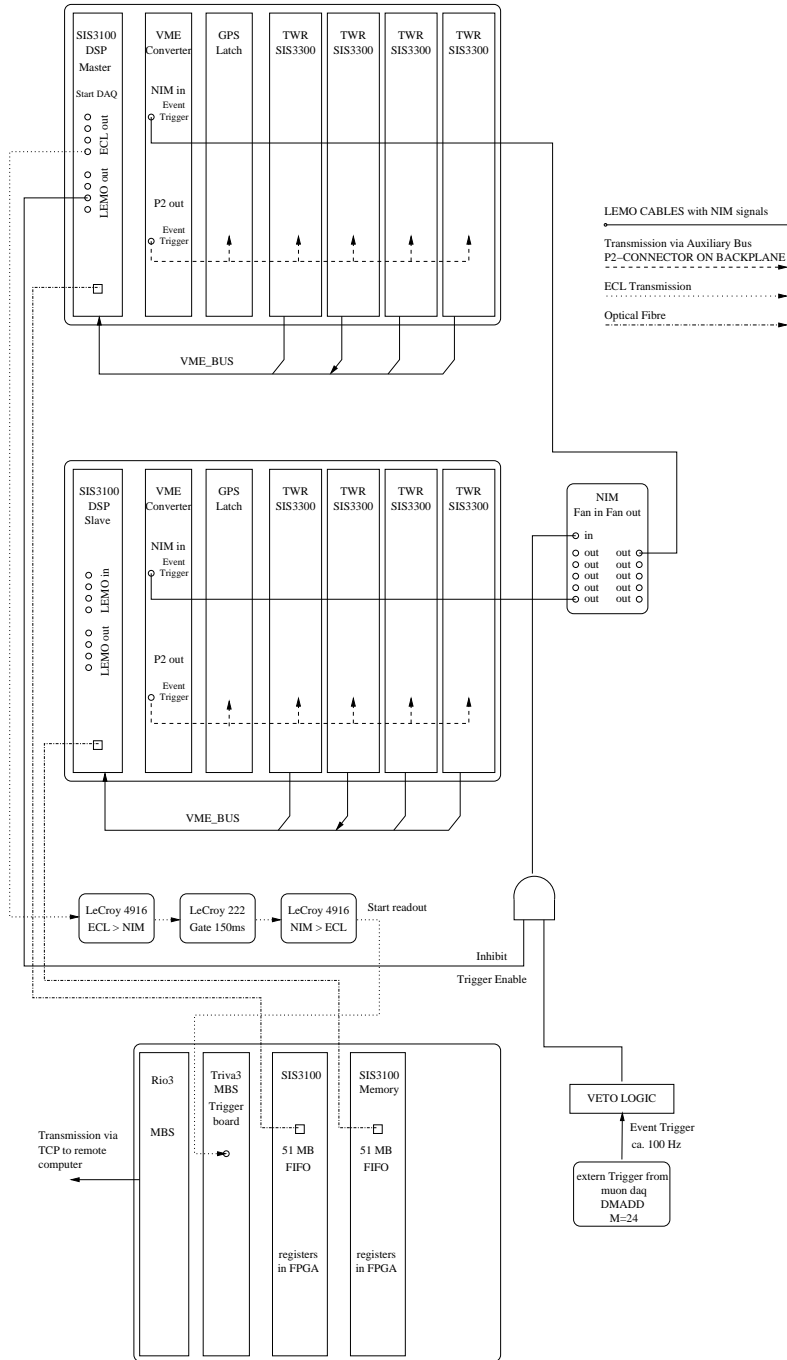


FIGURE 1: *The complete TWR DAQ system.*

1.2 The GPS-Latch GPSAMD

The external trigger signal is delivered simultaneously to all TWRs and to the GPS-latch, which is synchronized by a connected GPS-clock and samples a timestamp for every event.

The GPS-Latch holds a FIFO containing the timestamps each consisting of 4 longwords and is read out together with the TWRs every 128 events.

1.3 The Digital Signal Processor DSP

On the VME bridge module in the slave crates, which acts as a VME master in these crates a Digital Signal Processor is located. The DSP is optimized for large data amounts and small algorithms. Since it is programmed in assembler, it is very fast. The DSP reads out all the TWRs and the GPS Latch and sends the data to the VME bridge module located in the master crate. The program for the DSP is loaded during the initialisation by the MBS. (It is intended to apply the feature extraction algorithm to all waveforms in the next season.)

1.4 The Initialisation process

The complete system is controlled by the Multi Branch system - MBS, a data acquisition system, which has been developed by the german institute GSI ²⁾. The MBS-system is running on the RIO3 under Lynx. It consists of several independent processes, for instance the datacollector task, transport task and the readout task.

When starting MBS the configuration file TWR.cnf containing all important information is loaded. This file is described in chapter 3 and locates on twr-daql.spole.gov in the directory /export/seal1/lynx/RIO3_3.1/mbsusr/user1/mbsrun/rio3_newdaq .

The initialisation includes a small routine determining the baseline of each channel independently is started. The TWRs are started and will sample 128 waveforms (most of them are empty !) and will determine the baseline for every channel by integrating over all waveforms and calculating the mean value. Since most of the waveforms are empty this is a good approximation for the baseline.

These values are integrated to the data stream and will appear in the header of every raw data file. An analysis of baseline shifts is possible in this way and the daq is comparable independent from long term baseline shifts during the winter.

After this, the initialisation of the TWR will start. The threshold for detecting valid waveforms is calculated by

$$\text{(threshold for TWR)} = \text{(baseline)} - \text{(threshold from config file)}$$

The GPS-Latch is started and the TWRs are armed for sampling. Then the DSP program is loaded and startet.

In order to start the system synchronously, the trigger is blocked by the TWR daq during initialisation and will be enabled at the beginning of the run. This is explained in detail in the description of the trigger logic 1.6

²⁾Gesellschaft fuer Schwerionenforschung in Darmstadt

1.5 The Readout process and dataflow

The readout of the slave crates is done by the DSP located on the VME bridge module in the same crate. As soon as the DSP program is started, the DSP is polling the Bank full flag of the first TWR in the crate. If the first bank is filled the DSP starts reading out all the TWRs and the GPSAMD and sends a signal to the MBS triggerboard TRIVA3 (blue) located in the master crate. This signal will start the readout procedure of the MBS.

The MBS readout procedure will read the data from a FIFO located in the local VME bridge module in the master crate and will apply the feature extraction algorithm to all waveforms and writes the data to a pipe in the memory of the rio3.

Errors which are found during the readout like for synchronisation between the TWR timestamps and the GPS times can be found in an errorfile which is opened for every run.

An eventbuilder program running on the Linux-computer twr-daq1.spole.gov reads the data via TCP and starts building the events. In the rawdata coming from the TWRs the data is sorted by channels. Eventbuilding means to sort the data by time-events beginning with the GPS-timestamp. The output of the eventbuilderprogram is a binaryfile, which format can be seen in figure 5. The header of this file contains the whole structure of the system and all important numbers, like the threshold applied for the feature extraction. The structure of the header can be seen in figure 4. The eventbuilder writes all the data in files to a disk. The filename convention is described in chapter 4.

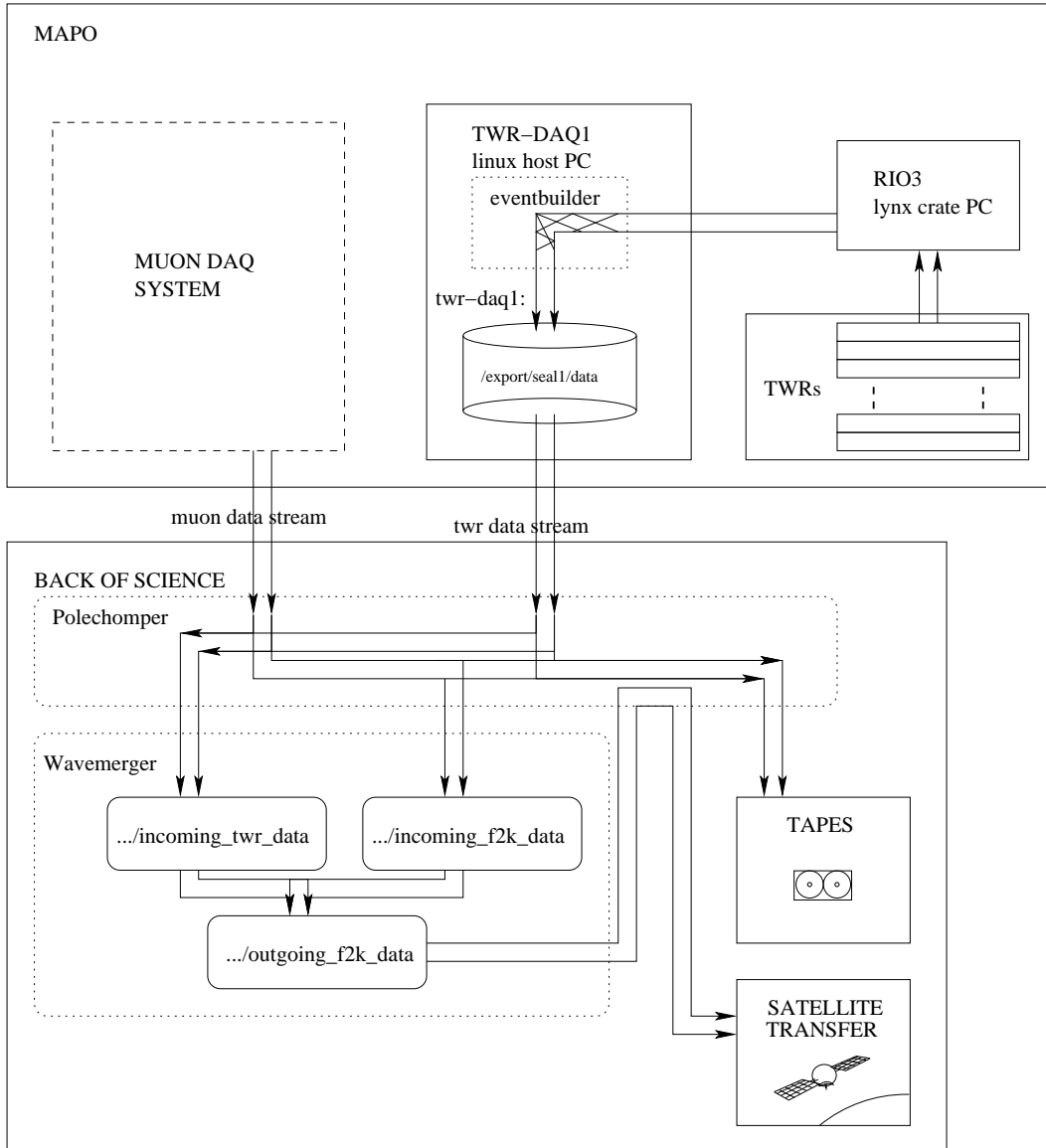
The TWR-data has to be merged with the muon-data. This means to identify events which belong to the same physical event and is performed by the wavemerge program, which is described in XXXXXXXXXXXX. An overview on the whole dataflow for the TWR daq is shown in figure 2.

1.6 The trigger system

Since the TWR DAQ in the current version has no internal trigger system, it has to use a trigger distributed by the current muon DAQ. While in 2002 a Majority = 80 trigger with a frequency of about 2.5 Hz was used, it is intended to use in 2003 the Majority = 24 trigger from the muon DAQ. The original trigger signal from the DMADD goes through a veto-logic to stabilize the system. The whole system can be seen in figure 3.

The external trigger pulses from the DMADD (event-trigger) is used as input for a level three coincidence. Only, if the TRIGGER ENABLE from the TWR daq and the and the VETO DISABLE show also a logic 1, the trigger is passing and starts the TWR. The veto is done by a gate generator which is connected to the output of the coincidence module with a delay of about 30ns. Since the inverse output of the gate generator is connected to the logic coincidence, it will disable the output of the coincidence module for the chosen length of the gate signal.

The passing trigger is distributed by a logic fan out to the different crates holding the TWRs and GPS Latches. In order to reduce the cabling in front of the crates and to make the system more reliable, the trigger is distributed via the P2 bus on the VME backplane. The trigger is received by a converter board in the crates, which changes the signal level to ECL and transmits it to the backplane. The incoming trigger signal is delayed by the TWR for a programmable number of clock cycles (internal clock of the TWR 100MHz)



Copyright: Thomas Feser

FIGURE 2: *Dataflow overview*

and serves as a COMMON STOP TRIGGER to the TWR, which switches to the next ring buffer and starts sampling for the next event. This programmable delay is useful for levelling of the different leading edge times of OMs in different depths. The first daq in 2002 used a fixed delay of about $7\mu\text{sec}$, which leads to a time window of about $3\mu\text{sec}$ before and $7\mu\text{sec}$ after the trigger arrival (in order to get also a large amount of afterpulses). This external trigger signal is delivered simultaneously to all TWRs and to the GPS-latch, which is synchronized by a connected GPS-clock and samples a timestamp for every event. The GPS-Latch holds a FIFO containing the timestamps.

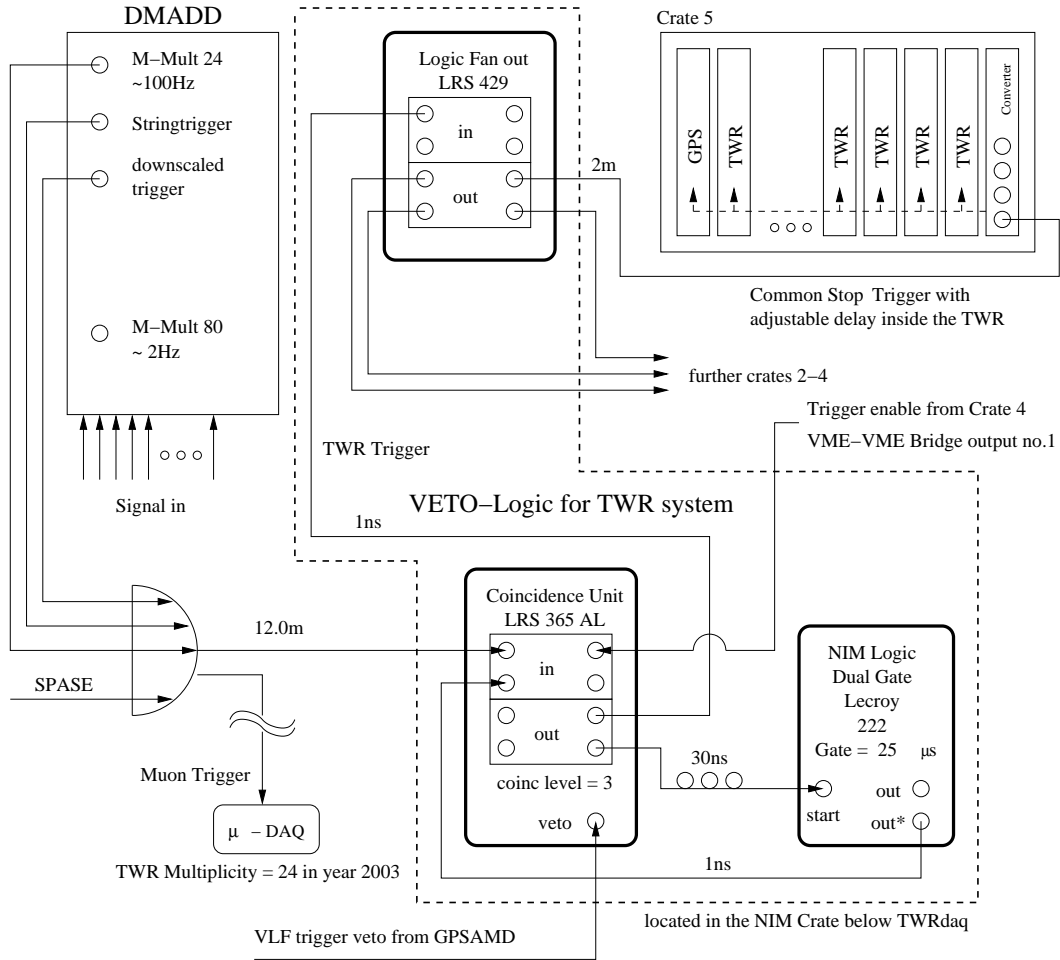


FIGURE 3: *Trigger setup overview*

The complete layout of the dataflow is displayed in figure 2. To reduce the amount of data all empty waveforms, that means all waveforms with signals below a certain threshold are not read out. The TWR is initialized
 A description of the trigger system including all important timedelays can be seen in figure 3.

2 Running the system

There are two possibilities to run the TWR-daq. Normally the TWR-daq is started together with the muon-daq by a PERL script in *Automatic Mode*. For testing purposes this document describes also running the TWR-DAQ step by step manually.

2.1 Automatic mode

The TWR is started by the PERL script `remote_start_twr_daq.pl` on `amanda-daq`, whenever the `muon-daq` is started. It is located in the directory `daqops/current_daq`. The corresponding stop-script, `remote_stop_twr_daq.pl` is located at the same position.

The start-script starts another script, `start_twr_daq.pl` on `twr-daq1`, which is located in the directory `/home/daq/current_daq`. `start_twr_daq.pl` starts all the necessary processes on `twr-daq1` and `rio2`. Stopping the TWR-daq is done by the script `remote_stop_twr_daq.pl` on `amanda-daq` and `stop_twr_daq.pl`. Both `start_twr_daq.pl` and `stop_twr_daq.pl` can be started independent from the `muon-daq` on `twr-daq1`. In case of problems the output of the eventbuilder can be checked by the log-file `log.eventbuilder`, which can be found in `/home/daq/current_daq`. In addition there is a restart-script, which stops the system and starts it again. These scripts use the same commands as described in chapter 2.2.

2.2 Starting the system manually

First the Linuxserver `twr-daq1.spole.gov` has to be started. Login as user and change to the directory `/home/daq/current_daq`.

After this start the VME-setup. When the power on the VME-crate is turned on, the `rio2` Power PC boots automatically via `nfs` after the reset switch on the `rio2` has been pushed. The `rio2` can be reached by

```
telnet rio3
```

or by `minicom`. Normally use `telnet` and login as user `user1`. To start the MBS system change the current directory to `/nfs/mbsusr/user1/mbsrun/rio2_only_vme` and type

```
resl ,  
mbs
```

The MBS-system is started and there should be the following message:

```
mbs> -decw13:msg_log :Message logger running
```

Otherwise MBS has perhaps already been started and is running in another window. In this case you have to leave MBS and type

```
resl ,
```

which means "`reset -local`" and kills all the running MBS-processes. A startup-procedure has to be started to initialize the MBS-system by typing

```
@startup
```

This will not work, if you are not in the right directory. During this procedure the MBS-setupfile (`setup.usf`), which contains a lot of necessary information on the MBS-system is read. Several tasks are started and the messages on the screen should look like

```
-decw13:util :task m_util started  
-decw13:util :setup file setup.usf successfully loaded  
-decw13:util :trigger module set up as MASTER, crate nr: 0  
-decw13:transport:task m_transport started  
mbs> -decw13:transport:starting server in inclusive mode  
-decw13:read_meb :task m_read_meb started  
-decw13:collector:task m_collector started
```

-decw13:transport:waiting for client (port 6000)

The system is now waiting for a client, which can connect from the same or from another computer via TCP. At this time the Eventbuilder has to be started on twr-daq1. Login as user daq and change to the directory current_daq and type

```
./eventbuilder rio3 <run no.> <file no.>
```

where file no. is the number of the first file, which is incremented for every following file of this run. In case of a successful connection, the MBS-system shows the following message:

```
-decw13:transport:Client 192.168.1.1 connected
```

Type

```
start acq
```

at the MBS-prompt to start the acquisition. There should be the following messages:

```
-decw13:util :start acquisition
```

```
mbs> -decw13:read_meb :found trig type 14 == start acquisition
```

```
Start TWR initialization
```

```
Start Sampling
```

```
pl_dat: 0x80005394 *l_se_read_len: 680
```

```
-decw13:collector:acquisition running
```

The MBS-system has started the initialization and read the configuration-file TWR.cnf in the current directory. In this file all the information of the VME-modules to be read out by the MBS are collected, as for example the base addresses. During the initialization the TWR and the GPSAMD-board are initialized. The MBS system generates a first softwaretrigger with number 14. MBS is capable to distinguish between 15 triggers, but only 3 are used in our system: 14, 1 and 15. Every trigger is connected with a special readout-function. Trigger 14 is only used once in every run as a start trigger. The readout-function started by this trigger reads back all the control registers of the modules and stores them with other configuration details at the beginning of every run. Then the acquisition is started by arming the TWR and the GPSAMD for sampling.

To get information about the data acquisition type

```
show acquisition
```

or

```
show rate
```

To stop the data acquisition type

```
stop acq
```

You will receive the following message:

```
End Sampling...
```

```
pl_dat: 0x800053a0 *l_se_read_len: 0
```

```
-decw13:util :stop acquisition
```

```
mbs> -decw13:read_meb :found trig type 15 == stop acquisition
```

```
-decw13:collector:acquisition NOT running
```

At the same time the eventbuilder program on the Linux-server will recognize the software trigger 15, which indicates the end of the current run and stops working. To leave the MBS-system type

quit

2.3 Stopping the system and shutting down

In all circumstances the rio3 has to be shut down before the Linux server. To shut down the rio2, log in as root and type

reboot -h

After this happened, the VME power can be turned off from the master VME crate and the Linux server can be rebooted.

3 Description of the config file TWR.cnf

The TWR.cnf, which contains important information on the channels like the number of the connected OM, the baseline (zerolevel) and the applied threshold, is located on: rio3 in /nfs/mbsusr/user1/mbsrun/rio2_only_vme

All changes have to be mirrored in this file. The clock_predivider sets the speed of the internal timestamp-counter relative to the TWR frequency of 100 MHz. A value of 100 means that the timestamp-counter is running with a frequency of $100 \text{ MHz} / 100 = 1 \text{ MHz}$ and has an accuracy of $1 \mu\text{sec}$.

A typical TWR.cnf can be seen here:

```
GENERAL
CLOCK_PREDIVIDER      100

MASTER_CRATE

N_CRATES 3
BASE_BRIDGE_1         0x0a000000
BASE_BRIDGE_2         0x08000000
BASE_BRIDGE_3         0x06000000

CRATE_1
BASE_100MHz           0x0
BASE_GPS               0x10000000
N_TWR                  0x10

BASE_TWR               0x00000000
STATUS_REG             0x1
ACQ_CONTROL_REG       0x2
EXTERN_STRT_DEL        0
EXTERN_STOP_DEL        700
EVT_CONFIG_REG        0x3
TWR_OM                 640  641  643  644  645  646  647  648
TWR_BASELINE           4000 4000 4000 4000 4000 4000 4000 4000
TWR_THRESHOLD          20   20   20   20   20   20   20   20
...
BASE_TWR               0xf0000000
STATUS_REG             0x1
ACQ_CONTROL_REG       0x2
EXTERN_STRT_DEL        0x0
EXTERN_STOP_DEL        700
EVT_CONFIG_REG        0x3
TWR_OM                 123  124  125  126  127  128  129  130
TWR_BASELINE           4000 4000 4000 4000 4000 4000 4000 4000
```

```

TWR_THRESHOLD          20  20  20  20  20  20  20  20
CRATE_2
BASE_100MHz            0x0
BASE_GPS               0x10000000
N_TWR                 0x10
...

```

All changes in the setup like changing OMs connected to the TWRs have to be documented in the TWR.cnf !//

4 Filename convention

The convention for the binary filenames is

```
twr_YYYY_ddd_rrr_fff.dat
```

with

```

YYYY    year
ddd     day in year
rrr     run number as received from muon daq
fff     file number incremented with every new file of actual run

```

/sectionLayout of the binary file format /labelbinaryformat

Header for the binary datafiles of the new TWRdaq

VER 1.0 6.10.02

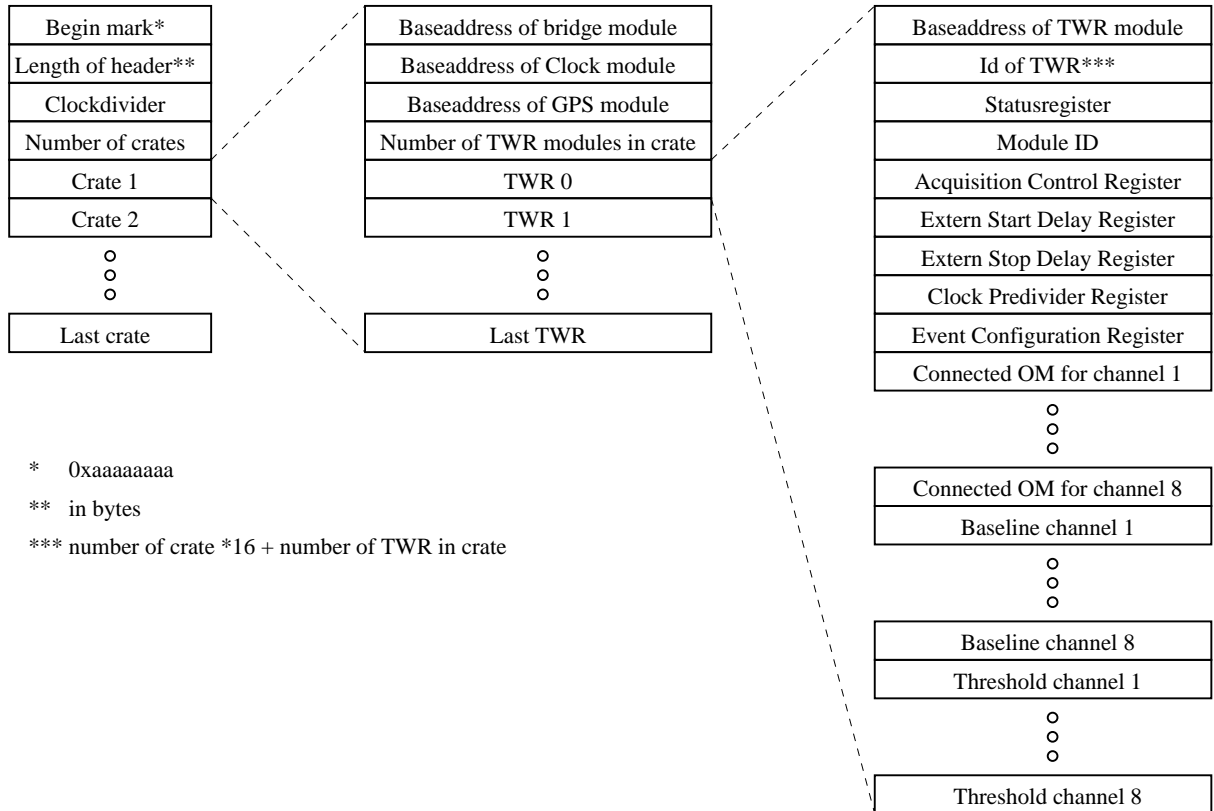


FIGURE 4: Header of binaryfile

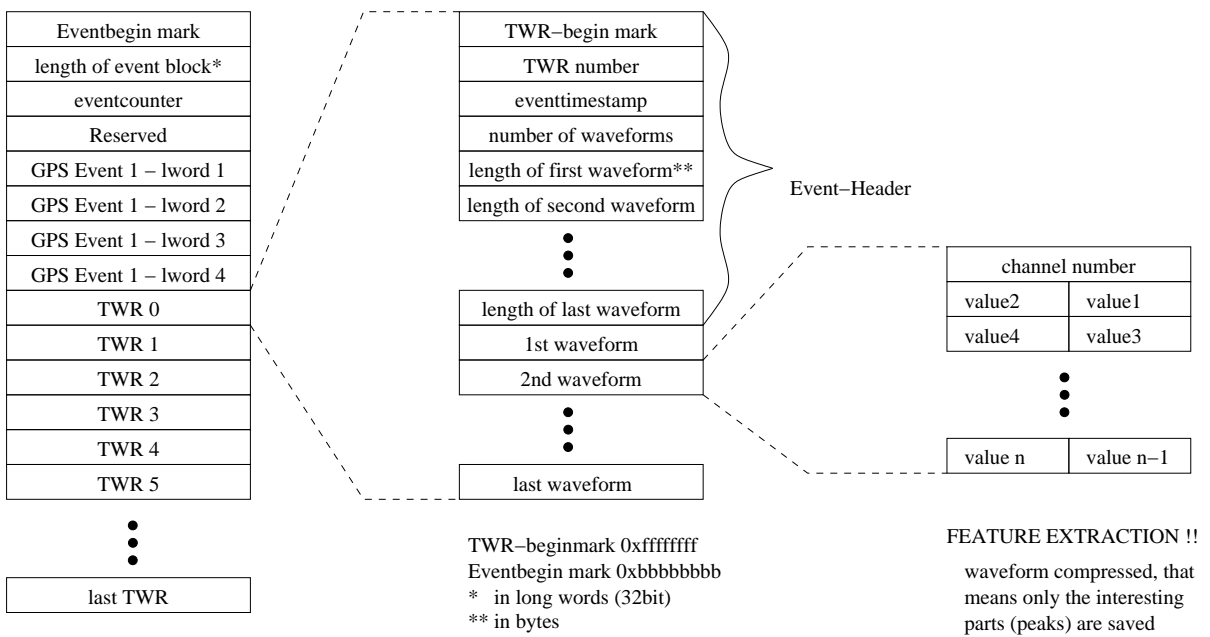


FIGURE 5: Dataformat of binaryfile