

DAQ data structure for the Muon g-2 experiment

Wes Gohn, Tim Gorringe, Ran Hong, Kim Siang Khaw, David Sweigart

January 5, 2017

Abstract

This document outlines the DAQ data structure of the Muon g-2 experiment. A detailed list of the MIDAS data bank will be shown and their contents will be described.

Contents

1	MIDAS DAQ output in a nutshell	2
2	MIDAS Bank list	3
2.1	Calorimeter-related banks	3
2.2	Auxiliary detector-related banks	3
2.3	CCC related banks	4
2.4	Field related banks	4
3	Bank contents	5
3.1	Calorimeter-related banks	5
3.2	Auxiliary detector-related banks	9
3.3	CCC related banks	9
3.4	Field related banks	9
4	Parsers for MIDAS bank data	12

1 MIDAS DAQ output in a nutshell

The main DAQ framework for the Muon g-2 experiment is based on MIDAS [cite]. MIDAS event structure is as depicted in Fig. 1.

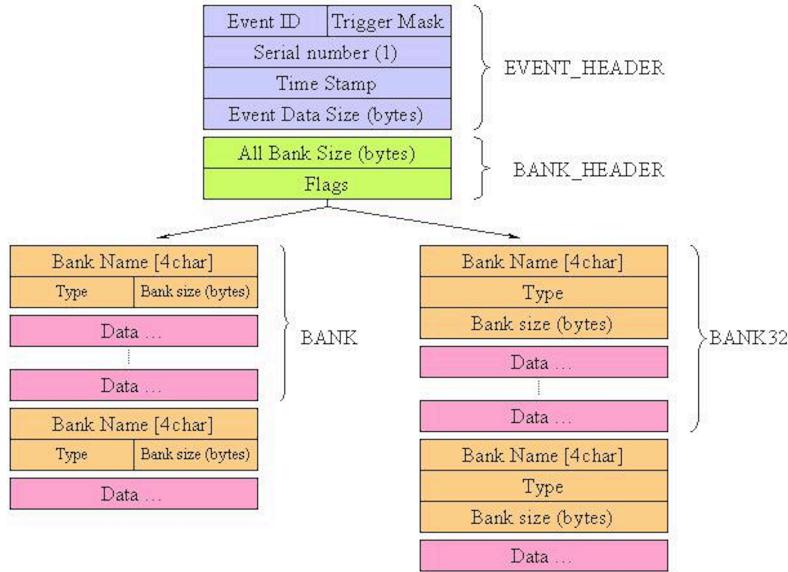


Figure 1: *MIDAS event structure. Each event has its header that is followed by the bank header. Then all the banks will appear according the defined order.*

2 MIDAS Bank list

Hundred of banks will be stored in each MIDAS event and it is very important to classify them properly. At the moment they can be grouped into 4 categories: calorimeter, auxiliary detector, CCC and magnetic field. Naming of these banks will be described in this section and their contents will be explained in the next section.

2.1 Calorimeter-related banks

There are 3 fill types for the calorimeter. Muon fill is the typical muon events, laser fill is event dedicated for laser calibration and monitoring events and pedestal fill is trivia from its name. Data from each fill type is identified from the bank name. The muon fill is denoted by “C”, the laser fill is denoted by “L” and the pedestal fill is denoted by “P”. A summary of the banks is listed in Tab. 1.

Table 1: *MIDAS bank list for the calorimetry data.*

muon fill	laser fill	pedestal fill	Description
Bank name			
CA	LA	PA	AMC13 Header
CB	LB	PB	WFD5 header
CC	LC	PC	GPU timing data
CF	LF	PF	GPU fitted data
CH	LH	PH	per-crystal Q-method data (N-th event, end of run)
CL	LL	PL	Clock data
CP	LP	PP	Pedestal
CQ	LQ	PQ	per-calorimeter Q-method data (every event)
CR	LR	PR	WFD5 raw data
CT	LT	PT	T-method islands
CZ	LZ	PZ	AMC13 CDF trailers

Since each bank has to be named exactly four characters, the last 2 characters denoted the calorimeter number, e.g. **CT03** means the T-method bank from calorimeter number 3.

2.2 Auxiliary detector-related banks

A separate T/Q-method is needed for auxiliary detectors. Their data banks are denoted with the initial “K”. A list of these banks are summarized in Tab. 2.

Table 2: *MIDAS bank list for auxiliary T/Q data. This is mainly for the fiber harps, quads and kickers.*

Bank name	Description
KH	Per aux. detector channel Q-method data (N-th event, end of run)
KQ	Per aux. detector Q-method data (every event)
KT	T-method data

The last 2 characters of the bank name will be decided soon.

2.3 CCC related banks

This is the bank storing the information regarding the CCC system based on FC7. A list of these banks are summarized in Tab. 3.

Table 3: MIDAS bank list for the CCC data.

TTCA	AMC13 Header
TTCR	CCC AMC13 Payload
TTCZ	AMC13 Trailer

2.4 Field related banks

All field-team banks are filled once per event and the definition of an event is different from the one of the ω_a related banks. For many field-team banks, a c struct is defined in the `field_struct.hh` file, accessible for all frontends and unpackers. Programmers should able to cast the read-out bank (array of bytes) onto a pointer of the corresponding struct. A midas bank can be an entire struct (like **TLNP**, **ABPR**, etc) or a array of structs (like **GALI**). A list of these banks are summarized in Tab. 4.

Table 4: MIDAS bank list for the magnetic field related data.

System	Name	Description
Fixed probe	FXPR	Fixed probe, header + NMR waveforms
Trolley	TLNP	Trolley NMR Pulse, header + NMR waveforms
	TLBC	Trolley Barcode, header + Barcode waveforms
	TLMN	Trolley Monitors (temperatures, voltages and pressure), header + voltage waveforms
	GALI	Galil (trolley and garage) data, positions + velocities + control voltages + tensions
Absolute probes	ABPR	Absolute probe (spherical probe and plunging probe are using the same bank), header + NMR waveforms
Flux gate	FLUX	Flux gate, fluxgate waveforms
Surface coil	SFCL	Surface coil, current readouts

3 Bank contents

This section details contents of each MIDAS bank. All the banks are packed in 32-bit word integer regardless of the original format.

3.1 Calorimeter-related banks

CA (LA, PA) banks

This is the bank for the AMC13 to DAQ header information. The first 64-bit word is the CDF header and the next 64-bit word is the payload header as shown in Fig. 2.

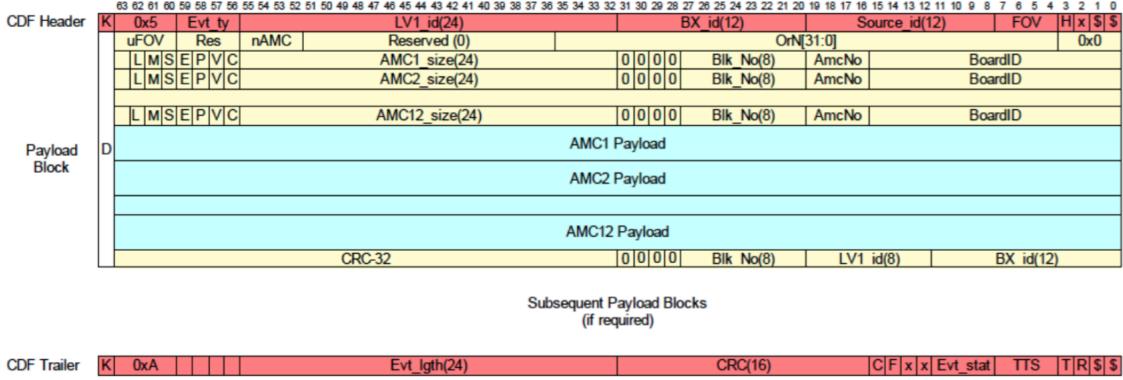


Figure 2: Data structure for AMC13 to DAQ. The first 2 64-bit words are stored in the CA (LA, PA) bank.

CZ (LZ, PZ) banks

This is the bank for the AMC13 to DAQ trailer information. The first 64-bit word is the last 64-bit word of the payload block and the next 64-bit word is the CDF trailer as shown in Fig. 2.

CB (LB, PB) banks

This is the bank for the WFD5 to AMC13 header information as shown in Fig. 3.

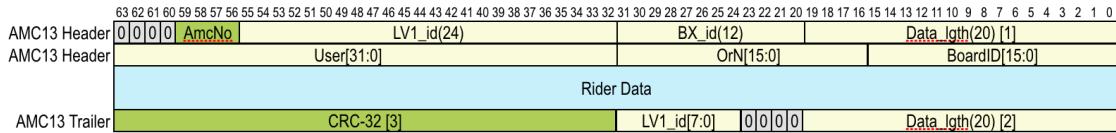


Figure 3: Data structure for Rider to AMC13.

CR (LR, PR) banks

This is the bank for the full WFD5 payload as shown in Fig. 4. Due to the huge data size of raw payload, a pre-scale of about 1000 event is usually being set in the MIDAS ODB.

	63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0			
Channel Header	0 1 Channel Tag [15:0]	Waveform Gap [21:0]	Waveform Count [11:0]	DDR3 Start Address [25:14]
Channel Header	DDR3 Start Address [13:0]	Waveform Length [22:0]	FT	Trigger Number [23:0]
Waveform 1 Header	Waveform Count [11:0]	DDR3 Start Address [25:0]	FT	Waveform Length [22:0]
Waveform 1 Header	0 1 0	Channel Tag [15:0]	Waveform Gap [21:0]	Waveform Index [11:0]
	Waveform 1 ADC Data			
Waveform 2 Header	Waveform Count [11:0]	DDR3 Start Address [25:0]	FT	Waveform Length [22:0]
Waveform 2 Header	0 1 0	Channel Tag [15:0]	Waveform Gap [21:0]	Waveform Index [11:0]
	Waveform 2 ADC Data			
	...			
Waveform N Header	Waveform Count [11:0]	DDR3 Start Address [25:0]	FT	Waveform Length [22:0]
Waveform N Header	0 1 0	Channel Tag [15:0]	Waveform Gap [21:0]	Waveform Index [11:0]
	Waveform N ADC Data			
Channel Trailer	Channel Checksum			
Channel Trailer	Channel Checksum			
Channel Trailer	Data Integrity Check		Data Transfer Time	

Figure 4: Data structure for the WFD5 raw payload.

CT (LT, PT) banks

This is the bank for calorimeter T-method chopped islands. A collection of 54 islands from 54 crystals/segments are extracted per trigger. The detailed structure is shown in Fig. 5.

Bank :	CT04	Length :	243196(I*1)/60799(I*4)/121598(Type)	Type :	Signed Integer*2
1 ->	-9480	1	23 54 27 0	13554	0
9 ->	98	0	1119 1129 1129 1125	1120	1148
17 ->	1134	1197	1182 1531 2046 2046	2046	2046
25 ->	1930	1507	1352 1285 1237 1223	1210	1197
33 ->	1190	1175	1160 1163 1164 1158	1143	1151
41 ->	1159	1159	1149 1140 1137 1150	1143	1143
49 ->	1143	1140	1142 1149 1128 1135	1138	1134
57 ->	1138	1135	1125 1135 1133 1131	1129	1128
65 ->	1137	1140	1136 1134 1132 1141	1130	1128

array of signed tow-byte integers

32-bit number of 16-bit words in bank (the above entry maps to 0x0001date = 121598)

16-bit number of islands

16-bit number of segments

32-bit CTAG /TBD

number of islands x (

32-bit island time +

32-bit island length +

+ number of segments * length of island * 16-bit ADC samples)

Figure 5: Data structure for the CT bank (T-method chopped islands).

CH (LH, PH) banks

This is the bank for calorimeter segment histograms. Histogram from each segment is being summed to itself each fill and is flushed out every N-th event. The detailed structure is shown in Fig. 6.

CQ (LQ, PQ) banks

This is the bank for calorimeter sum histograms. Histograms from all segments are summed together and decimated, and then are being flushed out every fill. The detailed structure is

```

Bank:CH01 Length: 3780016(I*1)/945004(I*4)/945004(Type) Type:Unsigned Integer*4
1-> 0x000e6b68 0x00000001 0x000222e0 0x00000030 0x00007084 0x0000707c 0x00007070 0x00007074
9-> 0x00007080 0x0000706c 0x00007074 0x0000707a 0x0000706c 0x0000707a 0x00007070 0x00007074
17-> 0x0000707e 0x00007082 0x0000708a 0x0000707e 0x00007076 0x00007072 0x00007084 0x00007072
25-> 0x00007076 0x00007082 0x00007076 0x00007070 0x0000707a 0x0000707e 0x00007078 0x00007080
33-> 0x0000707a 0x00007076 0x00007076 0x0000707e 0x0000707e 0x0000707e 0x00007080 0x00007074
41-> 0x00007076 0x00007078 0x00007076 0x0000707c 0x00007078 0x0000706e 0x00007072 0x0000707e
49-> 0x00007078 0x00007076 0x0000707c 0x00007070 0x00007076 0x00007074 0x00007078 0x00007078

```

CH databank words are signed 32-bit signed integers

first word - number of array elements of Q method histogram
second word - first ADC sample within fill of Q-method histogram (is an ODB parameter)
third word - last ADC sample within fill of Q-method histogram (is an ODB parameter)
fourth word - number of segments / detectors in histogram (derived from ODB parameters)
remaining words - Q-method histogram array elements of size specified by first word

Figure 6: Data structure for the CH bank (calo segment histograms).

shown in Fig. 7.

```

Bank:CQ04 Length: 70004(I*1)/17501(I*4)/17501(Type) Type:Unsigned Integer*4
1-> 0x0000445c 0xffffffff 0xffffffff 0x00000025 0xffffffff 0xffffffff 0x0000002a 0x0000000b
9-> 0x0000004d 0xffffffff 0x0000008a 0x0000007b 0x000000b7 0x000000a0 0x00000048 0x000000ee
17-> 0xffffffff 0x0000002c 0x00000022 0x00000024 0x000000a1 0x0000005a 0x00000041 0x0000007e
25-> 0x00000042 0x00000028 0x000000f6 0x0000003f 0x000000fe 0x0000007f 0x000000c0 0x00000056
33-> 0x0000009a 0x00000082 0x00000067 0x0000012c 0x000000cc 0x00000064 0x00000077 0x00000044
41-> 0xffffffff 0xffffffff 0x00000011 0x000000a7 0x0000004a 0x0000001c 0x00000065 0x00000021

```

(number of histogram array elements + 1) x signed four-byte integers

total number of data words, i.e. histogram array elements + 1
segment summed, time-decimated, pedestal subtracted histogram array elements

Figure 7: Data structure for the CQ bank (calo sum histograms).

CP (LP, PP) banks

This is the bank for calorimeter pedestals. The detailed structure is shown in Fig. 8.

CC (LC, PC) banks

This is the bank for the calorimeter DAQ performance related information. It has information like tcp timing and gpu timing. The detailed structure is shown in Fig. 9.

CR (LR, PR) banks, asynchronous

This is the bank for the WFD5 payload in the asynchronous mode. The detailed structure is shown in Fig. 10.

```

Bank:CP04 Length: 220(I*1)/55(I*4)/55(Type) Type:Real*4 (FMT machine dependent)
1-> 5.400e+01 1.126e+03 1.293e+03 1.301e+03 1.328e+03 1.329e+03 1.780e+03 1.761e+03
9-> 1.761e+03 1.768e+03 1.781e+03 1.774e+03 1.761e+03 1.751e+03 1.780e+03 1.781e+03
17-> 1.764e+03 1.736e+03 1.725e+03 1.711e+03 1.767e+03 1.779e+03 1.751e+03 1.759e+03
25-> 1.768e+03 1.760e+03 1.767e+03 1.752e+03 1.764e+03 1.772e+03 1.765e+03 1.753e+03
33-> 1.754e+03 1.752e+03 1.783e+03 1.780e+03 1.760e+03 1.747e+03 1.736e+03 1.779e+03
41-> 1.767e+03 1.753e+03 1.758e+03 1.730e+03 1.755e+03 1.771e+03 1.799e+03 1.765e+03
49-> 1.779e+03 1.752e+03 1.794e+03 1.753e+03 1.753e+03 1.759e+03 1.742e+03

```

(number of segments + 1) x four bytes float format

number of segments

number of segments x pedestal values

Figure 8: Data structure for the CP bank (*T-method pedestals*).

```

Bank:CC04 Length: 152(I*1)/38(I*4)/38(Type) Type:Unsigned Integer*4
1-> 0x2cf01551 0x0800c0f3 0x584127e1 0x00000000 0x000913c3 0x00000000 0x584127e1 0x00000000
9:- 0x000913c4 0x00000000 0x584127e1 0x00000000 0x0009e7b6 0x00000000 0x584127e1 0x00000000
17:- 0x000a1d59 0x00000000 0x584127e1 0x00000000 0x000a0be5 0x00000000 0x584127e1 0x00000000
25-> 0x000a1d58 0x00000000 0x584127e1 0x00000000 0x000a1d76 0x00000000 0x584127e1 0x00000000
33-> 0x000a1dce 0x00000000 0x00000e75 0x00000000 0x00000e75 0x00000000

```

array of 64-bit words (sec, usecs are obtained from gettimeofday() and struct timeval in sys/time.h)

64-bit CDF header word

TCP proc unlocked / started, first 64-bit word is seconds, second 64-bit word is usecs

got TCP header word, first 64-bit word is seconds, second 64-bit word is usecs

got TCP header word, first 64-bit word is seconds, second 64-bit word is usecs

GPU proc unlocked / started , first 64-bit word is seconds, second 64-bit word is usecs

GPU copy done , first 64-bit word is seconds, second 64-bit word is usecs

GPU proc done , first 64-bit word is seconds, second 64-bit word is usecs

MFE proc unlocked, first 64-bit word is seconds, second 64-bit word is usecs

MFE banks made, first 64-bit word is seconds, second 64-bit word is usecs

current TCP fill number

current GPU fill number

Figure 9: Data structure for the CC bank (*calo performance*).

Figure 10: Data structure for asynchronous mode for Rider.

3.2 Auxiliary detector-related banks

KH and KQ banks

These two banks have the same format as the CH and CQ banks.

KT bank

This bank has the same format as the CT bank.

3.3 CCC related banks

TTCA, TTCR, TTCZ banks

TTCA bank will have the same format as the CA bank in the calorimeter-related data since the uTCA crate has the same data format. TTCZ on the other hand will have the same format as the CZ bank.

There are two types of TTCR banks for the FC7 - encoder and fanout. The data format of the encoder and fanout FC7 are shown in Fig. 11 and Fig. 12, respectively.

AMC13 Header	63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
AMC13 Header	0 0 0 AmcNo Trig_Num[23:0] Timestamp[43:32] Data_Length[19:0]
AMC13 Header	User[12:0] TT Timestamp[31:0] BT Board_ID[12:0]
FC7 Data	Laser_Delay[31:0] Trig_Delay[31:0]
FC7 Data	FC7_Status[55:0] L12_Ports[7:0]
AMC13 Trailer	CRC[31:0] Trig_Num[7:0] 0 0 0 Data_Length[19:0]

Figure 11: Data structure for encoder FC7.

3.4 Field related banks

FXPR bank

This is the bank for fixed probes. It consists of a header and NMR waveforms.

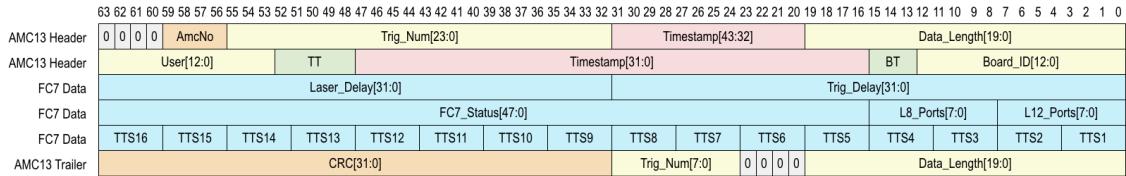


Figure 12: Data structure for fanout FC7.

Table 5: MIDAS bank structure for the FXPR bank.

start word index	type	array length	field name	content	struct name
0	Double_t	num_ch	sys_clock	system clock	fixed_t
4*num_ch	Double_t	num_ch	gps_clock	gps clock	
8*num_ch	Double_t	num_ch	dev_clock	device clock	
12*num_ch	Double_t	num_ch	snr	signal to noise ratio	
16*num_ch	Double_t	num_ch	len	length of each waveform	
20*num_ch	Double_t	num_ch	freq	frequency extracted	
24*num_ch	Double_t	num_ch	ferr	frequency error	
28*num_ch	Double_t	num_ch	freq_zc	frequency extracted, zero crossing	
32*num_ch	Double_t	num_ch	ferr_zc	frequency error, zero crossing	
36*num_ch	UShort_t	num_ch	health	health indicator of probes	
37*num_ch	UShort_t	num_ch	method	frequency extraction method	
38*num_ch	UShort_t	num_ch * rec_len	trace	NMR waveforms: Waveform_Ch1 + Waveform_Ch2 + ... + Waveform_Ch6	

Table 6: Hard-coded macros in the FXPR bank.

Name in the code	Name in this doc	Value
NMR_NUM_FIXED_PROBES	num_ch	378
NMR_FID_LENGTH_RECORD	rec_len	10000

TLNP bank

This is the bank for Trolley NMR pulses. It consists of a header and NMR waveforms.

Table 7: *MIDAS bank structure for the TLNP bank.*

start word index	type	array length	field name	content	struct name
0	ULong64_t	1	gps_clock	Time stamp of the first NMR sample	trolley_nmr_t
4	UShort_t	1	probe_index	probe index	
5	UShort_t	1	length	length of the NMR waveform	
6	Short_t	nmr_len	trace	Trolley Probe NMR waveform	

Table 8: *Hard-coded macros in the TLNP bank.*

Name in the code	Name in this doc	Value
TRLY_NMR_LENGTH	nmr_len	24000

TLBC bank

This is the bank for Trolley barcode readers. It consists of a header and barcode waveforms.

Table 9: *MIDAS bank structure for the TLBC bank.*

start word	index	type	array length	field name content	struct name
0	ULong64_t	1	gps_clock	Time stamp of the first barcode sample	trolley_barcode_t
4	UShort_t	1	length_per_ch	length of the barcode waveform per channel	
5	UShort_t	bc_ch*bc_len	traces	Barcode waveforms: Waveform_Ch1 + Waveform_Ch2 + ... + Waveform_Ch6	

4 Parsers for MIDAS bank data

Muon g-2 offline analysis framework relies on parsers in the gm2parser namespace hosted under repository gm2unpackers to decode the data. To checkout the codes,

```
git clone ssh://p-gm2dqm@cdcv.sfnal.gov/cvs/projects/gm2unpackers
```

Alternatively, you can also use

```
mrb g gm2dqm
```

in our g-2 environment. These parsers are written in C++ and are being used in the *art* producer modules. They can also be used in your standalone C++ codes, if you wish to. The parsers are located at

```
gm2unpackers/calo/parsers
```

at the moment for both calorimeter and CCC related information. This document will be updated accordingly once they are reorganized.