AMC13 DAQ Facilities *DRAFT* E. Hazen – 10 July 2014

The AMC13 provides clock and controls distribution and a generic DAQ interface for MicroTCA crates in CMS. Firmware in use through 2013 was rather HCAL-specific. This document describes a proposed firmware update which will hopefully address the needs of most AMC13 users for DAQ.

Each AMC13 can operate as 3 FED (front-end driver) units, each collecting data from a set of AMC cards, and each providing output on one fiber using the S-Link Express protocol.

Event Builder Overview

The AMC13 DAQ path builds events[1] from 12 AMC cards in a MicroTCA crate, and transmits the completed events to a central DAQ system over a fiber-optic link. Events may also be stored in on-board SDRAM for testing or diagnostic purposes. In response to a trigger[2] signal, each enabled AMC transmits an event fragment to the AMC13. The AMC13 packages these fragments into a single event record. Prototype AMC13 firmware used through 2013 supports a maximum of 4k bytes from each AMC card. This document describes a revision to the event builder to support very large (up to 2^{24} 64-bit words) event fragments.

Large blocks (> 32kB) from one AMC are broken into smaller blocks by the AMC13. Illustration 1 gives an overview of the process. Each 32kB from each AMC are read across the backplane in parallel, then concatenated to form blocks, each with it's own header and sent to the output stream. The entire event is wrapped with header and trailer.

NOTE: The data format in this document is ***PRELIMINARY***. Readers should evaluate this system as a "postal delivery" system which accepts event fragments at one end and delivers them in an easy-to-unpack format through CDAQ with extensive error checking, buffer control and monitoring facilities provided. The details of how this is accomplished may change.

1 In this document the terms "event" and "event builder" refer to data collected for one group of AMC cards per L1A. Of course, a complete event can only be built in the CDAQ.

2 Generic name for signal which initiates the data transfer (L1A in CMS)

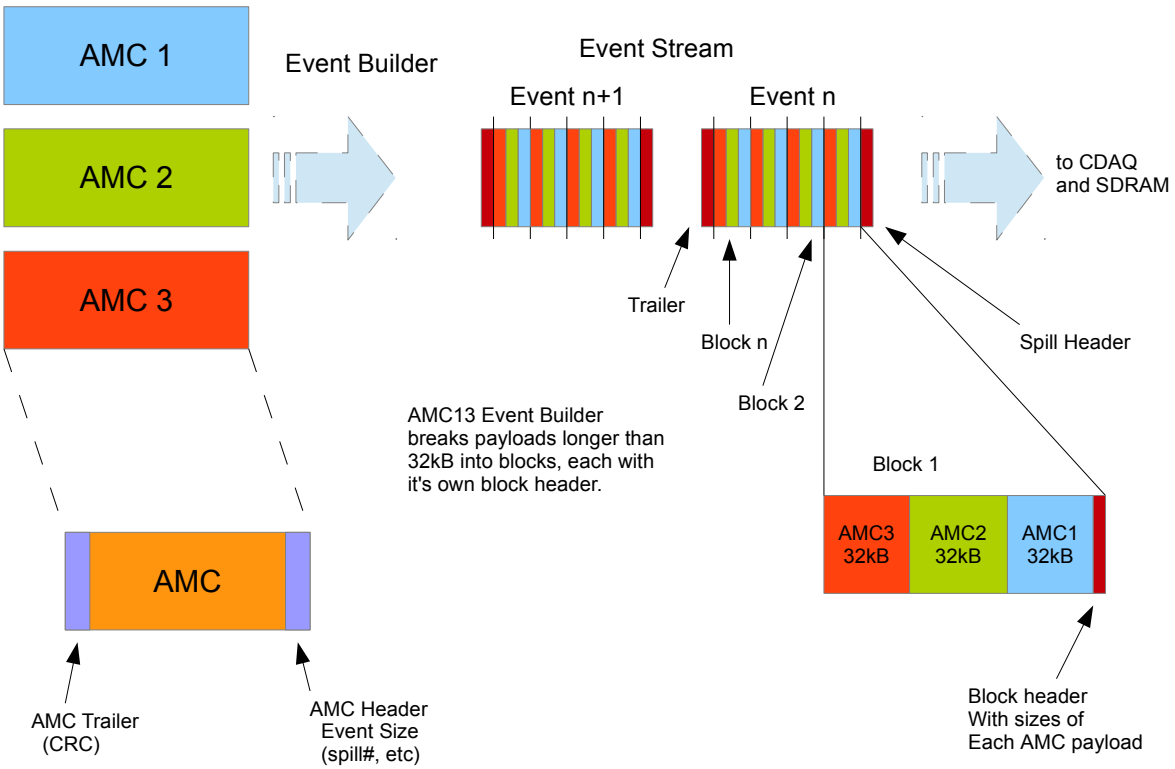

Illustration 1: AMC13 Event Building for Large Payloads

AMC to AMC13 Data Format

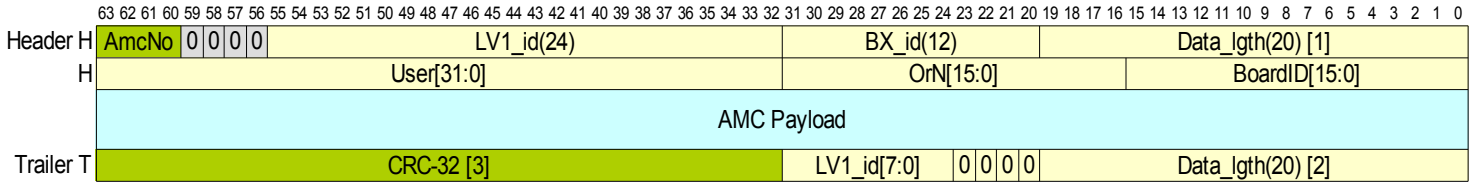

Illustration 2: AMC to AMC13 Format Details

The overall format of a fragment sent from an AMC to the AMC13 is shown in the table below, and in detail in Illustration 2. The size may be up to 2^{20} 64-bit words. A minimum of three 64-bit words (2 header+trailer) must be sent in response to each L1A.

Record Type	**Field Name**	**Source**	**Description**
AMC Header (first word)	AmcNo LV1_id[23:0] BX_id[11:0] Data_lgth[19:0]	AMC13 User User User	Slot number 1-12 filled in by AMC13 Level 1 ID Bunch crossing number Overall fragment size[1] in 64-bit words including header and trailer
AMC Header (second word)	OrN[15:0] BoardID[15:0] User[31:0]	User User User	Orbit number low 15 bits Board ID (not checked but copied to header) Availabe for user
AMC Payload (optional)		User	Arbitrary user content
AMC Trailer (one 64-bit word)	Data_lgth[19:0] LV1_id[7:0] CRC-32	User User AMC13	Overall fragment size[3] in 64-bit words including header and trailer. *Required.* Low 8 bits of event number for consistency check CRC calculated *by Boston AMC firmware* User should store zeroes in this field

Notes:

Bits shown as '0' are reserved and should not be used by AMC cards.

[1] Size must be set to 3 if there is no data for this AMC for this event. The header and trailer are still needed. If the size is not known, it may be set to all 1's *in the header only*.

[2] Size is required in the trailer. Minimum size is 3.

[3] A CRC-32 is calculated by our firmware using the Ethernet polynomial (0x04C11DB7).

In the “Source” column, “User” means user logic which delivers events to Boston-provided AMC link firmware running on an AMC card. “AMC13” means our link firmware or the AMC13 itself. Use data in these fields will be over-written.

AMC13 Output Data Format

The overall format of the AMC13 event builder output is shown in Illustration 3, with details in the following table.

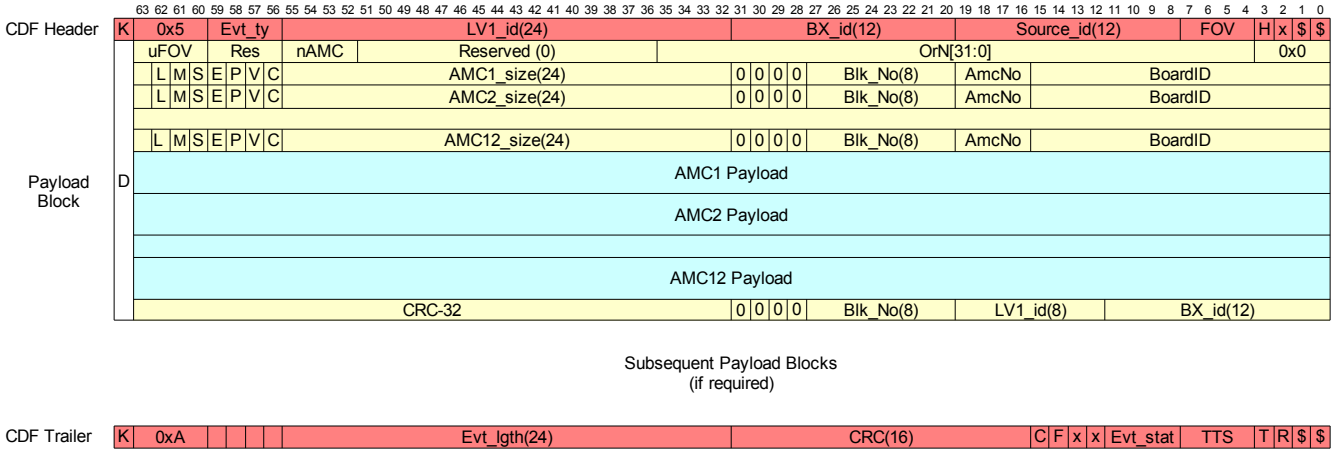

Illustration 3: AMC13 Output Data Format

CMS CDF Header (1 64-bit word)	LV1_id[23:0] BX_id[11:0] Source_ID[11:0]	Level 1 ID (hardware event number) Bunch crossing 0...3563 FED number assigned by CDAQ
AMC13 Header (1 64-bit word)	nAMC[3:0] uFOV[7:0] Res OrN[31:0]	Number of AMCs following (0 to 12) Format version: 0x1 (0 in all previous firmwares) Reserved Orbit Number
AMC13 Payload (up to 2^{24} 64-bit words)		One or more blocks as described below
AMC13 Trailer	Evt_lgth[23:0] CRC	Overall size Overall CRC (CRC-16?)

The AMC13 Payload consists of one or more data blocks. If each AMC sends no more than about 40kB (5k 64-bit words) then there will be only one block (a "non-segmented" event). If any AMC sends more than about 40kB then the event will be broken up into blocks, with each block containing a maximum of 32kB per AMC. Details of the data fields in the blocks are given in the table below.

AMC1 Header (1 word)	Size[19:0] BlkNo[7:0] AmcNo[3:0] * BoardID[15:0] * M S E * P * C V * L	Size: see details below [1] Block sequence number [2] AMC slot number (1-12) Board ID from 2nd header word "More" bit set to 1 for all but last block [3] "Segmented" bit set to 1 for all but first block [3] "Enabled" bit set to 1 if this AMC input enabled "Present" bit set to 1 if data present for this AMC "CRC" bit set to 1 if CRC is valid (last block only) "Valid" bit set to 1 if EvN, BcN match "Length" set to 1 if length in header is correct

AMC2...AMC12 Headers		One 64-bit word per AMC (12 words always present)
AMC1 Payload		Up to 4k words of payload for AMC1
AMC2...AMC12 Payloads		Up to 4k words for each AMC 2..12
Block Trailer (1 word)	BlkNo[7:0] CRC-32 LV1_id BX_id	For consistency checking Calculated on entire block Level 1 ID for consistency checking BX ID for consistency checking

Notes:
* Items marked identical for every block
  Bits L, C are valid only in the last block (when M=0)

[1] Block size:

When M=0 (last or only block) it is the actual size of the block in words

For the first block of a segmented stream (M=1, S=0):
Size is the total size of the data for this AMC, *except:*
Size = 0 if there is no data for this AMC
Size = all 1's (0xffffff) if size not provided by AMC

[3] Block number starts with 0 for each event and is duplicated in 12 AMC header words

[3] M, S bits specify the position in the data stream for the blocks. These bits have the same value for each AMC, as follows:

M=0 S=0	Indicates that this is a non-segmented event with only one block
M=1 S=0	First block in a segmented event
M=1 S=1	Intermediate blocks in a segmented event
M=0 S=1	Last block in a segmented event

TTC Services

The AMC13 distributes the recovered TTC clock at 40.xxx MHz on the backplane MCH1 CLK1 (typically routed to FCLKA on an AMC). The AMC13 distributes the raw 80MB/s TTC A/B channel data on Fabric B (typically routed to port 3 on an AMC). The intention is that the data and clock are sufficiently well-aligned that a DDR register may be used to capture the data.

We can provide on request a reference design for a simple TTC command decoder.

TTS and User Data Backplane Interface

Each AMC may provide a 4-bit TTS status. A clock is required, and should be provided from the clock domain to which the TTS bits are synchronized. The transmitter firmware monitors the TTS state for changes, and transmits updates to the AMC13. After a state change, subsequent changes are ignored for a brief period (50-100ns). The AMC13 will prioritize TTS state inputs in the “obvious” way as enumerated below, with the final TTS front-panel output for the FED representing the highest priority of the AMC inputs plus the internal AMC13 TTS state.

TTS State Priority

1. Disconnected

2. Error
3. Sync Lost
4. Busy
5. Overflow Warning
6. Ready

The AMC13 will transmit the highest-priority TTS state of all enabled event builders by fiber optic link to the TCDS system. If specific information about which source has contributed to a TTS output is required the status registers in the AMC13 may be consulted.

AMC DAQ Backplane Link Interface

We plan to provide firmware for a backplane link from users AMC to the AMC13. The proposed VHDL component declaration for this interface (in the AMC card) is given below:

```
component DAQ_Link
        Generic (
-- REFCLK frequency divider CLK25_DIVIDER = (REFCLK_Frequency/25)
      CLK25_DIVIDER : integer := 4; -- for REFCLK frequency 100MHz
      USE_TRIGGER_PORT : boolean := true );
    Port (
-- asynchronous reset, assert reset until GTX REFCLK stable
     reset : in  STD_LOGIC;
-- GTX signals
     GTX_REFCLK : in  STD_LOGIC;
     GTX_RXN : in  STD_LOGIC;
     GTX_RXP : in  STD_LOGIC;
     GTX_TXN : out  STD_LOGIC;
     GTX_TXP : out  STD_LOGIC;
-- TRIGGER port
     TTCclk : in  STD_LOGIC;
     BcntRes : in  STD_LOGIC;
     trig : in  STD_LOGIC_VECTOR (7 downto 0);
-- TTS
     TTSclk : in  STD_LOGIC; -- clock source which clocks TTS signals
     TTS : in  STD_LOGIC_VECTOR (3 downto 0);
-- Data
     EventDataClk : in  STD_LOGIC;
     EventData_valid : in  STD_LOGIC; -- used as data write enable
     EventData_header : in  STD_LOGIC; -- first data word
     EventData_trailer : in  STD_LOGIC; -- last data word
     EventData : in  STD_LOGIC_VECTOR (63 downto 0);
     AlmostFull : out  STD_LOGIC; -- buffer almost full
     Ready : out  STD_LOGIC);
end component;
```

The AMC user logic is expected to send a minimum of three words (header+trailer) in response to each trigger. Do not send data when `AlmostFull='1'` or `Ready='0'`.

Diagnostic and Monitoring Facilities

The AMC13 firmware provides extensive facilities for monitoring and diagnostic purposes. Below is an incomplete list, and specific additional features may be added at user request (within reasonable limits!). All monitored conditions increment counters which may be read using IPBus.

- Store built events (optionally prescaled) in SDRAM monitor buffer for IPBus readout
- Check every AMC payload for CRC, EvN, BcN and OrN match *vs* AMC13 values
  - Increment error counters per-condition and per-AMC
  - Assert TTS “sync lost” state (optional)
  - Freeze SDRAM monitor buffer with error event centered in window for analysis
- Monitor time spent in all TTS states with 25ns precision
- Recent history of TTS state transitions (per-AMC and per- event builder)
- Recent history of L1As received
- Error counters for all links
- … plus many others!