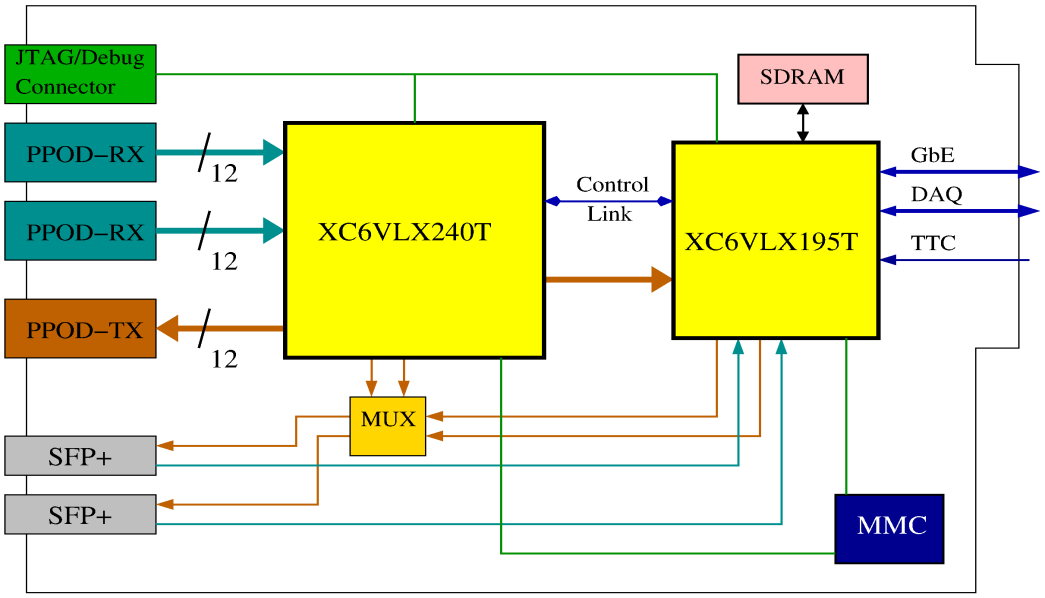
HCAL µTCA Trigger and Readout Module (uHTR)

October 6, 2014

Table of Contents

Overall Architecture



Data Formats on Links

The uHTR has a number of high-speed data links in use.

1. Front-end data links (1.6 Gbps and 4.8/5.0 Gbps)
2. Data links which carry DAQ data from front FPGA to back FPGA (4 Gbps)
3. Data link which carries luminosity and self-trigger information from front FPGA to back FPGA (LHC synchronous 4.8 Gbps or 3.2 Gbps)
4. Trigger links to RCT and GT
5. DAQ data output format to AMC13/DTC

Front-End-Data

Depending on the number of channels transferred, the data format varies. In all cases, the structure of the data is a sequence of twelve bytes for each LHC bunch crossing (effective data rate of 4.8 Gbps). The link can be run either synchronous, in which case the line rate is the same as the effective rate, or asynchronous, in which case the line rate must be somewhat higher than the effective rate. In asynchronous mode, sequences of [K29.7, K28.2, K28.2, K28.2] are inserted to provide latency balancing between the two LHC-synchronous ends of the link.

4-Channel Format (HF)

Byte	7	6	5	4	3	2	1	0
0	K28.5 Comma Character							
1	TDCE3	TDCE2	TDCE1	TDCE0	Reserved (0)			BC0
2	CapId 3		CapId 2		CapId 1		CapId 0	
3	QIE ADC 0							
4	QIE ADC 1							
5	QIE ADC 2							
6	QIE ADC 3							
7	LE TDC 3 [5:4]		LE TDC 2 [5:4]		LE TDC 1 [5:4]		LE TDC 0 [5:4]	
8	LE TDC 3 [3:2]		LE TDC 2 [3:2]		LE TDC 1 [3:2]		LE TDC 0 [3:2]	
9	LE TDC 3 [1:0]		LE TDC 2 [1:0]		LE TDC 1 [1:0]		LE TDC 0 [1:0]	
10	TE TDC 1				TE TDC 0			
11	TE TDC 3				TE TDC 2			

- BC0 : bunch counter alignment marker, sent once per orbit at an agreed phase
- TDCE*n* : Indicates the trailing-edge TDC has an error/multipulse, etc as encoded in the TE TDC field
- QIE ADC : Encoded charge-integrated amplitude
- LE TDC : leading-edge TDC measurement from inside the QIE
- TE TDC : trailing-edge TDC measurement from the Igloo2 FPGA(using the discriminator output from the QIE)

6-Channel Format

Byte	7	6	5	4	3	2	1	0
0	K28.5 Comma Character							
1	LE TDC 0 [3:0]				CapId		CE	BC0
2	QIE ADC 0							
3	QIE ADC 1							
4	QIE ADC 2							
5	QIE ADC 3							
6	QIE ADC 4							
7	QIE ADC 5							
8	LE TDC 1 [5:0]						LE TDC 0 [5:4]	
9	LE TDC 3 [1:0]		LE TDC 2 [5:0]					
10	LE TDC4 [3:0]				LE TDC3 [5:2]			
11	LE TDC 5 [5:0]						LE TDC4 [5:4]	

- BC0 : bunch counter alignment marker, sent once per orbit at an agreed phase
- CE : indicates a mismatch in capid between the four channels
- QIE ADC : Encoded charge-integrated amplitude
- LE TDC : leading-edge TDC measurement from inside the QIE

8-Channel Format

Byte	7	6	5	4	3	2	1	0
0	K28.5 Comma Character							
1	Reserved (0)				CapId		CE	BC0
2	QIE ADC 0							
3	QIE ADC 1							
4	QIE ADC 2							
5	QIE ADC 3							
6	QIE ADC 4							
7	QIE ADC 5							
8	QIE ADC 6							
9	QIE ADC 7							
10	LE TDC 3		LE TDC 2		LE TDC 1		LE TDC 0	
11	LE TDC 7		LE TDC 6		LE TDC 5		LE TDC 4	

- BC0 : bunch counter alignment marker, sent once per orbit at an agreed phase
- CE : indicates a mismatch in capid between the four channels
- QIE ADC : Encoded charge-integrated amplitude
- LE TDC : leading-edge TDC measurement from inside the QIE, remapped into a limited set of bins by the Igloo2 FPGA. The specific remapping may be indicated by the reserved bits

DAQ Front-to-Back Formats

1.6 Gbps Data

For the 1.6 Gbps data, the 32-bits of data on each Front-to-Back link has the following format:

- [7:0] DATA_e (MSB always zero for 1.6 Gbps)
- [9:8] CAPID_e
- [10] CAPID_ERROR_BIT_e, filled in by ZS algorithm in the back FPGA
- [11] OTHER_ERROR_BIT_e
- [12] GBT_ERROR_BIT_e
- [13] IS_ZERO_e, filled in by ZS algorithm in the back FPGA
- [15:14] RESERVED (0)

- [23:16] DATA_o
- [25:24] CAPID_o
- [26] CAPID_ERROR_BIT_o
- [27] OTHER_ERROR_BIT_o
- [28] GBT_ERROR_BIT_o
- [29] IS_ZERO_o
- [30] -> RESERVED (0)
- [31] FIRST_CHANNEL_OF_EVENT

The subscripts “e” and “o” refer to the even and odd samples in a sequence, starting from zero. Thus, “o”=”e”+1.

4.8 Gbps Data

For the 4.8 Gbps data, the 32-bits of data on each Front-to-Back link has the following format:

- [7:0] DATA
- [13:8] RISING_EDGE_TDC
- [17:14] FALLING_EDGE_TDC
- [19:18] CAPID
- [20] CAPID_ERROR
- [21] LINK_ERROR
- [23:22] TDC_ERROR
- [29:24] Reserved (0)
- [30] FIRST_SAMPLE_OF_CHANNEL
- [31] FIRST_CHANNEL_IN_EVENT

Lumi/BHM/Local Trigger Front-to-Back Formats

The Lumi/BHM/Local trigger link must run at a multiple of the LHC frequency. Depending on the available clocks, the link may run at 4.8 Gbps or 3.2 Gbps. The details of the meanings of the various codes are given in the Lumi and BHM sections.

4.8 Gbps Data for HF Luminosity

Index	15							8	7							0
0	0	0	0	0	0	Trig BC0	Lumi OC0	Lumi BC0	K28.5 comma character							
1	Channels-above-threshold (LHC)								Local trigger byte							
2	Channels-above-threshold1 (CMS)								Channels-valid (LHC)							
3	Channels-valid (CMS)								Channels-above-threshold2 (CMS)							
4	SumET [15:0]															
5	Unassigned (0)								Unassigned (0)							

3.2 Gbps Data for HF Luminosity

Index	15							8	7							0
0	Local trigger bits [4:0]					Trig BC0	Lumi OC0	Lumi BC0	K28.5 comma character							
1	Channels-above-threshold (LHC)								Channels-valid							
2	Channels-above-threshold1 (CMS)								Channels-above-threshold2 (CMS)							
3	SumET [15:0]															

4.8 Gbps Data for BHM

Index	15							8	7							0
0	Local trigger bits [4:0]					Trig BC0	Lumi OC0	Lumi BC0	K28.5 comma character							
1	#5[0]	Hit Chan #4			Hit Chan #3			Hit Chan #2			Hit Chan #1			Hit Chan #0		
2	Hit#10[2:1]		Hit Chan #9			Hit Chan #8			Hit Chan #7			Hit Chan #6			Hit#5[2:1]	
3	Hit Chan #15			Hit Chan #14			Hit Chan #13			Hit Chan #12			Hit Chan #11			#10[0]
4	#21[0]	Hit Chan #20			Hit Chan #19			Hit Chan #18			Hit Chan #17			Hit Chan #16		
5	0	0	0	0	0	0	0	0	Hit Chan #23			Hit Chan #22			Hit#21[2:1]	

uHTR DAQ Data Format

Each FED data payload from the AMC13/DTC will contain a number of uHTR blocks. These blocks are best understood as sequences of 16-bit words and consist of a header followed by channel data blocks followed by a trailer. Two versions of the payload have been present in the history of the uHTR firmware.

1. The first version (v0) is identical to the VME HTR data format with the trailer significantly modified.
2. The second version (v1) is based on the 64-bit header for production AMC13 firmware, recast into 16-bit format for convenience of understanding. This version applies for back-FPGA firmwares 0.E.10 and later.

Header v0

Index	**15**							**8**	**7**							**0**
0	Reserved for DCC								EvN[7:0]							
1	EvN[23:8]															
2	1	Space reserved for flags (used in DQM, debugging)														
3	OrN[4:0]					ModuleId[10:0]										
4	FormatVer[3:0] = 0x6				BcN[11:0]											
5	N (TP samples)								Presamples[3:0]				1	1	1	
6	nZS	1	0	0	FirmwareRev[11:0]											
7	0	FirmwareRev [18:12]							Pipeline[7:0]							

Header v1

Index	**15**							**8**	**7**							**0**
0	Data_Length64[15:0]															
1	BcN[11:0]												Data_Length64[19:16]			
2	EvN[15:0]															
3	(Filled in by AMC13)								EvN[23:16]							
4	Presamples[3:0]				SlotId[3:0]				CrateId[7:0]							
5	OrN[15:0]															
6	PayloadFormat=1				EventType				Firmware Flavor[7:0]							
7	FirmwareVersion[19:16],FirmwareVersion[13:8],FirmwareVersion[5:0]															

In the AMC13 specification, word 4 is called “BoardId[15:0]” and is copied to the overall AMC13

header.

“Presamples” is used only for the 1.6 Gbps firmwares, as it is necessary to identify the sample-of-interest for that data format. It is not really a board id function.

Channel Data

The channel data comes in several flavors, depending on whether the channel has a single or dual TDC functionality.

Flavor=0 (or 1)

HB/HE QIE/TDC data, one 6-bit TDC hit-per-channel

It is suggested that flavor=1 could be used for channel data which would have been zero-suppressed but has been passed through under “Mark-and-Pass” behavior.

Index	**15**							**8**	**7**							**0**
n	1	Flavor[2:0]=0			ErrF[1:0]		CapId0[1:0]		ChannelId[7:0]							
n+1	0	SOI	TDCData[5:0] (TS 0)						QIEData[7:0] (TS 0)							
...	0	SOI	TDCData[5:0] (TS 1)						QIEData[7:0] (TS 1)							

- CapId0[1:0]: capacitor id for the first sample. The CapIds are assumed to rotate (0->1->2->3) through the rest of the samples, unless otherwise indicated by the error field
- ErrF[1:0]: Encodes errors observed for this channel (0=none, 1=capid misrotation, 2=link error)
- SOI: high for the “sample-of-interest”, zero for all other samples. Indicates same information as previous “presamples” field

Flavor=2 (or 3)

HF Data with rising and falling edge TDC values (generally fewer TS/readout than HB/HE). In this case, two 16-bit words are required for each time sample.

It is suggested that flavor=3 could be used for channel data which would have been zero-suppressed but has been passed through under “Mark-and-Pass” behavior.

Index	**15**							**8**	**7**							**0**
n	1	Flavor[2:0]=2			ErrF[1:0]		CapId0[1:0]		ChannelId[7:0]							
n+1	0	0	SOI	OK	Resv (0)				QIEData[7:0] (TS 0)							
n+2	0	1	Resv (0)		TDCFall[5:0] (TS 0)					TDCRise[5:0] (TS 0)						
n+3	0	0	SOI	OK	Resv (0)				QIEData[7:0] (TS 1)							
n+4	0	1	Resv (0)		TDCFall[5:0] (TS 1)					TDCRise[5:0] (TS 1)						

Flavor=4

Trigger Data

Index	**15**							**8**	**7**							**0**
n	1	Flavor[2:0]=4			ErrF[1:0]		Reserved(0)		ChannelId[7:0]							
n+1	0	SOI	OK	TPGData[12:0] (TS 0)												
...	0	SOI	OK	TPGData[12:0] (TS 1)												

Flavor=5

Pre-upgrade HTR data (7-bit QIE)

In this case, the HTR header and trailer may be identical to the 2011 HTR data format.
Only even numbers of samples are allowed – expected number is 6-10 samples generally.

Index	**15**							**8**	**7**							**0**
n	1	Flavor[2:0]=5			ErrF[1:0]		CapId0[1:0]		ChannelId[7:0]							
n+1	0	QIE Sample 1							0	QIE Sample 0						
n+2	0	QIE Sample 3							0	QIE Sample 2						
n+3	0	QIE Sample 5							0	QIE Sample 4						

Flavor=7

Technical data. This flavor is used to encode technical, changing data packets which hold information such as link status information, etc. This information for use by DQM only, and would not be unpacked into “detector-geometry” oriented information.

One class of technical data (TechnicalDataType=0) is simply padding words as needed to achieve 64-bit alignment for the overall uHTR payload.

Index	**15**							**8**	**7**							**0**
n	1	Flavor[2:0]=7			TechnicalDataType[3:0]				TechnicalChannelId[7:0]							
n+1	0	Technical Payload (depends on DataType)														
...	0	Technical Payload (depends on DataType)														

Trailer

The trailer is made shorter than the current HTR trailer to reduce the overall data volume slightly. The header+trailer combination is 128 bits (two 64-bit words), so alignment need only be considered within the data portion itself.

Index	**15**							**8**	**7**							**0**
0	CRC[15:0]															
1	EvN[7:0]								Zeroes (possibly overwritten by DTC)							

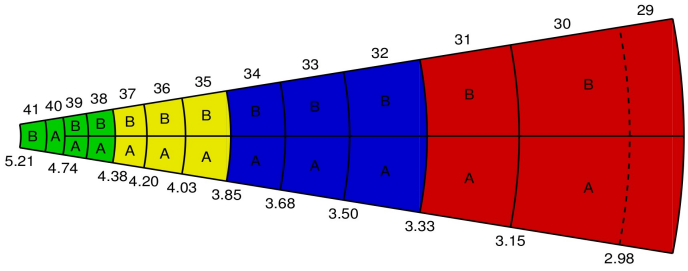
Trigger Primitive Formation

The general data flow for all trigger primitives is similar. For each channel, the raw ADC value from the QIE is converted into a linear E_T scale. This is performed by a lookup-table (LUT) which combines pedestal subtraction, gain and response correction, and the cosh(η) for the conversion between E and E_T. This LUT is generally called the *linearization* LUT. The number of bins in the LUT depends on whether the QIE8 (7-bit ADC) or QIE10/11 (8-bit ADC) is in use.

After linearization, a range of algorithms can be applied depending on the portion of the detector. Generally, some set of channels is summed, possibly after applying a pulse-shape filter. These sums define the various trigger channels which are produced by the uHTR. There are always fewer trigger channels than precision channels in the uHTR. Other algorithms can be applied at this point to define "fine-grain" or "feature" bits. The algorithms in use are discussed in detail in the respective sections below.

After the algorithms are applied, the range of the trigger tower ET is compressed by an LUT agreed between the HCAL and L1 Trigger groups for output. This LUT also handles saturation.

Trigger Primitive Formation in the HF (1.6 Gbps Firmware)



In the HF 1.6 Gbps firmware, the uHTR produces two sets of trigger primitives. The drawing to the right helps explain the two sets of trigger primitives produced for each wedge of HF (each wedge is handled by a single uHTR). In all cases, the short and long fibers are summed for each HF tower. No time-dependent filtering is applied – the trigger primitive for a given bunch crossing is based on the energy from that bunch-crossing alone and independent of any signals before or after.

The first set of trigger primitives is the large colored regions, which sum a 3x2 set of fundamental HF towers. These trigger towers are the "RCT-style" trigger towers. Each uHTR produces four RCT trigger towers, which are sent on a single 4.8 Gbps data fiber to RCT and received by an oRM on the Jet Summary Card.

For the upgraded calorimeter trigger, the uHTR produces a trigger tower for each fundamental tower, except that the tail-catcher tower (HF tower 29) is summed with the HF tower 30 at the same φ location. For the upgraded calorimeter trigger, the long and short fiber E_T values are also passed into a LUT which identifies possible electromagnetic showers based on the ratio between long and short fibers. The trigger primitives are transmitted from each uHTR on two 6.4 Gbps fibers. The channels on each fiber are indicated by "A" and "B" in the figure above.

There are four sets of LUTs in the HF 1.6 Gbps firmware.

- **Group 0.** The linearization LUTs which convert QIE8 ADC values into ET values. There are 128 entries in the LUT, each of which is a 10-bit integer. This allows a dynamic range of 1024.

The LSB must be agreed with the trigger group, but 0.5 GeV E_T is a typical choice, allowing a single-tower saturation at 512 GeV E_T. There are 48 LUTs, as only fibers 2-9 of each PPOD are in use for 1.6 Gbps operation.

- The indexing for group 0 is broken into two pieces. Index values 0-23 refer to PPOD0 and values 24-47 to PPOD1. For each PPOD, the index is `(fiber_in_PPOD-2)*3+channel_on_fiber`.

- **Group 1:** The compression LUTs for the upgrade-trigger fibers. After the sum of pairs of channels, the range is increased by one. Therefore, there are 2048 entries in each compression LUT, with each entry being an eight-bit value for transmission to the trigger.
  - The index for the compression LUTs is `(trig_channel*2+trig_fiber)`, where `trig_channel` runs from 0..10 for each fiber and `trig_fiber` has values 0=A and 1=B.
- **Group 2:** The electromagnetic fine-grain LUTs for the trigger upgrade fibers. There is one for each trigger fiber which compares the long and short fiber ADC values. Each entry is a single bit, encoding the fine-grain bit value to be set for the particular combination of long and short fiber ADC values. There are 16384 total entries in the LUT (128^2).
  - The index for the fine-grain LUTs is the same as for the Group 1 compression LUTs.
  - Within each LUT, the addressing is (`long*128+short`).
- **Group 3:** The compression LUTs for RCT are handled by group 3. There are four LUTs.
  - The index in the LUTs is increasing with `|eta|`

Functionality for the Measurement of Luminosity

The uHTR luminosity system is based on the formation of bunch-by-bunch histograms which can be read out by IPbus or transferred into the DAQ via the AMC13. The following histograms must be calculated:

- Channels-above-threshold for LHC
- Channels-valid for LHC
- Channels-above-threshold-1 for physics
- Channels-above-threshold-2 for physics
- Channels-valid for physics
- Sum(ET) for physics

The histograms which are labeled “for LHC” should be calculated at all times, without respect to run boundaries. The histograms which are labeled “for physics” must respect the CMS run boundaries and line up on CMS luminosity segment boundaries in time.

Within the system, counts of channels and total ET sum across HF will be performed using the front FPGA. These counts will be transferred to the back FPGA using the Front-Back link. The back FPGA will be responsible for accumulating the histograms and making them available for use. Each histogram contains 3564 BX bins, each with a 32-bit dynamic range. Three sets of histograms are hosted in the back FPGA and each is used for an integration period in turn. This allows the readout software to read all the histograms before they are cleared automatically for the next integration period. The firmware also keeps track of the situation when the readout did not occur quickly-enough to avoid overwriting a histogram.

Functionality for the Beam Halo Monitor

For the beam halo monitor system, it is necessary to count the number of hits above threshold in each phototube separately. These hits are binned by BX but also into four sub-bins in each BX based on the TDC value observed from the QIE10. As a result of these specifications, the BHM firmware has very high resource requirements. These requirements have been met by configuring the BHM firmware to support only the use of six fibers (24 channels) in each uHTR. The other fiber inputs are unused. The histograms for the BHM are 3564x4 = 14256 bins long. As the expected bin occupancy at full luminosity is only O(20), each bin is limited to a maximum value of 511 (9 bits) to limit the resource usage. Histograms are triplicated as for the HF luminosity case. The four sub-bins are *not* a fixed division of the TDC code by a factor of 8. As discussed below, the sub-bins are flexible.

The configuration for the BHM requires the following information:

- Channel-by-channel thresholds in QIE10 ADC units to define the minimum amplitude for a beam halo hit. These thresholds can be tuned using the per-channel amplitude histogramming support of the uHTR.

- Mapping between six-bit TDC code and BX-sub-bin id. This is a global setting (all channels share). The mapping can be arbitrary and could, for example, identify “BHM-halo-time” hits +/- 1 ns, “collision-hits” +/- 3 ns, “early-hits” and “late-hits”. The full range of TDC codes can be mapped to sub-bins or mapped to zero, which causes the code to be ignored.

Mezzanine Cards

Formally, a mezzanine cards should be parallel to the base board it is attached to. This is not the case for the sub-assemblies on the uHTR, which are perpendicular to the base board. However, the term “power module” is highly confusing in the μTCA context, as the primary low voltage supply in a μTCA crate is through a unit called the power module. To limit confusion, we have decided to call the sub-assemblies “mezzanines”.

General Characteristics

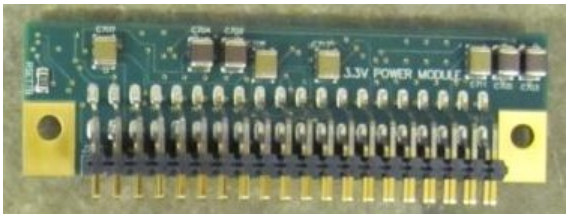
Except for the MMC, the rest of the mezzanines contain an EEPROM (either 24AA02E48T or 25AA02E48T). This 2kbit (256 byte) EEPROM is pre-programmed with a unique MAC address at the factory. The top half of the EEPROM is made non-reprogrammable, so the MAC address is protected, the bottom half is writable. During commissioning of each mezzanine, additional information is programmed into the EEPROM. The EEPROM should not then be modified during operations.

The structure of this data is as follows:

```
struct EEPROM_data {
  uint8_t data_format_version; // this is version 1
  uint8_t mezz_type_code;      // see codes in the sections below
  uint8_t mezz_subtype_code;   // see codes in the sections below
  uint8_t mezz_type[16];       // String version of mezzanine type+sub type
                               // (zero -terminated)
  uint8_t serial_number[2];    // construction serial number ([0]+[1]*256)
  uint8_t manu_date[11];       // DAY-MON-YEAR (zero-terminated)
  uint8_t manu_site[16];       // Site of manufacture (zero-terminated)
  uint8_t manu_tester[16];     // Name of tester (zero-terminated)
  uint8_t test_release[8];     // Release version of test code used
  uint8_t notes[56];           // String notes field in this version,
                               //  available for future use if needed
};
```

This structure is aligned at address zero of the EEPROM.

Power Mezzanine



The primary power mezzanines are designed to convert up to 20W of power from the bulk 12V input power, either as 20A of 1V or 6A of 3.3V power. Each mezzanine contains two LTM4601AV µmodule power converters, one operating as a slave to the other. The LTM4601AV supports voltage-margining, which raises or lowers the voltage around the nominal voltage. This variation is useful to test aging effects and identify possible weak components or boards. On the power mezzanines, the voltage margin is set as ±5%.

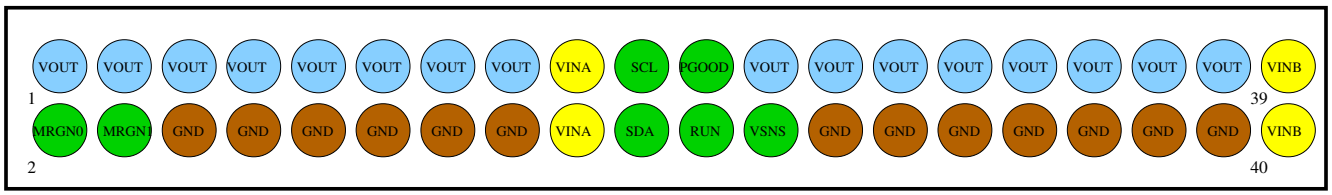
The output voltage is set by a feedback resistor, allowing the same mezzanine to provide any of many voltages by an appropriate resistor setting. Two particular values are used in the uHTR:

- 3.3V power mezzanine
  - R_{SET} = 6.62 k
  - `mezz_type_code`=1
  - `mezz_subtype_code`=3
  - `mezz_type` = “PM3.3”
- 1V power mezzanine
  - R_{SET} = 45 k
  - `mezz_type_code`=1
  - `mezz_subtype_code`=1
  - `mezz_type` = “PM1.0”

The power mezzanine supports an I2C bus which contains the following devices:

- Address 1010000 : ID EEPROM
- Address 0100000 : PCA9534BS3 8-bit GPIO device which allows control over the margining functionality (bits 0 and 1), reading the margin bits (bits 2 and 3), and reading the power-good signal (bit 7)
- Address 1101000 : MCP3426A0-E/MC two-channel ADC. Channel 1 is attached to a TMP36GRT temperature sensor and channel 2 to the output voltage of the µmodules.

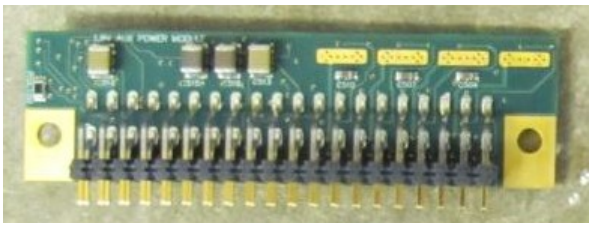
The mezzanine connector is a 40-pin, dual-row 0.1” rectangular connector.



Pins	Function
VOUT	Output voltage
VINA, VINB	Input (12V) power to the master (A) and slave (B) micromodules
SDA/SCL	(I/O) I2C control lines
MRGN0, MRGN1	(Input) Set MRGN0=3V, MRGN1=GND for margin down Set MRGN0=GND, MRGN1=3V for margin up Any other values, provide nominal voltage
PGOOD	(Output) Open-drain, active-high signal that the power is good. On-mezzanine pull-up to 5V through a 100k resistor.
RUN	(Input) Active high (2V to 5V) control to turn on the power converters. Pulled high on the mezzanine through a 100k resistor
VSNS	(Input) Sense input to be tied to the voltage plane on the carrier board

The testing program for the power mezzanine produces (what).

Auxiliary Power Mezzanine



The auxiliary power mezzanines are designed to provide a range of separate voltages at a lower total current than the power mezzanine. Each mezzanine contains one LTM4601AV µmodule power converter and eight linear regulators operating in pairs. The LTM4601AV supports voltage-margining, which raises or lowers the voltage around the nominal voltage, set to ±5% on the LTM4601AV. The linear regulator voltages are set proportionally to the primary regulator, so they follow the margin variations.

There are two APMs used on the uHTR, one where the µmodule provides 2.5V power and one where it provides either 1.8V or 1.5V. In both cases, the linear regulators provide two independent sources of 1V and two of 1.2V power. These lower voltage, lower current supplies are used for supplying the PLL and gigabit transceiver portions of the FPGAs. The 1.8V or 1.5V APM is used for the front FPGA, where a larger current is required – the use of the lower voltage as the supply to the linear regulators saves power and heating of the linear regulators. The 2.5V APM primary output is used in several places on the board and the linear outputs power the back FPGA, which has fewer GTX blocks.

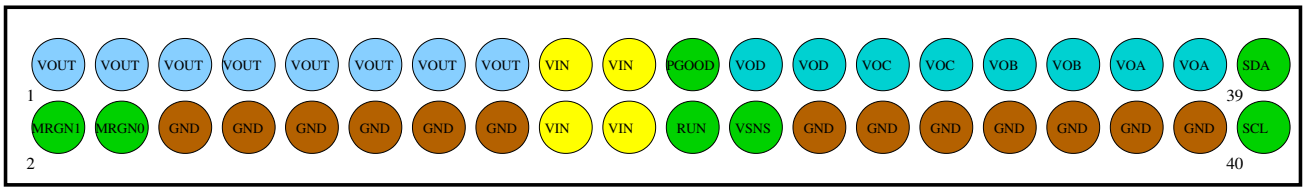
- 2.5V auxillary power mezzanine
  - `mezz_type_code`=2
  - `mezz_subtype_code`=2
  - `mezz_type` = “APM2.5”
- 1.8V auxillary power mezzanine
  - `mezz_type_code`=2
  - `mezz_subtype_code`=1
  - `mezz_type` = “APM1.8”
- 1.5V auxillary power mezzanine
  - `mezz_type_code`=2
  - `mezz_subtype_code`=5
  - `mezz_type` = “APM1.5”

The power mezzanine supports an I2C bus which contains the following devices:

- Address 1010000 : ID EEPROM
- Address 0100000 : PCA9534BS3 8-bit GPIO device which allows control over the margining functionality (bits 0 and 1), reading the margin bits (bits 2 and 3), and reading the power-good signal (bit 7)

- Address 1101000 : MCP3426A0-E/MC two-channel ADC. Channel 1 is attached to a TMP36GRT temperature sensor and channel 2 to the output voltage of the µmodule.

The mezzanine connector is a 40-pin, dual-row 0.1” rectangular connector.



Pins	Function
VOUT	Primary output voltage (2.5V or 1.8V)
VOA	Secondary output voltage (1.2V)
VOB	Secondary output voltage (1.2V)
VOC	Secondary output voltage (1.0V)
VOD	Secondary output voltage (1.0V)
VIN	Input (12V) power
SDA/SCL	(I/O) I2C control lines
MRGN0, MRGN1	(Input) Set MRGN0=3V, MRGN1=GND for margin down Set MRGN0=GND, MRGN1=3V for margin up Any other values, provide nominal voltage
PGOOD	(Output) Open-drain, active-high signal that the power is good. On-mezzanine pull-up to 5V through a 100k resistor.
RUN	(Input) Active high (2V to 5V) control to turn on the power converters. Pulled high on the mezzanine through a 100k resistor
VSNS	(Input) Sense input for the primary output (2.5V/1.8V) to be tied to the voltage plane on the carrier board

The testing program for the power mezzanine produces (what).

FLASH Mezzanine

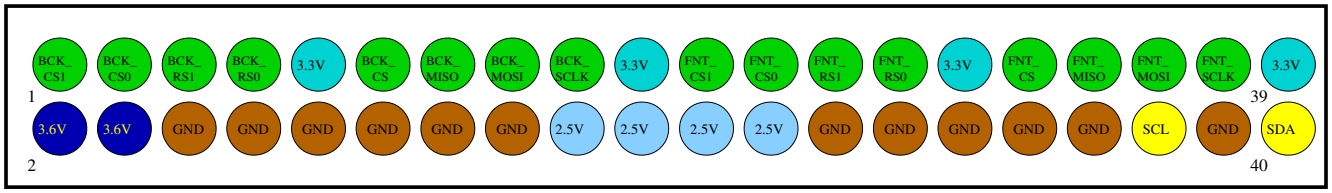
The FLASH mezzanine contains the SPI FLASH modules which contain the primary and backup FPGA configurations and which can also contain user data. The mezzanine can contain a total of eight 128Mb (M25P128) FLASH memories. The memories are accessed via SPI interfaces. There are separate SPI interfaces for the two FPGAs on the uHTR.

- FLASH Mezzanine
  - `mezz_type_code`=3
  - `mezz_subtype_code`=(# of FLASH chips for back FPGA)+8*(# of FLASH chips for the front FPGA)
  - `mezz_type` = "FLASH[(# back),(# front)]"

The power mezzanine supports an I2C bus which contains a single device:

- Address 1010000 : ID EEPROM

The mezzanine connector is a 40-pin, dual-row 0.1" rectangular connector.



Pins	Function
3.3V	3.3V power supply, used for on-board power purposes. If this is connected to the management power of the AMC card, the 3.6V pin should be supplied by payload power.
2.5V/VIO	2.5V power supply, used for I/O on and off the board (could be a different voltage in a different application)
3.6V	3.6V power which is diode-reduced to 3.3V for on-board power purposes. Should be provided from payload power if 3.3V comes from management power.
SDA/SCL	(I/O) I2C control lines
BCK_CS1	(Input, active-low) SPI select line for the second "User" FLASH memory for the back FPGA. Only one of BCK_CS, BCK_CS0, BCK_CS1 should be active at a single time.
BCK_CS0	(Input, active-low) SPI select line for the first "User" FLASH memory for the back FPGA. Only one of BCK_CS, BCK_CS0, BCK_CS1 should be active at a single time.
BCK_CS	(Input, active-low) SPI select line for the FPGA configuration memories. The active configuration memory is chosen by the RS lines. Only one of BCK_CS, BCK_CS0, BCK_CS1 should be active at a single time.
BCK_RS0, BCK_RS1	(Input) Revision-select to pick the active FPGA configuration memory for either programming or FPGA configuration. In the production version of the

	mezzanine, BCK_RS1 is ignored.
FNT_CS1	(Input, active-low) SPI select line for the second “User” FLASH memory for the front FPGA. Only one of FNT_CS, FNT_CS0, FNT_CS1 should be active at a single time.
FNT_CS0	(Input, active-low) SPI select line for the first “User” FLASH memory for the front FPGA. Only one of FNT_CS, FNT_CS0, FNT_CS1 should be active at a single time.
FNT_CS	(Input, active-low) SPI select line for the FPGA configuration memories. The active configuration memory is chosen by the RS lines. Only one of FNT_CS, FNT_CS0, FNT_CS1 should be active at a single time.
FNT_RS0, FNT_RS1	(Input) Revision-select to pick the active FPGA configuration memory for either programming or FPGA configuration. In the production version of the mezzanine, FNT_RS1 is ignored.

CPLD Mezzanine

The CPLD mezzanine is designed to host a moderate-scale Xilinx CPLD and an EEPROM. On the uHTR, the CPLD mezzanine is used to configure the Silicon Labs clock multiplier chips automatically on power-on.

To do:

1. More description
2. Pinout
3. Usage plans

JTAG Mezzanine

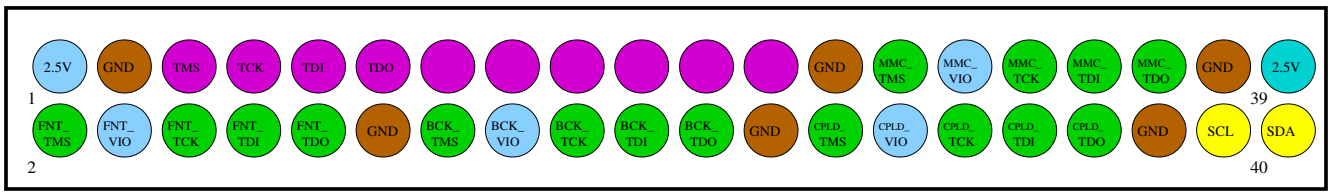
The JTAG mezzanine connects to the JTAG ports of the front and back FPGAs as well as the MMC microcontroller and the CPLD mezzanine. It also connects to the front-panel RJ45 connectors. Two versions of the JTAG mezzanine have been made. One is a simple implementation which provides connectors on the mezzanine for each of the JTAG chains. The second, which is the production version for the uHTR, provides an interface between the JTAG chains and the RJ45 connectors, allowing reprogramming of the FPGAs from the front-panel of a full uTCA crate.

- JTAG Mezzanine
  - `mezz_type_code`=4
  - `mezz_subtype_code`=1
  - `mezz_type` = “JTAG”

The power mezzanine supports an I2C bus which contains a single device:

- Address 1010000 : ID EEPROM

The mezzanine connector is a 40-pin, dual-row 0.1” rectangular connector.



Pins	Function
2.5V	2.5V power supply
***X*_VIO**	I/O voltage for the given JTAG chain
***X*_TMS, *X*_TCK, *X*_TDO, *X*_TDI**	JTAG signals for the given JTAG chain
SDA/SCL	(I/O) I2C control lines for the id EEPROM
TMS, TCK, TDO, TDI	Master JTAG signals (from the front panel)
Magenta Pins	Control pins connected to the front-panel JTAG pins

MMC Mezzanine

The MMC mezzanine (formally, this means “Mezzanine Management Controller Mezzanine”), provides the µTCA-standard control functionality required to identify the power requirements of the AMC and carry out the necessary house-keeping functions. The MMC design and base firmware was provided by the University of Wisconsin group. The uHTR MMC mezzanine is a re-formatting of the design into a compact board which can be easily reused on many AMC cards.