Micro−Research Finland Oy

Event System with Delay Compensation

VME-EVM-300,
mTCA-EVM-300,
VME-EVR-300,
mTCA-EVR-300,
PCIe-EVR-300DC

Technical Reference

VME-EVM-300 Firmware 220C0207
mTCA-EVM-300 Firmware 280E0207
VME-EVR-300 Firmware 12090207
PCIe-EVR-300DC Firmware 170A0207
mTCA-EVR-300DC Firmware 18190207
mTCA-EVR-300RF Firmware 18110207

November 27, 2025
http://www.mrf.fi/

Contents

1. The MRF Timing System

The MRF Timing System provides a complete timing distribution system including timing signal generation with only a few components.

The system is capable of generating and synchronizing frequencies, trigger signals and sequences of events, etc. synchronous to an externally provided master clock reference and mains voltage phase signal. Support for timestamps makes the system a global timebase and allows attaching timestamps to collected data and performed actions.

1.1. Timing System Topology

As a basic setup the timing system consists of an Event Generator (EVG), the distribution layer (Fan-Out) and Event Receivers (EVR). With the active delay compensation feature the Event Generator and distribution layer have been integrated into a single product, the Event Master (EVM).

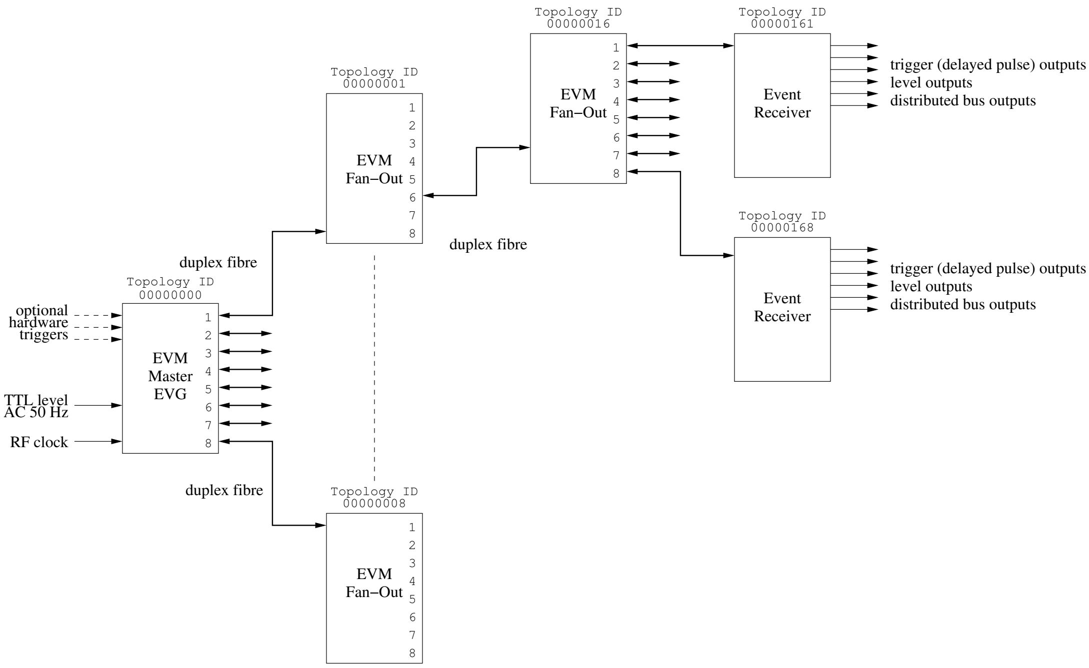

Figure 1: Timing System Topology

1.1.1. Topology ID

Each device in the timing system is given a unique identifier, the Topology ID. The master EVM is given ID 0x00000000. The downstream devices are given IDs with the least significant four bits representing the port number the device is connected to. Each EVM left shifts its own ID by four bits and assigns the downstream port number to the lowest four bits to form the topology ID for the downstream devices in the next level. The topology IDs are represented above the devices in the example layout in figure 1.

1.2. Active Delay Compensation

Delay compensation is achieved in measuring the propagation delay of events from the delay compensation master EVM through the distribution network up to the Event Receivers. At the last stage the EVR is aware of the delay through the network and adjusts an internal FIFO depth to match a programmed target delay value.

1.2.1. Timing System Master

The top node in the Timing System has the important task to generate periodic beacon events and to initialise and send out delay compensation data using the segmented data buffer. Only the top node can be the master and only one master is allowed in the system, all other EVMs have to be initialised in fan-out mode.

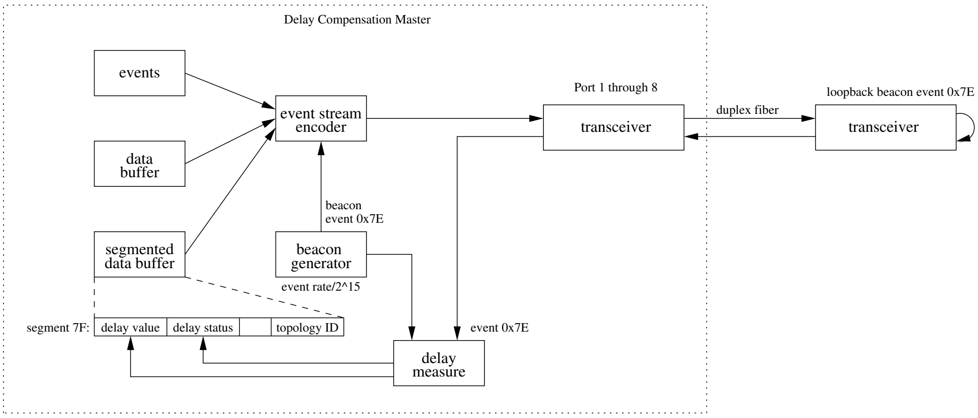

Figure 2: Timing System Master

The beacon generator sends out the beacon event (code 0x7e) at a rate of event clock/2^{15}. When the next node receives the beacon it sends it immediately back to the master which measures the propagation delay of the beacon event. The delay measurement precision improves with time and takes up to 15 minutes to stabilise. The delay value (half of the loop delay) and delay status for each SFP port is sent out using the segmented data buffer. In case the link returning the beacon (receiving side of port 1 through 8) is lost the measurement value is reset and the path delay value status is invalidated. Also if the delay value between consecutive measurements varies significantly (by more than +/- 4 event clock cycles) the delay measurement and delay value status is reset.

1.2.2. Timing System Fan-Out

In EVMs configured as fan-outs (DCMST = 0 and BCGEN = 0) beacons from the Timing System Master are received by the port U transceiver. The recovered event clock from the transceiver is filtered by a clock cleaner external to the FPGA. A FIFO separates the cleaned event clock domain from the recovered clock domain. The depth of the FIFO is kept constant by adjusting the phase of the cleaned clock. Beacon events are propagated through the fan-out and the propagation delay from port U to the fan-out ports 1 through 8 is measured. This delay value can be read from the IntDCValue register.

Further on the beacon events get sent out on the fan-out ports and returned by the next level of fan-outs or event receivers. The loop delay for each port gets measured and the individual port delay values (half of the loop delay value) can be read from the registers Port1DCValue through Port8DCValue. Similar to the timing system master if the link returning the beacon (receiving side of port 1 through 8) is lost the measurement value is reset and the path delay value status is invalidated.

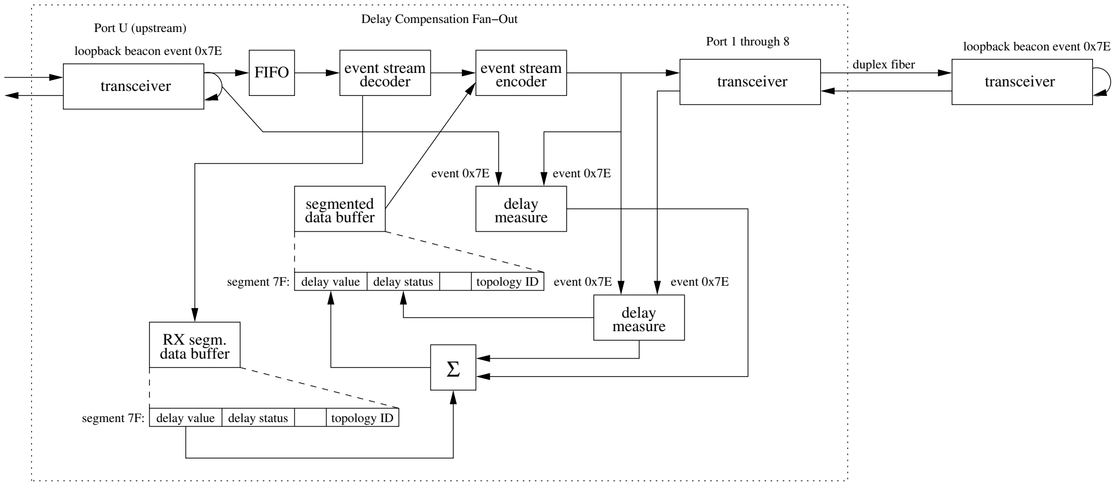

Figure 3: Timing System Fan-Out

The fan-out receives the delay compensation segment on the segmented data buffer. This segment contains information about the delay value from the Timing System Master up to this fan-out and the delay value quality. The integrated delay value from the Timing System Master up to the fan-out can be retrieved from the UpDCValue register. The fan-out modifies the delay value and delay status fields in the DC segment for each port and sends out the new DC segments through ports 1 through 8.

1.2.3. Timing System Event Receiver

The Event Receiver receives the beacon event and returns it back immediately. Based on the recovered event clock from the gigabit transceiver the event receiver generates a local phase shifted and cleaned event clock. The majority of the event receiver logic is running in the cleaned event clock domain. A delay compensation FIFO separates the transceiver receive logic from the main event receiver logic.

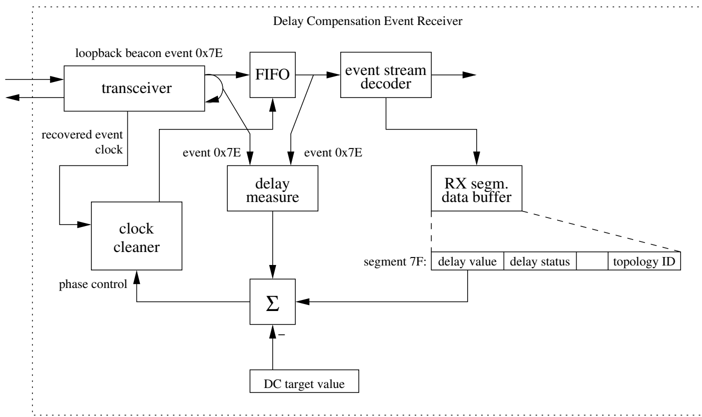

Figure 4: Timing System Event Receiver

The delay compensation segment of the segmented data buffer provides information of the fiber path delay from the timing master up to the event receiver. The event receiver adjusts the phase of the cleaned event clock to control the depth of the delay compensation FIFO. When in delay compensation mode (DCENA = 1) the event receiver controls the FIFO depth such that the sum of the FIFO delay and the data path delay matches the delay compensation target value set in the DCTarget register.

If the event receiver is not in delay compensation mode (DCENA = 0) then the FIFO depth is adjusted to match the target delay value ignoring the received data path delay value altogether.

1.3. Event Stream Details

The structure of the event stream is described to help understand the functioning of the event system. The event stream should be considered as a continuous flow of event frames which consist of two bytes, the event code and distributed bus data byte.

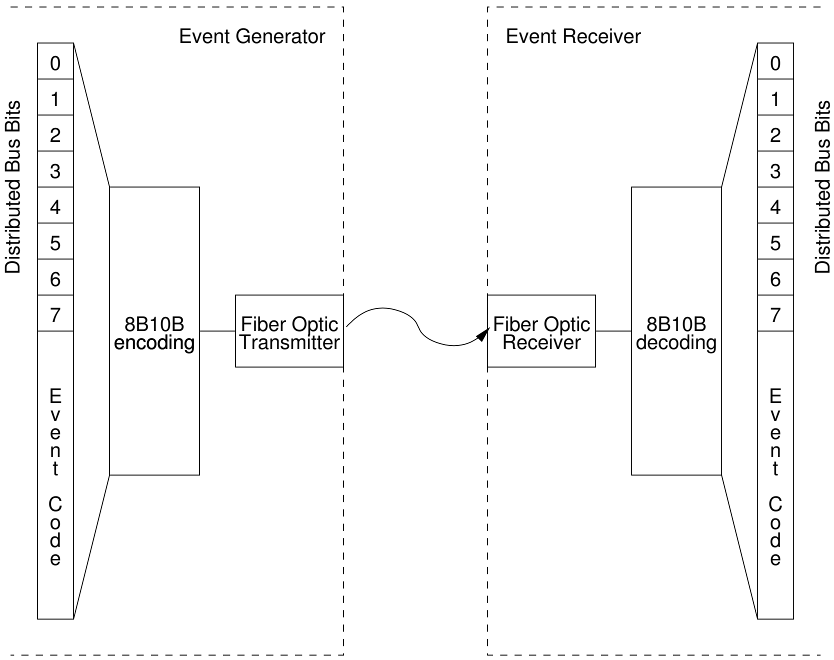

Figure 5: Event Frame

1.3.1. Event Codes

There are 256 event codes from which a few have special functions. The special function event codes are listed below. Only one event code may be transferred at a time. If there is no event code to be transferred, the null event code (0x00) is transmitted. Every now and then a special 8B10B character K28.5 is transmitted instead of the null event code. The K28.5 comma character is transmitted to allow the event receivers to synchronise on the correct word boundary on the serial bit stream.

Table 1: Event Codes

Event Code	Code Name	EVG Function	EVR Function
0x00	Null Event Code	-	-
0x01 – 0x6F	-	User Defined	User Defined
0x70	Seconds ‘0’	-	Shift in ‘0’ to LSB of Seconds Shift Register

Event Code	Code Name	EVG Function	EVR Function
0x71	Seconds '1'	-	Shift in '1' to LSB of Seconds Shift Register
0x72 – 0x78	-	User Defined	User Defined
0x79	Stop Event Log	-	Stop writing events to Event log
0x7A	Heartbeat	-	Reset Heartbeat Monitor
0x7B	Synchronise Prescalers	-	Synchronise Prescaler Outputs
0x7C	Timestamp Counter Increment	-	Increment Timestamp Counter
0x7D	Timestamp Counter Reset	-	Reset Timestamp Counter
0x7E	Beacon event	Delay Compensation signal	Delay Compensation signal
0x7F	End of Sequence	Stop Sequence	-
0x80-FF	-	User Defined	User Defined

1.3.2. Protocol Details

The MRF Timing system protocol is based on 8B10B encoded characters. Two characters are transmitted on every event clock cycle. The first encoded byte is an event code and the second encoded byte is shared by the distributed bus and synchronous data buffer.

Event codes are encoded as characters D01.0 through D31.7. Character D00.0 is reserved and transmitted when there is no event code to transmit. To synchronize the receivers every fourth empty event slot is transmitted as a synchronisation character K28.5.

The example below shows the following details:

- Synchronisation characters at cycle 0, 4, 8, 12 and 20. Please not that event code transmission overrides synchronisation character transmission at cycle 16
- Event code transmissions at following cycles:
  - 2 - Beacon event (beacon event code is reserved and shall not be used for user events)
  - 6 – User event code 0x10
  - 16 – User event code 0x20
- A clock with frequency event clock / 4 on distributed bus bit zero:
  - '0' at cycle 0, 4, 8, 12, 16, 20
  - '1' at cycle 2, 6, 10, 14, 18, 22
- Data packet transmission on the synchronous data bus:
  - Four bytes 0xC0FFEE99 are transmitted to address 0x0A0 of the data buffer

Table 2: Event Protocol Example

Cycle	Event slot	Distributed Bus / Data slot
0	K28.5 Sync character, no event	D00.0 Distributed bus byte
1	D00.0 Null event	D00.0 Data buffer, null data
2	D30.3 Beacon event	D01.0 Distributed bus byte
3	D00.0 Null event	D00.0 Data buffer, null data
4	K28.5 Sync character, no event	D00.0 Distributed bus byte
5	D00.0 Null event	K28.2 Segmented data buffer, start of transfer
6	D16.0 Event code 0x10	D01.0 Distributed bus byte
7	D00.0 Null event	D10.0 Data buffer, segment address 0xA0
8	K28.5 Sync character, no event	D00.0 Distributed bus byte
9	D00.0 Null event	D00.6 Data buffer, data byte 0xC0
10	D00.0 Null event	D01.0 Distributed bus byte
11	D00.0 Null event	D31.7 Data buffer, data byte 0xFF
12	K28.5 Sync character, no event	D00.0 Distributed bus byte
13	D00.0 Null event	D14.7 Data buffer, data byte 0xEE
14	D00.0 Null event	D01.0 Distributed bus byte
15	D00.0 Null event	D25.4 Data buffer, data byte 0x99
16	D00.1 Event code 0x20	D00.0 Distributed bus byte
17	D00.0 Null event	K28.1 Data buffer, end transfer
18	D00.0 Null event	D01.0 Distributed bus byte
19	D00.0 Null event	D28.7 Data buffer, checksum MSB 0xFFFF – 0xA0 – 0xC0 – 0xFF – 0xEE – 0x99 = 0xFC19, MSB 0xFC
20	K28.5 Sync character, no event	D00.0 Distributed bus byte
21	D00.0 Null event	D25.0 Data buffer, checksum LSB 0x19
22	D00.0 Null event	D01.0 Distributed bus byte
23	D00.0 Null event	D00.0 Data buffer, null data

1.4. Migrating to 0207 Firmware

It is important to notice that firmware version 0207 for both the EVM and EVR has some changes that are incompatible with earlier firmware versions. All components in a system have to be running either firmware 0207 or later firmware.

Major changes include:

- The delay compensation data segment has been moved from the beginning of the buffer (segment 0) to the last segment address (0x7f).
- The original non-segmented data buffer has been returned to its original memory location and the segmented data buffer has been given new addresses. The segmented data buffer is now using the K28.2 K-character as a start symbol whereas K28.0 remains for the standard data buffer. In addition to this the segmented data buffer now has a receive data length register for each segment.

- The beacon event has been given a new event code 0x7e not to overlap with the heartbeat event 0x7a.

With these changes the system should remain compatible with the previous non-DC firmware allowing the use of older EVRs in the delay compensated environment if they are specified for the event clock rate in use naturally without support for delay compensation.

Also “DC” event receivers can be used in earlier generation systems.

2. Event Master

The Event Generator is responsible of creating and sending out timing events to an array of Event Receivers. High configurability makes it feasible to build a whole timing system with a single Event Generator without external counters etc.

Events are sent out by the event generator as event frames (words) which consist of an eight bit event code and an eight bit distributed bus data byte. The event transfer rate is derived from an external RF clock or optionally an on-board clock generator. The optical event stream transmitted by the Event Generator is phase locked to the clock reference.

There are several sources of events: trigger events, sequence events, software events and events received from an upstream Event Generator. Events from different sources have different priority which is resolved in a priority encoder.

In addition to events the Event Generator enables the distribution of eight simultaneous signals sampled with the event clock rate, the distributed bus. Distributed bus signals may be provided externally or generated on-board by programmable multiplexed counters.

2.1. Distributed Bus

The distributed bus allows transmission of eight simultaneous signals with half of the event clock rate time resolution (20 ns at 100 MHz event clock rate). The source for distributed bus signals may come from an external source or the signals may be generated with programmable multiplexed counters (MXC) inside the event generator. The distributed bus signals may be programmed to be available as hardware outputs on the event receiver.

2.2. Trigger Events

There are eight trigger event sources that send out an event code on a stimulus. Each trigger event has its own programmable event code register and various enable bits. The event code transmitted is determined by contents of the corresponding event code register. The stimulus may be a detected rising edge on an external signal or a rising edge of a multiplexed counter output.

Trigger Event 0 has also the option of being triggered by a rising edge of the AC mains voltage synchronization logic output signal.

The external inputs accept TTL level signals. The input logic is edge sensitive and the signals are synchronized internally to the event clock.

2.3. Upstream Events

Event Generators may be cascaded. The event generator receiver includes a first-in-first-out (FIFO) memory to synchronize incoming events which may be synchronized to a clock unrelated to the event clock. Usually there are no events in the FIFO. An event code from an upstream EVG is transmitted as soon as there is no other event code to be transmitted.

2.4. Event Sequencer

Event sequencers provide a method of transmitting or playing back sequences of events stored in random access memory with defined timing. In the event generator there are two event sequencers. 8-bit event

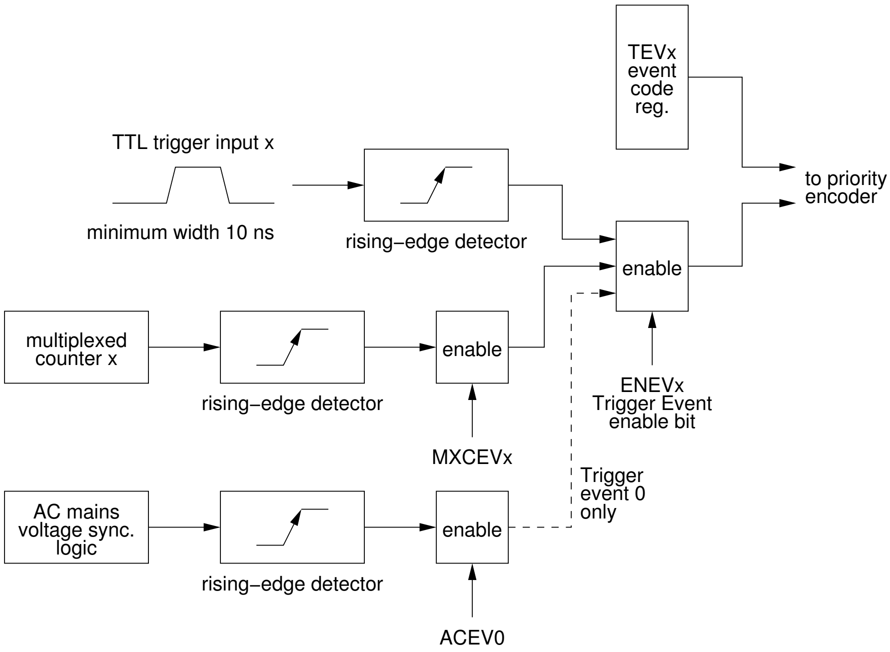

Figure 6: Trigger Event

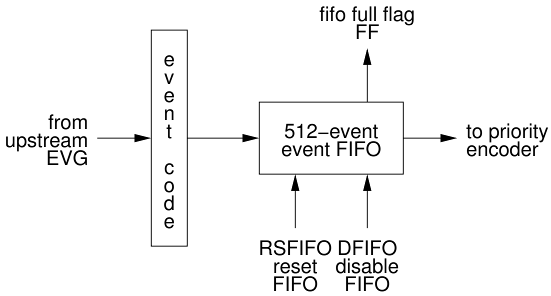

Figure 7: Upstream Events

codes are stored in a RAM table each attached with a 32-bit timestamp relative to the start of sequence. Both sequencers can hold up to 2048 event code – timestamp pairs.

The contents of a sequencer RAM may be altered at any time, however, it is recommended only to modify RAM contents when the RAM is disabled. The sequencer runs at the event clock rate. Starting with firmware version 0200 a mask field has been added. Bits in the mask field allow masking events from being send out based on external signal input states or software mask bits.

The Sequencers may be triggered from several sources including software triggering, triggering on a multiplexed counter output or AC mains voltage synchronization logic output.

The sequencers are enabled by writing a ‘1’ bit to SQxEN in the Sequence RAM control Register. The RAMs may be disabled any time by writing a ‘1’ to SQxDIS bit. Disabling sequence RAMs does not reset the RAM address and timestamp registers. By writing a ‘1’ to the bit SQxRES in the Control Register the sequencer is both disabled and the RAM address and timestamp register is reset.

When the sequencer is triggered the internal event address counter starts counting. The counter value is compared to the event address of the next event in the RAM table. When the counter value matches or is greater than the timestamp in the RAM table, the attached event code is transmitted. The time offset

ram address	32 bits	8 bits	8 bits	
0	event address / timestamp	mask	code	1. event
1	event address / timestamp	mask	code	2. event
2	event address / timestamp	mask	code	3. event
2047	event address / timestamp	mask	code	2048. event

Figure 8: Sequencer RAM Structure

between two consecutive events in the RAM is allowed to be 1 to 2^{32} sequence clock cycles i.e. the internal event address counter rolls over when to 0 when 0xffffffff is reached.

There are two special event codes which are not transmitted, the null event code 0x00 and end sequence code 0x7f. The null event code may be used if the time between two consecutive events should exceed 2^{32} event clock cycles by inserting a null event with a timestamp value of 0xffffffff. In this case the sequencer time will roll over from 0xffffffff to 0x00000000. The end sequence code resets the sequencer RAM table address and timestamp register and depending on configuration bits, disables the sequencer (single sequence, SQxSNG=1) or restarts the sequence either immediately (recycle sequence, SQxREC=1) or waits for a new trigger (SQxREC=0).

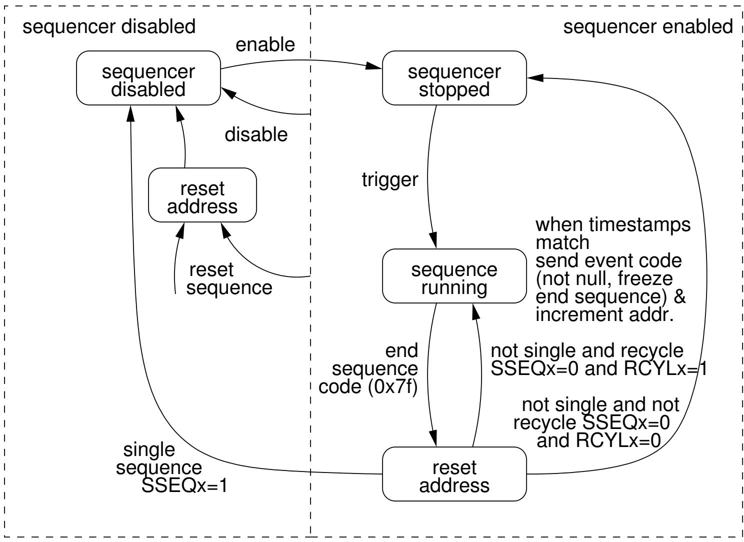

Figure 9: Sequencer RAM Control

2.4.1. Sequencer Interrupt Support

The sequencers provide several interrupts: a sequence start and sequence stop interrupt and a two interrupts based on the position of the playback pointer in the sequencer RAM: a sequence halfway through interrupt and a sequence roll-over interrupt. The sequence start interrupt is issued when a sequencer is in enabled state, gets triggered and was not running before the trigger.

A sequence stop interrupt is issued when the sequence is running and reaches the ‘end of sequence’ code.

2.4.2. Sequence Counting

Starting with version 120207, sequence start/end counters are available for each sequencer.

Sequence start counter (register *SeqRamStartCntx*) is incremented each time a sequence is triggered. Note that in the recycle mode (SQxREC=1), only the start of the first (triggered) sequence is counted - subsequent recycled sequence starts are not counted. Sequence end counter (register *SeqRamEndtCntx*) is incremented each time a sequence reaches the end (code 0x7f). Both counters are reset each time the sequencer is enabled (by setting the SQxEN bit of the *SeqRamCtrl* register).

All the sequence counters are 32-bit numbers so they may eventually overflow and restart counting from zero. When this happens, an interrupt will be sent to the host, if it is enabled. For this purpose, each counter has bits assigned to it in the Interrupt Flag Register (bits IFEOVFx and IFSOVFx) and Interrupt Enable Register (bits IEEOVFx and IESOVFx).

2.4.3. Sequence Countdown Mode

Starting with version 120207, sequence countdown mode is available. In this mode the Sequencer is disabled after a set number of sequence repetitions. The number of repetitions is a 64-bit unsigned integer and can be set for each sequencer separately via two 32-bit registers - *SeqRamRepLowx* (lower 32 bits) and *SeqRamRepHighx* (higher 32-bits). In case both registers are set to 0 (default behavior), this mode is not active and the sequences will be repeated indefinitely, until the sequencer is disabled.

The countdown mode works differently depending on the sequencer configuration bits. In normal triggering mode (SQxSNG=0 and SQxREC=0) the sequence is repeated for the set number of triggers, then the sequencer is disabled. in recycled mode (SQxREC=1), when triggered, the sequence is repeated continuously for the set number of times, before the sequencer is disabled. The single mode (SQxSNG=1) overrides the countdown mode, and the sequence is only generated for one trigger.

2.5. Event Priority

Events in the EVM have certain priority. An event with higher priority is sent out before lower priority events. The event priority is following:

Event source	Priority
Upstream Event	highest
Trigger Event 0	
Trigger Event 1	
Trigger Event 2	
Trigger Event 3	
Event Sequencer 1	
Event Sequencer 2	
Trigger Event 4	
Trigger Event 5	
Trigger Event 6	
Trigger Event 7	
Beacon Event	
Software Event	
Timestamping '0' Event	
Timestamping '1' Event	
Timestamping Second Event	lowest

Each of the sources of the priority encoder has only one buffer stage that allows the pending event to wait for an empty slot. If another event from the same source occurs before the preceding events gets sent out the earlier event is lost.

2.6. Distributed Bus

The bits of the distributed bus are sampled at the event rate from external signals; alternatively the distributed bus signals may be generated by multiplexed counter outputs. If there is an upstream EVG, the state of all distributed bus bits may be forwarded by the EVG.

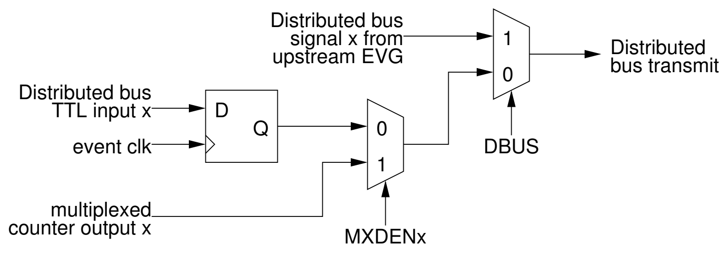

Figure 10: Distributed Bus

2.7. Timestamping Inputs

Starting from firmware version E306 a few distributed bus input signals have dual function: transition board input DBUS5-7 can be used to generate special event codes controlling the timestamping in Event Receivers.

The two clocks, timestamp clock and timestamp reset clock, are assumed to be rising edge aligned. In the EVG the timestamp reset clock is sampled with the falling edge of the timestamp clock. This is to prevent a

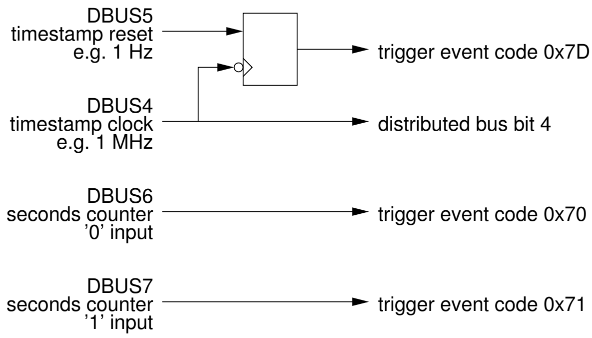

Figure 11: Timestamping Inputs

race condition between the reset and clock signals. In the EVR the reset is synchronised with the timestamp clock.

The two seconds counter events are used to shift in a 32-bit seconds value between consecutive timestamp reset events. In the EVR the value of the seconds shift register is transferred to the seconds counter at the same time the higher running part of the timestamp counter is reset.

The distributed bus event inputs can be enabled independently through the distributed bus event enable register. The events generated through these distributed bus input ports are given lowest priority.

2.8. Timestamp Generator

Logic has been added to automatically increment and send out the 32-bit seconds value. Using this feature requires the two externally supplied clocks as shown above, but the events 0x70 and 0x71 get generated automatically.

After the rising edge of the slower clock on DBUS4, the internal seconds counter is incremented and the 32 bit binary value is sent out LSB first as 32 events 0x70 and 0x71. The seconds counter can be updated by software by using the TSValue and TSControl registers.

2.9. Multiplexed Counters

Eight 32-bit multiplexed counters generate clock signals with programmable frequencies from event clock/2^{32}-1 to event clock/2. Even divisors create 50% duty cycle signals. The counter outputs may be programmed to trigger events, drive distributed bus signals and trigger sequence RAMs. The output of multiplexed counter 7 is hard-wired to the mains voltage synchronization logic.

Each multiplexed counter consists of a 32-bit prescaler register and a 31-bit count-down counter which runs at the event clock rate. When count reaches zero, the output of a toggle flip-flop changes and the counter is reloaded from the prescaler register. If the least significant bit of the prescaler register is one, all odd cycles are extended by one clock cycle to support odd dividers.

Table 3: Multiplexed Counter Prescaler values

Prescaler value	**Duty Cycle**	**Frequency at 125 MHz Event Clock**
0, 1 not allowed	undefined	Undefined

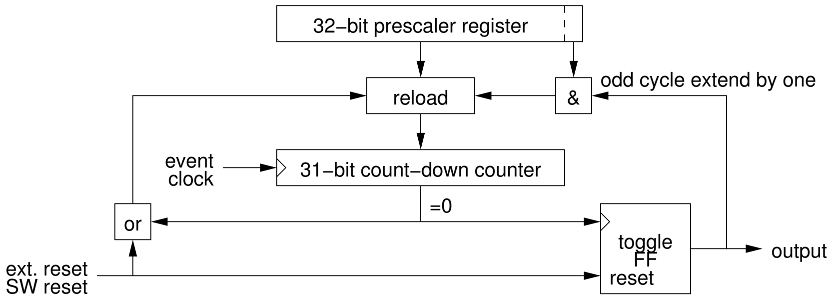

Figure 12: Multiplexed Counters

Prescaler value	Duty Cycle	Frequency at 125 MHz Event Clock
2	50/50	62.5 MHz
3	33/66	41.7 MHz
4	50/50	31.25 MHz
5	40/60	25 MHz
...	...	...
$2^{32}-1$	approx. 50/50	0.029 Hz

The multiplexed counters may be reset by software or hardware input. The reset state is defined by the multiplexed counter polarity register.

2.10. Configurable Size Data Buffer

A buffer of up to 2k bytes can be transmitted over the event link.

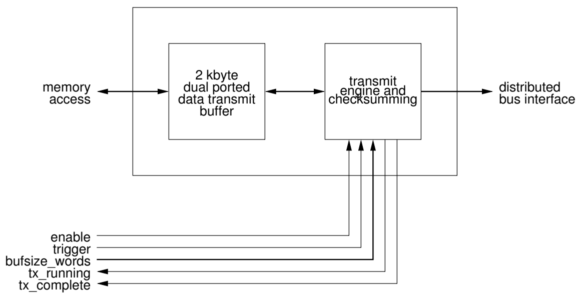

Figure 13: Configurable Size Data Buffer

The data to be transmitted is stored in a 2 kbyte dual-ported memory starting from the lowest address 0. This memory is directly accessible from VME. The transfer size is determined by bufsize register bits in four byte increments. The transmission is triggered by software. Two flags tx_running and tx_complete represent the status of transmission.

Transmission utilises two K-characters to mark the start and end of the data transfer payload, the protocol looks following:

Table 4: Configurable Size Data Buffer Transfer Example

8B10B-character	Description
K28.0	Start of data transfer
Dxx.x	1st data byte (address 0)
Dxx.x	2nd data byte (address 1)
Dxx.x	3rd data byte (address 2)
Dxx.x	4th data byte (address 3)
...	...
Dxx.x	nth data byte (address n-1)
K28.1	End of data
Dxx.x	Checksum (MSB)
Dxx.x	Checksum(LSB)

2.11. Segmented Data Buffer

In addition to the configurable size data buffer a new way to transfer information is provided. With the introduction of active delay compensation in firmware version 0200 the use of the “data buffer mode” has become mandatory. The segmented data buffer memory is divided into 128 segments of 16 bytes each and it is possible to transmit the contents of a single segment or a block of consecutive segments without affecting contents of other segments.

The active delay compensation logic does use the last segment of the segmented data buffer memory for propagating delay compensation information and this segment is reserved for system use.

The data to be transmitted is stored in a 2 kbyte dual-ported memory starting from the lowest address 0. This memory is directly accessible from VME. The transfer size is determined by bufsize register bits in four byte increments. The transmission is triggered by software. Two flags tx_running and tx_complete represent the status of transmission.

Transmission utilises two K-characters to mark the start and end of the data transfer payload, the protocol looks following:

Table 5: Segmented Data Transfer Example

8B10B-character	Description
K28.2	Start of data transfer
Dxx.x	Block address of 16 byte segment
Dxx.x	1st data byte (address 0)
Dxx.x	2nd data byte (address 1)
Dxx.x	3rd data byte (address 2)
Dxx.x	4th data byte (address 3)
...	...
Dxx.x	nth data byte (address n-1)
K28.1	End of data
Dxx.x	Checksum (MSB)

8B10B-character	Description
Dxx.x	Checksum(LSB)

2.11.1. Delay Compensation and Topology ID data

The last segment is reserved for system management and is used to propagate delay compensation and topology data. The contents of the last segment are represented below. Please note that the word values are in little endian byte order.

segment	byte 0 - 3	byte 4 - 7	byte 8 - 11	byte 12 - 15
127	DCDelay	DCStatus	reserved	TopologyID

DCDelay represents the delay from DC master to receiving node. The value is a fixed point number with the point between the two 16 bit words. The delay is measured in event clock cycles.

DCStatus shows the quality of the delay value: 1 - initial lock, 3 - locked with precision < event clock cycle, 7 - fine precision.

TopologyID shows the geographical address of the node.

2.12. Programmable Outputs

All the outputs are programmable: multiplexed counters and distributed bus bits can be mapped to any output. The mapping is shown in table below.

Table 6: Mapping IDs

Mapping ID	Signal
0 to 31	(Reserved)
32	Distributed bus bit 0 (DBUS0)
...	...
39	Distributed bus bit 7 (DBUS7)
40	Multiplexed Counter 0
...	...
47	Multiplexed Counter 7
48	AC trigger logic output
49 to 60	(Reserved)
61	Tri-state output (for bidirectional signals only)
62	Force output high (logic 1)
63	Force output low (logic 0)

2.13. AC Line Synchronisation

The Event Generator provides synchronization to the mains voltage frequency or another external clock. The mains voltage frequency can be divided by an eight bit programmable divider. The output of the divider may be delayed by 0 to 25.5 ms by a phase shifter in 0.1 ms steps to be able to adjust the triggering position relative to mains voltage phase. After this the signal is synchronized to the event clock or the output of multiplexed counter 7. The option to synchronize to an external clock provided in front panel TTL input IN1 or IN2 has been added in firmware version 22000207.

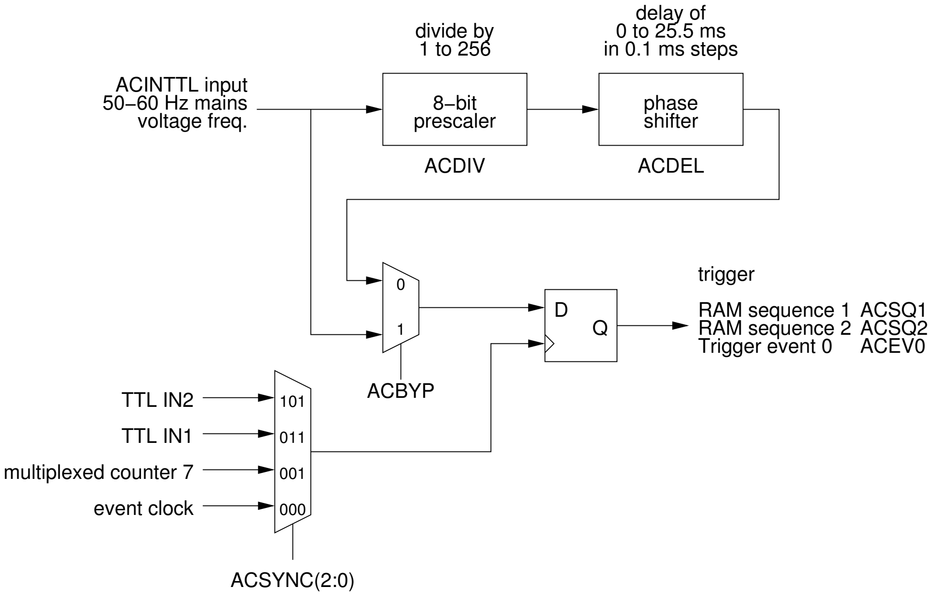

Figure 14: AC Input Logic

The phase shifter operates with a clock of 1 MHz which introduces jitter. If the prescaler and phase shifter are not required this circuit may be bypassed. This also reduces jitter because the external trigger input is sampled directly with the event clock.

2.14. Front Panel TTL Input with Phase Monitoring

Starting from firmware 22000207 a new phase select and phase monitoring feature for the front panel TTL inputs has been added. This allows for monitoring the signal phase and selecting the sampling point of external signals that are phase locked to the event clock.

The external signal is sampled with four phases of the event clock, 0ï¿½, 90ï¿½, 180ï¿½ and 270ï¿½ and synchronized to the event clock. The signal being used for the internal FPGA logic is selected by the PHSEL bits in the phase monitoring register.

The phase monitoring logic detects rising and falling edges of the incoming signal and stores the phase offset in two registers PHRE for the rising edge and PHFE for the falling edge. The contents of the registers are updated on each edge detected and the values can be reset to 0000 for PHRE and 1111 for PHFE by writing a ‘1’ to the PHCLR bit.

Table 7: Phase Monitoring Edge Values

PHRE value	Rising edge position
0000	Reset value, no edge detected
0001	Edge between 180ï¿½ and 270ï¿½

PHFE value	Falling edge position
1111	Reset value, no edge detected
1110	Edge between 180ï¿½ and 270ï¿½

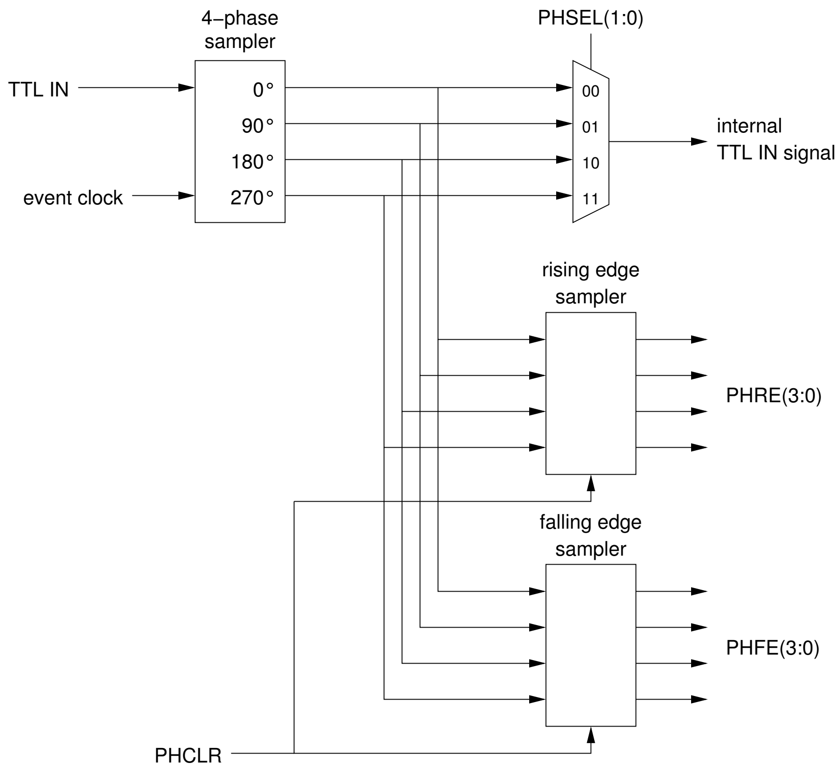

Figure 15: TTL Input Phase Monitoring

PHRE value	Rising edge position
0011	Edge between 90ï¿½ and 180ï¿½
0111	Edge between 0ï¿½ and 90ï¿½
1111	Edge between 270ï¿½ and 0ï¿½

PHFE value	Falling edge position
1100	Edge between 90ï¿½ and 180ï¿½
1000	Edge between 0ï¿½ and 90ï¿½
0000	Edge between 270ï¿½ and 0ï¿½

If the input signal is phase locked to the event clock the phase monitoring values should be stable or toggling between two values if the signal is close to the clock sampling edge. A sampling point as far as possible from the transition point should be selected. Selecting the correct edge is not automated. The edge position of interest should be monitored by the user application and the correct phase should be selected by software.

Table 8: Phase Monitoring Rising Edge Select Values

PHRE value	Rising edge position	PHSEL
0000	Reset value, no edge detected	
0001	Edge between 180ï¿½ and 270ï¿½	01, sample at 90ï¿½
0001/0011	Edge around 180ï¿½	00, sample at 0ï¿½
0011	Edge between 90ï¿½ and 180ï¿½	00, sample at 0ï¿½
0011/0111	Edge around 90ï¿½	11, sample at 270ï¿½
0111	Edge between 0ï¿½ and 90ï¿½	11, sample at 270ï¿½

PHRE value	Rising edge position	PHSEL
0111/1111	Edge around 0ï¿½	10, sample at 180ï¿½
1111	Edge between 270ï¿½ and 0ï¿½	10, sample at 180ï¿½
1111/0001	Edge around 270ï¿½	01, sample at 90ï¿½

Table 9: Phase Monitoring Falling Edge Select Values

PHFE value	Falling edge position	PHSEL
1111	Reset value, no edge detected	
1110	Edge between 180ï¿½ and 270ï¿½	01, sample at 90ï¿½
1110/1100	Edge around 180ï¿½	00, sample at 0ï¿½
1100	Edge between 90ï¿½ and 180ï¿½	00, sample at 0ï¿½
1100/1000	Edge around 90ï¿½	11, sample at 270ï¿½
1000	Edge between 0ï¿½ and 90ï¿½	11, sample at 270ï¿½
1000/0000	Edge around 0ï¿½	10, sample at 180ï¿½
0000	Edge between 270ï¿½ and 0ï¿½	10, sample at 180ï¿½
0000/1110	Edge around 270ï¿½	01, sample at 90ï¿½

In the DC firmware the distributed bus is operating at half rate of the event clock and when using an external clock with an even sub-harmonic the phase of the distributed bus transmission is arbitrary after restarting the system. To overcome this the phase monitoring inputs have a status bit that shows the phase of the distributed bus on the rising edge of the external input. The user can monitor this bit and verify that the phase is correct each time the system is restarted. If the phase is incorrect the phase may be toggled by writing a '1' into the PHTOGG bit in the clock control register.

2.15. Event Clock RF Source

All operations on the event generator are synchronised to the event clock which is derived from an externally provided RF clock. For laboratory testing purposes an on-board fractional synthesiser may be used to deliver the event clock. The serial link bit rate is 20 times the event clock rate. The acceptable range for the event clock and bit rate is shown in the following table.

Table 10: Event Clock Requirement

	Event Clock	Bit Rate
Minimum	50 MHz	1.0 Gb/s
Maximum	142.8 MHz	2.9 Gb/s

During operation the reference frequency should not be changed more than ±100 ppm.

2.16. RF Clock and Event Clock

The event clock may be derived from an external RF clock signal. The front panel RF input is 50 ohm terminated and AC coupled to a LVPECL logic input, so either an ECL level clock signal or sine-wave signal with a level of maximum +10 dBm can be used.

Table 11: RF Input Requirements

Divider	RF Input Frequency	Event Clock	Bit Rate
÷ 1	50 MHz – 142.8 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 2	100 MHz – 285.6 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 3	150 MHz – 428.4 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 4	200 MHz – 571.2 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 5	250 MHz – 714 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 6	300 MHz – 856.8 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 7	350 MHz – 999.6 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 8	400 MHz – 1.142 GHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 9	450 MHz – 1.285 MHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 10	500 MHz – 1.428 GHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 11	550 MHz – 1.571 GHz	50 MHz – 142.8 MHz	1.0 Gb/s – 2.9 Gb/s
÷ 12	600 MHz – 1.6 GHz	50 MHz – 133 MHz	1.0 Gb/s – 2.667 Gb/s
÷ 14	700 MHz – 1.6 GHz *)	50 MHz – 114 MHz	1.0 Gb/s – 2.286 Gb/s
÷ 15	750 MHz – 1.6 GHz *)	50 MHz – 107 MHz	1.0 Gb/s – 2.133 Gb/s
÷ 16	800 MHz – 1.6 GHz *)	50 MHz – 100 MHz	1.0 Gb/s – 2.0 Gb/s
÷ 17	850 MHz – 1.6 GHz *)	50 MHz – 94 MHz	1.0 Gb/s – 1.882 Gb/s
÷ 18	900 MHz – 1.6 GHz *)	50 MHz – 88 MHz	1.0 Gb/s – 1.777 Gb/s
÷ 19	950 MHz – 1.6 GHz *)	50 MHz – 84 MHz	1.0 Gb/s – 1.684 Gb/s
÷ 20	1.0 GHz – 1.6 GHz *)	50 MHz – 80 MHz	1.0 Gb/s – 1.600 Gb/s
÷ 21	1.05 GHz – 1.6 GHz *)	50 MHz – 76 MHz	1.0 Gb/s – 1.523 Gb/s
÷ 22	1.1 GHz – 1.6 GHz *)	50 MHz – 72 MHz	1.0 Gb/s – 1.454 Gb/s
÷ 23	1.15 GHz – 1.6 GHz *)	50 MHz – 69 MHz	1.0 Gb/s – 1.391 Gb/s
÷ 24	1.2 GHz – 1.6 GHz *)	50 MHz – 66 MHz	1.0 Gb/s – 1.333 Gb/s
÷ 25	1.25 GHz – 1.6 GHz *)	50 MHz – 64 MHz	1.0 Gb/s – 1.280 Gb/s
÷ 26	1.3 GHz – 1.6 GHz *)	50 MHz – 61 MHz	1.0 Gb/s – 1.230 Gb/s
÷ 27	1.35 GHz – 1.6 GHz *)	50 MHz – 59 MHz	1.0 Gb/s – 1.185 Gb/s
÷ 28	1.4 GHz – 1.6 GHz *)	50 MHz – 57 MHz	1.0 Gb/s – 1.142 Gb/s
÷ 29	1.45 GHz – 1.6 GHz *)	50 MHz – 55 MHz	1.0 Gb/s – 1.103 Gb/s
÷ 30	1.5 GHz – 1.6 GHz *)	50 MHz – 53 MHz	1.0 Gb/s – 1.066 Gb/s
÷ 31	1.55 GHz – 1.6 GHz *)	50 MHz – 51 MHz	1.0 Gb/s – 1.032 Gb/s
÷ 32	1.6 GHz *)	50 MHz	1.0 Gb/s

*) Range limited by AD9515 maximum input frequency of 1.6 GHz

2.16.1. Fractional Synthesiser

The event master requires a reference clock to be able to synchronise on the incoming event stream sent by the system master. A Micrel (http://www.micrel.com) SY87739L Protocol Transparent Fractional-N Synthesiser with a reference clock of 24 MHz is used. The following table lists programming bit patterns for a few frequencies.

Please note before programming a new operating frequency in the fractional synthesizer the operating frequency (in MHz) has to be set in the UsecDivider register. This is essential as the board's PLL cannot lock if it does not know the frequency range to lock to.

Table 12: Fractional Synthesiser

Event Rate	**Configuration Bit Pattern**	**Reference Output**	**Precision (theoretical)**
142.8 MHz	0x0891C100	142.857 MHz	0
499.8 MHz/4 = 124.95 MHz	0x00FE816D	124.95 MHz	0
499.654 MHz/4 = 124.9135 MHz	0x0C928166	124.907 MHz	-52 ppm
476 MHz/4 = 119 MHz	0x018741AD	119 MHz	0
106.25 MHz (fibre channel)	0x049E81AD	106.25 MHz	0
499.8 MHz/5 = 99.96 MHz	0x025B41ED	99.956 MHz	-40 ppm
50 MHz	0x009743AD	50.0 MHz	0
499.8 MHz/10 = 49.98 MHz	0x025B43AD	49.978 MHz	-40 ppm
499.654 MHz/4 = 124.9135 MHz	0x0C928166	124.907 MHz	-52 ppm
50 MHz	0x009743AD	50.0 MHz	0

2.17. VME-EVM-300 Front Panel Connections

The front panel of the Event Generator is shown in Figure 16.

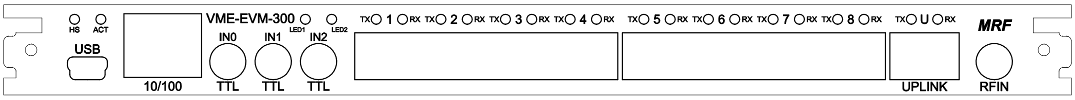

Figure 16: VME-EVM-300 Front Panel

The front panel of the Event Generator includes the following connections and status leds:

Table 13: VME-EVM-300 Front Panel Connections

Connector / Led	**Style**	**Level**	**Description**
HS	Red Led		Module Failure
HS	Blue Led		Module Powered Down
ACT	3-color Led		SAM3X Activity Led
USB	Micro-USB		SAM3X Serial port / JTAG interface
10/100	RJ45		SAM3X Ethernet Interface
IN0	LEMO	TTL	ACIN / TTL0 Trigger input
IN1	LEMO	TTL	Configurable front panel input
IN2	LEMO	TTL	Configurable front panel input
TX 1	LC	optical	Fan-Out Port 1 Transmit (TX 1)
RX 1	LC	optical	Concentrator Port 1 Receiver (RX 1)
TX 2	LC	optical	Fan-Out Port 2 Transmit (TX 2)
RX 2	LC	optical	Concentrator Port 2 Receiver (RX 2)
TX 3	LC	optical	Fan-Out Port 3 Transmit (TX 3)
RX 3	LC	optical	Concentrator Port 3 Receiver (RX 3)
TX 4	LC	optical	Fan-Out Port 4 Transmit (TX 4)
RX 4	LC	optical	Concentrator Port 4 Receiver (RX 4)
TX 5	LC	optical	Fan-Out Port 5 Transmit (TX 5)
RX 5	LC	optical	Concentrator Port 5 Receiver (RX 5)
TX 6	LC	optical	Fan-Out Port 6 Transmit (TX 6)
RX 6	LC	optical	Concentrator Port 6 Receiver (RX 6)
TX 7	LC	optical	Fan-Out Port 7 Transmit (TX 7)
RX 7	LC	optical	Concentrator Port 7 Receiver (RX 7)
TX 8	LC	optical	Fan-Out Port 8 Transmit (TX 8)
RX 8	LC	optical	Concentrator Port 8 Receiver (RX 8)
TX UP	LC	optical	Upstream Transmit Optical Output (TX)

Connector / Led	Style	Level	Description
RX UP	LC	optical	Upstream Receiver Optical Input (RX)
RFIN	LEMO	RF +10 dBm	RF Reference Input

2.17.1. TTL Input Levels

The VME-EVM-300 has three front panel TTL inputs. The inputs are terminated with 50 ohm to ground and are 5V tolerant even when powered down.

Input specifications are following:

parameter	value
connector type	LEMO EPK.00.250.NTN
input impedance	50 ohm
V_{IH}	> 2.3 V
V_{IL}	< 1.0 V

2.18. VME-EVM-300 VME P2 User I/O Pin Configuration

The following table lists the connections to the VME P2 User I/O Pins.

Table 14: VME-EVM-300 VME P2 User I/O Pin Configuration

Pin	Signal
A1	Transition board ID0
A2	Transition board ID1
A3-A10	Ground
A11	Transition board ID2
A12	Transition board ID3
A13-A15	Ground
A16	Transition board handle switch
A17-A26	Ground
A27-A31	+5V
A32	Power control for transition board
C1	Transition board input 0
C2	Transition board input 1
C3	Transition board input 2
C4	Transition board input 3
C5	Transition board input 4
C6	Transition board input 5
C7	Transition board input 6

Pin	Signal
C8	Transition board input 7
C9	Transition board input 8
C10	Transition board input 9
C11	Transition board input 10
C12 – C27	(reserved input)
C28	Transition board input 11
C29	Transition board input 12
C30	Transition board input 13
C31	Transition board input 14
C32	Transition board input 15

2.19. VME-EVM-300 CR/CSR Support

The VME Event Generator module provides CR/CSR Support as specified in the VME64x specification. The CR/CSR Base Address Register is determined after reset by the inverted state of VME64x P1 connector signal pins GA4*-GA0*. In case the parity signal GAP* does not match the GAx* pins the CR/CSR Base Address Register is loaded with the value 0xf8 which corresponds to slot number 31.

After power up or reset the board responds only to CR/CSR accesses with its geographical address. Prior to accessing Event Generator functions the board has to be configured by accessing the boards CSR space.

The Configuration ROM (CR) contains information about manufacturer, board ID etc. to identify boards plugged in different VME slots. The following table lists the required field to locate an Event Generator module.

Table 15: VME-EVM-300 CR/CSR

CR address	Register	EVG
0x27, 0x2B, 0x2F	Manufacturer’s ID (IEEE OUI)	0x000EB2
0x33, 0x37, 0x3B, 0x3F	Board ID	0x4547012C

2.19.1. Function 0 and 1 Registers

The Event Generator specific register are accessed via Function 0 or 1 as specified in the VME64x specification. To enable Function 0, the address decoder compare register for Function 0 in CSR space has to be programmed.

```
vmeCSRWriteADER(3, 0, (slot << 19) | (VME_AM_STD_USR_DATA << 2));

MrfEvgStruct *pEvg;

sysBusToLocalAdrs(VME_AM_STD_USR_DATA, (char *) (slot << 19),
        (void *) pEvg);
```

2.19.2. Function 2 Registers

The Fan-Out/Concentrator specific register are accessed via Function 2 as specified in the VME64x specification. To enable Function 2, the address decoder compare register for Function 2 in CSR space has to be programmed.

2.20. mTCA-EVM-300 Front Panel Connections

The front panel of the Event Generator is shown in Figure 17.

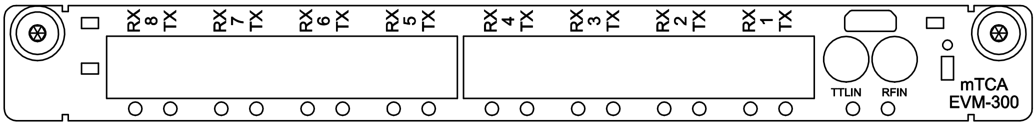

Figure 17: mTCA-EVM-300 Front Panel

The front panel of the Event Generator includes the following connections and status leds:

Table 16: mTCA-EVM-300 Front Panel Connections

Connector / Led	Style	Level	Description
RFIN	LEMO	RF +10 dBm	RF Reference Input
TTLIN	LEMO	TTL	ACIN / TTL0 Trigger input
TX 1	LC	optical	Fan-Out Port 1 Transmit (TX 1)
RX 1	LC	optical	Concentrator Port 1 Receiver (RX 1)
TX 2	LC	optical	Fan-Out Port 2 Transmit (TX 2)
RX 2	LC	optical	Concentrator Port 2 Receiver (RX 2)
TX 3	LC	optical	Fan-Out Port 3 Transmit (TX 3)
RX 3	LC	optical	Concentrator Port 3 Receiver (RX 3)
TX 4	LC	optical	Fan-Out Port 4 Transmit (TX 4)
RX 4	LC	optical	Concentrator Port 4 Receiver (RX 4)
TX 5	LC	optical	Fan-Out Port 5 Transmit (TX 5)
RX 5	LC	optical	Concentrator Port 5 Receiver (RX 5)
TX 6	LC	optical	Fan-Out Port 6 Transmit (TX 6)
RX 6	LC	optical	Concentrator Port 6 Receiver (RX 6)
TX 7	LC	optical	Fan-Out Port 7 Transmit (TX 7)
RX 7	LC	optical	Concentrator Port 7 Receiver (RX 7)
TX 8 (UP)	LC	optical	Upstream Transmit Optical Output (TX)
RX 8 (UP)	LC	optical	Upstream Receiver Optical Input (RX)

2.20.1. TTL Input Levels

The mTCA-EVM-300 has one front panel TTL input. The input is terminated with 50 ohm to ground and is 5V tolerant even when powered down.

Input specifications are following:

parameter	value
connector type	LEMO EPK.00.250.NTN
input impedance	50 ohm
V_{IH}	> 2.3 V
V_{IL}	< 1.0 V

2.21. mTCA-EVM-300 mTCA.4 Backplane Trigger Bus Lines

The EVM can drive and receive all of the eight differential trigger bus lines on the mTCA.4 backplane, ports 17 through 20. The configuration is following:

Table 17: mTCA-EVM-300 mTCA.4 Backplane Trigger Bus Lines

mTCA.4 Pin	EVM Backplane I/O Port
Rx17	0
Tx17	1
Rx18	2
Tx18	3
Rx19	4
Tx19	5
Rx20	6
Tx20	7

The differential line is an input when the BPOutMap register for this line has the value 0x3d (61 decimal).

2.22. mTCA-EVM-300 mTCA.4 Rear Transition Module Signals

The EVM supports driving and receiving signals to/from a RTM. The signal configuration is following:

Table 18: mTCA-EVM-300 RTM Signals

mTCA.4 J30/RP30 Pin	EVM RTM I/O Port
A3/B3	n/a
C3/D3	n/a
E3-/F3+	TBOut8
A4/B4	n/a
C4-/D4+	TBOut9
E4-/F4+	TBOut7
A5	SDA (I2C LED/GPIO control)
B5	SCL (I2C LED/GPIO control)
C5-/D5+	TBIn9

mTCA.4 J30/RP30 Pin	EVM RTM I/O Port
E5-/F5+	TBOut6
A6-/B6+	TBIn8
C6-/D6+	TBIn7
E6-/F6+	TBOut5
A7-/B7+	TBIn6
C7-/D7+	TBIn5
E7-/F7+	TBOut4
A8-/B8+	TBIn4
C8-/D8+	TBIn3
E8-/F8+	TBOut3
A9-/B9+	TBIn2
C9-/D9+	TBIn1
E9-/F9+	TBOut2
A10-/B10+	TBIn0
C10-/D10+	TBOut0
E10-/F10+	TBOut1

2.23. Register Mapping

The mTCA-EVM-300 does use following PCI IDs.

ID name	value	description
PCI Vendor ID	0x10ee	Xilinx
PCI Device ID	0x7011	Kintex 7
PCI Subdevice VID	0x1a3e	Micro-Research Finland Oy
PCI Subdevice DID	0x232c	mTCA-EVM-300

All EVM functions are memory mapped through PCI BAR0.

Address	function	description
0x00000 - 0x0FFFF	EVG	Event generator
0x10000 - 0x1FFFF	FCT	Fan-out registers
0x20000 - 0x2FFFF	EVRD	Downstream Event Receiver
0x30000 - 0x3FFFF	EVRU	Upstream Event Receiver

2.24. EVG Function Register Map

Table 19: Event Generator Register Map

Address	Register	Type	Description
0x000	Status	UINT32	Status Register
0x004	Control	UINT32	Control Register
0x008	IrqFlag	UINT32	Interrupt Flag Register
0x00C	IrqEnable	UINT32	Interrupt Enable Register
0x010	ACControl	UINT32	AC divider control
0x014	ACMap	UINT32	AC trigger event mapping
0x018	SWEvent	UINT32	Software event register
0x01C	PCIMasterIrqEnable	UINT32	PCI Master Interrupt Enable Register
0x020	DataBufControl	UINT32	Data Buffer Control Register
0x024	DBusMap	UINT32	Distributed Bus Mapping Register
0x028	DBusEvents	UINT32	Distributed Bus Timestamping Events Register
0x02C	FWVersion	UINT32	Firmware Version Register
0x030	SegBufControl	UINT32	Segmented Data Buffer Control Register
0x034	TSControl	UINT32	Timestamp event generator control register
0x038	TSValue	UINT32	Timestamp event generator value register
0x040	FPInput	UINT32	Front Panel Input state register
0x044	UnivInput	UINT32	Universal Input state register
0x048	TBInput	UINT32	Transition Board Input state register
0x04C	UsecDivider	UINT32	Divider to get from Event Clock to 1 MHz
0x050	ClockControl	UINT32	Event Clock Control Register
0x060	EvanControl	UINT32	Event Analyser Control Register
0x064	EvanCode	UINT32	Event Analyser Distributed Bus and Event Code Register
0x068	EvanTimeHigh	UINT32	Event Analyser Time Counter (bits 63 – 32)
0x06C	EvanTimeLow	UINT32	Event Analyser Time Counter (bits 31 – 0)
0x070	SeqRamCtrl0	UINT32	Sequence RAM 0 Control Register
0x074	SeqRamCtrl1	UINT32	Sequence RAM 1 Control Register
0x080	FracDiv	UINT32	Micrel SY87739L Fractional Divider Configuration Word
0x0A0	SPIData	UINT32	SPI Data Register
0x0A4	SPIControl	UINT32	SPI Control Register
0x100	EvTrig0	UINT32	Event Trigger 0 Register
0x104	EvTrig1	UINT32	Event Trigger 1 Register
0x108	EvTrig2	UINT32	Event Trigger 2 Register
0x10C	EvTrig3	UINT32	Event Trigger 3 Register

Address	Register	Type	Description
0x110	EvTrig4	UINT32	Event Trigger 4 Register
0x114	EvTrig5	UINT32	Event Trigger 5 Register
0x118	EvTrig6	UINT32	Event Trigger 6 Register
0x11C	EvTrig7	UINT32	Event Trigger 7 Register
0x140	SeqRamStartCnt0	UINT32	Sequence RAM 0 Start Counter Register
0x144	SeqRamStartCnt1	UINT32	Sequence RAM 1 Start Counter Register
0x150	SeqRamEndCnt0	UINT32	Sequence RAM 0 End Counter Register
0x154	SeqRamEndCnt1	UINT32	Sequence RAM 1 End Counter Register
0x160	SeqRamRepLow0	UINT32	Sequence RAM 0 Repetitions Register (LSB)
0x164	SeqRamRepLow1	UINT32	Sequence RAM 1 Repetitions Register (LSB)
0x170	SeqRamRepHigh0	UINT32	Sequence RAM 0 Repetitions Register (MSB)
0x174	SeqRamRepHigh1	UINT32	Sequence RAM 1 Repetitions Register (MSB)
0x180	MXCCtrl0	UINT32	Multiplexed Counter 0 Control Register
0x184	MXCPresc0	UINT32	Multiplexed Counter 0 Prescaler Register
0x188	MXCCtrl1	UINT32	Multiplexed Counter 1 Control Register
0x18C	MXCPresc1	UINT32	Multiplexed Counter 1 Prescaler Register
0x190	MXCCtrl2	UINT32	Multiplexed Counter 2 Control Register
0x194	MXCPresc2	UINT32	Multiplexed Counter 2 Prescaler Register
0x198	MXCCtrl3	UINT32	Multiplexed Counter 3 Control Register
0x19C	MXCPresc3	UINT32	Multiplexed Counter 3 Prescaler Register
0x1A0	MXCCtrl4	UINT32	Multiplexed Counter 4 Control Register
0x1A4	MXCPresc4	UINT32	Multiplexed Counter 4 Prescaler Register
0x1A8	MXCCtrl5	UINT32	Multiplexed Counter 5 Control Register
0x1AC	MXCPresc5	UINT32	Multiplexed Counter 5 Prescaler Register
0x1B0	MXCCtrl6	UINT32	Multiplexed Counter 6 Control Register
0x1B4	MXCPresc6	UINT32	Multiplexed Counter 6 Prescaler Register
0x1B8	MXCCtrl7	UINT32	Multiplexed Counter 7 Control Register
0x1BC	MXCPresc7	UINT32	Multiplexed Counter 7 Prescaler Register
0x400	FPOutMap0	UINT16	Front Panel Output 0 Mapping Register
0x402	FPOutMap1	UINT16	Front Panel Output 1 Mapping Register
0x404	FPOutMap2	UINT16	Front Panel Output 2 Mapping Register
0x406	FPOutMap3	UINT16	Front Panel Output 3 Mapping Register
0x420	BPOutMap0	UINT16	Backplane Output 0 Mapping Register
0x422	BPOutMap1	UINT16	Backplane Output 1 Mapping Register
0x424	BPOutMap2	UINT16	Backplane Output 2 Mapping Register
0x426	BPOutMap3	UINT16	Backplane Output 3 Mapping Register
0x428	BPOutMap4	UINT16	Backplane Output 4 Mapping Register

Address	Register	Type	Description
0x42A	BPOutMap5	UINT16	Backplane Output 5 Mapping Register
0x42C	BPOutMap6	UINT16	Backplane Output 6 Mapping Register
0x42E	BPOutMap7	UINT16	Backplane Output 7 Mapping Register
0x440	UnivOutMap0	UINT16	Universal Output 0 Mapping Register
0x442	UnivOutMap1	UINT16	Universal Output 1 Mapping Register
0x444	UnivOutMap2	UINT16	Universal Output 2 Mapping Register
0x446	UnivOutMap3	UINT16	Universal Output 3 Mapping Register
0x448	UnivOutMap4	UINT16	Universal Output 4 Mapping Register
0x44A	UnivOutMap5	UINT16	Universal Output 5 Mapping Register
0x44C	UnivOutMap6	UINT16	Universal Output 6 Mapping Register
0x44E	UnivOutMap7	UINT16	Universal Output 7 Mapping Register
0x450	UnivOutMap8	UINT16	Universal Output 8 Mapping Register
0x452	UnivOutMap9	UINT16	Universal Output 9 Mapping Register
0x454	UnivOutMap10	UINT16	Universal Output 10 Mapping Register
0x456	UnivOutMap11	UINT16	Universal Output 11 Mapping Register
0x458	UnivOutMap12	UINT16	Universal Output 12 Mapping Register
0x45A	UnivOutMap13	UINT16	Universal Output 13 Mapping Register
0x45C	UnivOutMap14	UINT16	Universal Output 14 Mapping Register
0x45E	UnivOutMap15	UINT16	Universal Output 15 Mapping Register
0x480	TBOutMap0	UINT16	Transition Board Output 0 Mapping Register
0x482	TBOutMap1	UINT16	Transition Board Output 1 Mapping Register
0x484	TBOutMap2	UINT16	Transition Board Output 2 Mapping Register
0x486	TBOutMap3	UINT16	Transition Board Output 3 Mapping Register
0x488	TBOutMap4	UINT16	Transition Board Output 4 Mapping Register
0x48A	TBOutMap5	UINT16	Transition Board Output 5 Mapping Register
0x48C	TBOutMap6	UINT16	Transition Board Output 6 Mapping Register
0x48E	TBOutMap7	UINT16	Transition Board Output 7 Mapping Register
0x490	TBOutMap8	UINT16	Transition Board Output 8 Mapping Register
0x492	TBOutMap9	UINT16	Transition Board Output 9 Mapping Register
0x494	TBOutMap10	UINT16	Transition Board Output 10 Mapping Register
0x496	TBOutMap11	UINT16	Transition Board Output 11 Mapping Register
0x498	TBOutMap12	UINT16	Transition Board Output 12 Mapping Register
0x49A	TBOutMap13	UINT16	Transition Board Output 13 Mapping Register
0x49C	TBOutMap14	UINT16	Transition Board Output 14 Mapping Register
0x49E	TBOutMap15	UINT16	Transition Board Output 15 Mapping Register
0x500	FPInMap0	UINT32	Front Panel Input 0 Mapping Register
0x504	FPInMap1	UINT32	Front Panel Input 1 Mapping Register (VME only)

Address	Register	Type	Description
0x508	FPInMap2	UINT32	Front Panel Input 2 Mapping Register (VME only)
0x520	FPPhMon0	UINT32	Front Panel Input 0 Phase Monitoring Register
0x524	FPPhMon1	UINT32	Front Panel Input 1 Phase Monitoring Register (VME only)
0x528	FPPhMon2	UINT32	Front Panel Input 2 Phase Monitoring Register (VME only)
0x540	UnivInMap0	UINT32	Universal Input 0 Map Register
0x544	UnivInMap1	UINT32	Universal Input 1 Map Register
0x548	UnivInMap2	UINT32	Universal Input 2 Map Register
0x54C	UnivInMap3	UINT32	Universal Input 3 Map Register
0x550	UnivInMap4	UINT32	Universal Input 4 Map Register
0x554	UnivInMap5	UINT32	Universal Input 5 Map Register
0x558	UnivInMap6	UINT32	Universal Input 6 Map Register
0x55C	UnivInMap7	UINT32	Universal Input 7 Map Register
0x560	UnivInMap8	UINT32	Universal Input 8 Map Register
0x564	UnivInMap9	UINT32	Universal Input 9 Map Register
0x568	UnivInMap10	UINT32	Universal Input 10 Map Register
0x56C	UnivInMap11	UINT32	Universal Input 11 Map Register
0x570	UnivInMap12	UINT32	Universal Input 12 Map Register
0x574	UnivInMap13	UINT32	Universal Input 13 Map Register
0x578	UnivInMap14	UINT32	Universal Input 14 Map Register
0x57C	UnivInMap15	UINT32	Universal Input 15 Map Register
0x580	BPInMap0	UINT32	Backplane Input 0 Map Register
0x584	BPInMap1	UINT32	Backplane Input 1 Map Register
0x588	BPInMap2	UINT32	Backplane Input 2 Map Register
0x58C	BPInMap3	UINT32	Backplane Input 3 Map Register
0x590	BPInMap4	UINT32	Backplane Input 4 Map Register
0x594	BPInMap5	UINT32	Backplane Input 5 Map Register
0x598	BPInMap6	UINT32	Backplane Input 6 Map Register
0x59C	BPInMap7	UINT32	Backplane Input 7 Map Register
0x600	TBInMap0	UINT32	Transition Board Input 0 Mapping Register
0x604	TBInMap1	UINT32	Transition Board Input 1 Mapping Register
0x608	TBInMap2	UINT32	Transition Board Input 2 Mapping Register
0x60C	TBInMap3	UINT32	Transition Board Input 3 Mapping Register
0x610	TBInMap4	UINT32	Transition Board Input 4 Mapping Register
0x614	TBInMap5	UINT32	Transition Board Input 5 Mapping Register
0x618	TBInMap6	UINT32	Transition Board Input 6 Mapping Register
0x61C	TBInMap7	UINT32	Transition Board Input 7 Mapping Register

Address	Register	Type	Description
0x620	TBInMap8	UINT32	Transition Board Input 8 Mapping Register
0x624	TBInMap9	UINT32	Transition Board Input 9 Mapping Register
0x628	TBInMap10	UINT32	Transition Board Input 10 Mapping Register
0x62C	TBInMap11	UINT32	Transition Board Input 11 Mapping Register
0x630	TBInMap12	UINT32	Transition Board Input 12 Mapping Register
0x634	TBInMap13	UINT32	Transition Board Input 13 Mapping Register
0x638	TBInMap14	UINT32	Transition Board Input 14 Mapping Register
0x63C	TBInMap15	UINT32	Transition Board Input 15 Mapping Register
0x800 – 0xFFF	DataBuf		Data Buffer Transmit Memory
0x1000 – 0x10FF	configROM		
0x1100 – 0x11FF	scratchRAM		
0x1200 – 0x12FF	SFPEEPROM		Upstream SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1300 – 0x13FF	SFPDIAG		Upstream SFP Transceiver diagnostics (SFP address 0xA2)
0x2000 – 0x27FF	SegBuf		Segmented Data Buffer Transmit Memory
0x8000 – 0xBFFF	SeqRam0		Sequence RAM 0
0xC000 – 0xFFFF	SeqRam1		Sequence RAM 1

2.24.1. Status Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x000	RDB7	RDB6	RDB5	RDB4	RDB3	RDB2	RDB1	RDB0

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x001	TDB7	TDB6	TDB5	TDB4	TDB3	TDB2	TDB1	TDB0

Bit	Function
RDB7	Status of received distributed bus bit 7 (from upstream EVG)
RDB6	Status of received distributed bus bit 6 (from upstream EVG)
RDB5	Status of received distributed bus bit 5 (from upstream EVG)
RDB4	Status of received distributed bus bit 4 (from upstream EVG)
RDB3	Status of received distributed bus bit 3 (from upstream EVG)
RDB2	Status of received distributed bus bit 2 (from upstream EVG)
RDB1	Status of received distributed bus bit 1 (from upstream EVG)
RDB0	Status of received distributed bus bit 0 (from upstream EVG)
TDB7	Status of transmitted distributed bus bit 7
TDB6	Status of transmitted distributed bus bit 6
TDB5	Status of transmitted distributed bus bit 5

Bit	Function
TDB4	Status of transmitted distributed bus bit 4
TDB3	Status of transmitted distributed bus bit 3
TDB2	Status of transmitted distributed bus bit 2
TDB1	Status of transmitted distributed bus bit 1
TDB0	Status of transmitted distributed bus bit 0

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x004	EVGEN	RXDIS	RXPWD	FIFORS		SRST	LEMDE	MXCRES

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x005	BCGEN	DCMST						SRALT

Bit	Function
EVGEN	Event Generator Master enable
RXDIS	Disable event reception
RXPWD	Receiver Power down
FIFORS	Reset RX Event Fifo
SRST	Soft reset IP
LEMDE	Little endian mode (cPCI-EVG-300) 0 – PCI core in big endian mode (power up default) 1 – PCI core in little endian mode
MXCRES	Write 1 to reset multiplexed counters
BCGEN	Delay Compensation Beacon generator enable 0 – Beacon generator disabled 1 – Beacon generator enabled, sends out 0x7E events and delay compensation data. Set only for master EVG in system
DCMST	System Master enable 0 – System Master disabled 1 – System Master enabled – has to be set and only for master EVG in system
SRALT	(reserved)

2.24.2. Interrupt Flag Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x008			IFEOVF1	IFEOVF0			IFSOVF1	IFSOVF0

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x009			IFSOV1	IFSOV0			IFSHF1	IFSHF0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x00A			IFSSTO1	IFSSTO0			IFSSTA1	IFSSTA0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x00B		IFEXT	IFDBUF				IFFF	IFVIO

Bit	Function
IFEOVF1	Sequence RAM 1 end counter overflow interrupt flag
IFEOVF0	Sequence RAM 0 end counter overflow interrupt flag
IFSOVF1	Sequence RAM 1 start counter overflow interrupt flag
IFSOVF0	Sequence RAM 0 start counter overflow interrupt flag
IFSOV1	Sequence RAM 1 sequence roll over interrupt flag
IFSOV0	Sequence RAM 0 sequence roll over interrupt flag
IFSHF1	Sequence RAM 1 sequence halfway through interrupt flag
IFSHF0	Sequence RAM 0 sequence halfway through interrupt flag
IFSSTO1	Sequence RAM 1 sequence stop interrupt flag
IFSSTO0	Sequence RAM 0 sequence stop interrupt flag
IFSSTA1	Sequence RAM 1 sequence start interrupt flag
IFSSTA0	Sequence RAM 0 sequence start interrupt flag
IFEXT	External Interrupt flag
IFDBUF	Data buffer flag
IFFF	RX Event FIFO full flag
IFVIO	Port U Receiver violation flag

2.24.3. Interrupt Enable Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x00C	IRQEN		IEEOVF1	IEEOVF0			IESOVF1	IESOVF0

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x00D			IESOV1	IESOV0			IESHF1	IESHF0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x00E			IESSTO1	IESSTO0			IESSTA1	IESSTA0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x00F		IEEXT	IEDBUF				IEFF	IEVIO

Bit	Function
IRQEN	Master interrupt enable
IEEOVF1	Sequence RAM 1 end counter overflow interrupt enable
IEEOVF0	Sequence RAM 0 end counter overflow interrupt enable
IESOVF1	Sequence RAM 1 start counter overflow interrupt enable
IESOVF0	Sequence RAM 0 start counter overflow interrupt enable

Bit	Function
IESOV1	Sequence RAM 1 sequence roll over interrupt enable
IESOV0	Sequence RAM 0 sequence roll over interrupt enable
IESHF1	Sequence RAM 1 sequence halfway through interrupt enable
IESHF0	Sequence RAM 0 sequence halfway through interrupt enable
IESSTO1	Sequence RAM 1 sequence stop interrupt enable
IESSTO0	Sequence RAM 0 sequence stop interrupt enable
IESSTA1	Sequence RAM 1 sequence start interrupt enable
IESSTA0	Sequence RAM 0 sequence start interrupt enable
IEEXT	External interrupt enable
IEDBUF	Data buffer interrupt enable
IEFF	Event FIFO full interrupt enable
IEVIO	Receiver violation interrupt enable

2.24.4. AC Trigger Control Register

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x011					ACSYN2	ACSYN1	ACBYP	ACSYN0
address	**bit 15**	**bit 14**	**bit 13**	**bit 12**	**bit 11**	**bit 10**	**bit 9**	**bit 8**
0x012	AC Trigger Divider							
address	**bit 7**	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
0x013	AC Trigger Phase Shift							

Bit	Function
ACBYP	AC divider and phase shifter bypass (0 = divider/phase shifter enabled, 1 = divider/phase shifter bypassed)
ACSYN(2:0)	Synchronization select 000 = Event clock 001 = Multiplexed counter 7 output 011 = Front panel TTL input IN1 101 = Front panel TTL input IN2

2.24.5. AC Trigger Mapping Register

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x017	ACM7	ACM6	ACM5	ACM4	ACM3	ACM2	ACM1	ACM0

Bit	Function
ACM7	If set AC circuit triggers Event Trigger 7
ACM6	If set AC circuit triggers Event Trigger 6
ACM5	If set AC circuit triggers Event Trigger 5
ACM4	If set AC circuit triggers Event Trigger 4
ACM3	If set AC circuit triggers Event Trigger 3
ACM2	If set AC circuit triggers Event Trigger 2
ACM1	If set AC circuit triggers Event Trigger 1
ACM0	If set AC circuit triggers Event Trigger 0

2.24.6. Software Event Register

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x01A							SWPEND	SWENA

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x01B	Event Code to be sent out							

Bit	Function
SWPEND	Event code waiting to be sent out (read-only). A new event code may be written to the event code register when this bit reads '0'.
SWENA	Enable software event When enabled '1' a new event will be sent out when event code is written to the event code register.

2.24.7. PCI Master Interrupt Enable Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x01C		PCIIE						

Bit	Function
PCIIE	PCI core interrupt enable (mTCA-EVM-300) This bit is used by the low level driver to disable further interrupts before the first interrupt has been handled in user space

2.24.8. Data Buffer Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x020								

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x021				TXCPT	TXRUN	TRIG	ENA	1

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x022						DTSZ(10:8)		

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x023	DTSZ(7:2)						0	0

Bit	Function
TXCPT	Data Buffer Transmission Complete
TXRUN	Data Buffer Transmission Running – set when data transmission has been triggered and has not been completed yet
TRIG	Data Buffer Trigger Transmission Write ‘1’ to start transmission of data in buffer
ENA	Data Buffer Transmission enable ‘0’ – data transmission engine disabled ‘1’ – data transmission engine enabled
DTSZ	Data Transfer size 4 bytes to 2k in four byte increments

2.24.9. Distributed Bus Mapping Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x024	DBMAP7(3:0)				DBMAP6(3:0)			

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x025	DBMAP5(3:0)				DBMAP4(3:0)			

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x026	DBMAP3(3:0)				DBMAP2(3:0)			

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x027	DBMAP1(3:0)				DBMAP0(3:0)			

Bit	Function
DBMAP7(3:0)	Distributed Bus Bit 7 Mapping: 0 – Off, output logic ‘0’ 1 – take bus bit from external input 2 – Multiplexed counter output mapped to distributed bus bit 3 – Distributed bus bit forwarded from upstream EVG
DBMAP6(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)
DBMAP5(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)
DBMAP4(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)
DBMAP3(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)
DBMAP2(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)
DBMAP1(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)

Bit	**Function**
DBMAP0(3:0)	Distributed Bus Bit 7 Mapping (see above for mappings)

2.24.10. Distributed Bus Event Enable Register

address	**bit 7**	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
0x02B	DBEV7	DBEV6	DBEV5					

Bit	**Function**
DBEV5	Distributed bus input 5 “Timestamp reset” 0x7D event enable
DBEV6	Distributed bus input 6 “Seconds ‘0’” 0x70 event enable
DBEV7	Distributed bus input 7 “Seconds ‘1’” 0x71 event enable

2.24.11. FPGA Firmware Version Register

address	**bit 31**	**bit 30**	**bit 29**	**bit 28**	**bit 27**	**bit 26**	**bit 25**	**bit 24**
0x02C	EVG = 0x2				Form Factor			
address	**bit 23**	**bit 22**	**bit 21**	**bit 20**	**bit 19**	**bit 18**	**bit 17**	**bit 16**
0x02D	Subrelease ID							
address	**bit 15**	**bit 14**	**bit 13**	**bit 12**	**bit 11**	**bit 10**	**bit 9**	**bit 8**
0x02E	Firmware ID							
address	**bit 7**	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
0x02F	Revision ID							

Bit	**Function**
Form Factor	0 – CompactPCI 3U 1 – PMC 2 – VME64x 3 – CompactRIO 4 – CompactPCI 6U 6 – PXIe 7 – PCIe 8 – mTCA.4

Bit	Function
Subrelease ID	For production releases the subrelease ID counts up from 00. For pre-releases this ID is used “backwards” counting down from ff i.e. when approacing release 22000207, we have prereleases 22FF0206, 22FE0206, 22FD0206 etc. in this order.
Firmware ID	00 – Modular Register Map firmware (no delay compensation) 01 – Reserved 02 – Delay Compensation firmware
Revision ID	See end of manual

2.24.12. Segmented Data Buffer Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x030	SADDR(7:0)							

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x031				TXCPT	TXRUN	TRIG	ENA	1

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x032						DTSZ(10:8)		

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x033	DTSZ(7:2)						0	0

Bit	Function
SADDR	Transfer Start Segment Address (16 byte segments)
TXCPT	Data Buffer Transmission Complete
TXRUN	Data Buffer Transmission Running – set when data transmission has been triggered and has not been completed yet
TRIG	Data Buffer Trigger Transmission Write ‘1’ to start transmission of data in buffer
ENA	Data Buffer Transmission enable ‘0’ – data transmission engine disabled ‘1’ – data transmission engine enabled
DTSZ(10:8)	Data Transfer size 4 bytes to 2k in four byte increments

2.24.13. Timestamp Generator Control Register

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x037							TSGLOAD	TSGENA

Bit	Function
TSGLOAD	Timestamp Generator Load new value into Timestamp Counter Write ‘1’ to load new value
TSGENA	Timestamp Generator Enable (‘0’ = disable, ‘1’ = enable)

2.24.14. Microsecond Divider Register

address	bit 15 … bit 0
0x04E	Rounded integer value of 1 μs * event clock

This register shall be written with an integer value of the event clock rate in MHz. For 100 MHz event clock this register should read 100, for 50 MHz event clock this register should read 50. This value is used to set the parameters for the clock cleaner PLL and e.g. for the phase shifter in the AC input logic.

2.24.15. Clock Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x050	PLLLOCK	BWSEL(2:0)				RFSEL(2:0)		

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x051	PHTOGG		RFDIV(5:0)					

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x052							CGLOCK	

Bit	Function
PLLLOCK	Clock cleaner locked
BWSEL2:0	PLL Bandwidth Select (see Silicon Labs Si5317 datasheet) 000 – Si5317, BW setting HM (lowest loop bandwidth) 001 – Si5317, BW setting HL 010 – Si5317, BW setting MH 011 – Si5317, BW setting MM 100 – Si5317, BW setting ML (highest loop bandwidth)
PHTOGG	Distributed bus phase toggle
RFDIV5-0	External RF divider select: 000000 – RF/1 000001 – RF/2 000010 – RF/3 000011 – RF/4 000100 – RF/5 000101 – RF/6 000110 – RF/7 000111 – RF/8 001000 – RF/9 001001 – RF/10 001010 – RF/11 001011 – RF/12 001100 – OFF 001101 – RF/14 001110 – RF/15 001111 – RF/16 010000 – RF/17 010001 – RF/18 010010 – RF/19 010011 – RF/20 010100 – RF/21 010101 – RF/22 010110 – RF/23 010111 – RF/24 011000 – RF/25 011001 – RF/26 011010 – RF/27 011011 – RF/28 011100 – RF/29 011101 – RF/30 011110 – RF/31 011111 – RF/32
RFSEL2-0	RF reference select: 000 – Use internal reference (fractional synthesizer) 001 – Use external RF reference (front panel input through divider) 010 – PXIe 100 MHz clock 100 – Use recovered RX clock, Fan-Out mode 101 – Use external RF reference for downstream ports, internal reference for upstream port, Fan-Out mode, event rate down conversion 110 – PXIe 10 MHz clock through clock multiplier 111 – Recovered clock /2 decimate mode, event rate is halved

Bit	Function
CGLOCK	Micrel SY87739L reference clock locked (read-only)

Please note that after changing the Event clock source the fractional synthesizer control word must be reloaded to initialize an internal reset.

2.24.16. Event Analyser Control Register

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x063				EVANE	EVARS	EVAOF	EVAEN	EVACR

Bit	Function
EVANE	Event Analyser FIFO not empty flag: 0 – FIFO empty 1 – FIFO not empty, events in FIFO
EVARS	Event Analyser Reset 0 – not in reset 1 – reset
EVAOF	Event Analyser FIFO overflow flag: 0 – no overflow 1 – FIFO overflow
EVAEN	Event Analyser enable 0 – Event Analyser disabled 1 – Event Analyser enabled
EVACR	Event Analyser 64 bit counter reset 0 – Counter running 1 – Counter reset to zero.

2.24.17. Sequence RAM Control Registers

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x070							SQ0RUN	SQ0ENA

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x071	SQ0XTR	SQ0XEN	SQ0SWT	SQ0SNG	SQ0REC	SQ0RES	SQ0DIS	SQ0EN

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x072	SQSWMASK(3:0)				SQSWENA(3:0)			

address	bit 7	bit 0
0x073	SQ0TSEL	

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x074							SQ1RUN	SQ1ENA

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x075	SQ1XTR	SQ1XEN	SQ1SWT	SQ1SNG	SQ1REC	SQ1RES	SQ1DIS	SQ1EN

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x076	SQSWMASK(3:0)				SQSWENA(3:0)			

address	bit 7	bit 0
0x077	SQ1TSEL	

Bit	Function
SQxRUN	Sequence RAM running flag (read-only)
SQxENA	Sequence RAM enabled flag (read_only)
SQxSWT	Sequence RAM software trigger, write ‘1’ to trigger
SQxSNG	Sequence RAM single mode
SQxREC	Sequence RAM recycle mode
SQxRES	Sequence RAM reset, write ‘1’ to reset
SQxDIS	Sequence RAM disable, write ‘1’ to disable
SQxEN	Sequence RAM enable, write ‘1’ to enable/arm
SQxXEN	Sequence RAM allow external enable, ‘1’ - allow
SQxXTR	Sequence RAM allow external trigger enable, ‘1’ - allow
SQSWMASK	Sequence RAM SW mask register, the mask bits are common for all RAMS
SQSWENA	Sequence RAM SW enable register, the mask bits are common for all RAMS
SQxTSEL	Sequence RAM trigger select: 0 – trigger from MXC0 1 – trigger from MXC1 2 – trigger from MXC2 3 – trigger from MXC3 4 – trigger from MXC4 5 – trigger from MXC5 6 – trigger from MXC6 7 – trigger from MXC7 16 – trigger from AC synchronization logic 17 – trigger from sequence RAM 0 software trigger 18 – trigger from sequence RAM 1 software trigger 19 – trigger always immediately when enabled 24 – trigger from sequence RAM 0 external trigger 25 – trigger from sequence RAM 1 external trigger 31 – trigger disabled (default after power up)

Bit	Function

2.24.18. SY87739L Fractional Divider Configuration Word

Configuration Word	Frequency with 24 MHz reference oscillator
0x0891C100	142.857 MHz
0x00DE816D	125 MHz
0x00FE816D	124.95 MHz
0x0C928166	124.9087 MHz
0x018741AD	119 MHz
0x072F01AD	114.24 MHz
0x049E81AD	106.25 MHz
0x008201AD	100 MHz
0x025B41ED	99.956 MHz
0x0187422D	89.25 MHz
0x0082822D	81 MHz
0x0106822D	80 MHz
0x019E822D	78.900 MHz
0x018742AD	71.4 MHz
0x0C9282A6	62.454 MHz
0x009743AD	50 MHz
0x0C25B43AD	49.978 MHz
0x0176C36D	49.965 MHz

2.25. SPI Configuration Flash Registers

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x0A3	SPIDATA(7:0)							

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x0A7	E	RRDY	TRDY	TMT	TOE	ROE	OE	SSO

Bit	Function
SPIDATA(7:0)	Read SPI data byte / Write SPI data byte
E	Overrun Error flag
RRDY	Receiver ready, if ‘1’ data byte waiting in SPI_DATA
TRDY	Transmitter ready, if ‘1’ SPI_DATA is ready to accept new transmit data byte
TMT	Transmitter empty, if ‘1’ data byte has been transmitted
TOE	Transmitter overrun error

Bit	Function
ROE	Receiver overrun error
OE	Output enable for SPI pins, ‘1’ enable SPI pins
SSO	Slave select output enable for SPI slave device, ‘1’ device selected

2.25.1. Event Trigger Registers

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x102								EVEN0

address	bit 7 … bit 0
0x103	EVCD0(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x106								EVEN1

address	bit 7 … bit 0
0x107	EVCD1(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x10A								EVEN2

address	bit 7 … bit 0
0x10B	EVCD2(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x10E								EVEN3

address	bit 7 … bit 0
0x10F	EVCD3(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x112								EVEN4

address	bit 7 … bit 0
0x113	EVCD4(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x116								EVEN5

address	bit 7 … bit 0
0x117	EVCD5(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x11A								EVEN6

address	bit 7 … bit 0
0x11B	EVCD6(7:0)

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x11E								EVEN7

address	bit 7	bit 0
0x11F	EVCD7(7:0)	

Bit	Function
EVENx	Enable Event Trigger x
EVCDx	Event Trigger Code for Event trigger x

2.25.2. Multiplexed Counter Registers

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x180	MXC0	MXCP0						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x183	MX0EV7	MX0EV6	MX0EV5	MX0EV4	MX0EV3	MX0EV2	MX0EV1	MX0EV0

address	bit 31	bit 0
0x184	Multiplexed Counter 0 Prescaler	

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x188	MXC1	MXCP1						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x18B	MX1EV7	MX1EV6	MX1EV5	MX1EV4	MX1EV3	MX1EV2	MX1EV1	MX1EV0

address	bit 31	bit 0
0x18C	Multiplexed Counter 1 Prescaler	

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x190	MXC2	MXCP2						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x193	MX2EV7	MX2EV6	MX2EV5	MX2EV4	MX2EV3	MX2EV2	MX2EV1	MX2EV0

address	bit 31	bit 0
0x194	Multiplexed Counter 2 Prescaler	

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x198	MXC3	MXCP3						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x19B	MX3EV7	MX3EV6	MX3EV5	MX3EV4	MX3EV3	MX3EV2	MX3EV1	MX3EV0

address	bit 31	bit 0
0x19C	Multiplexed Counter 3 Prescaler	

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x1A0	MXC4	MXCP4						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x1A3	MX4EV7	MX4EV6	MX4EV5	MX4EV4	MX4EV3	MX4EV2	MX4EV1	MX4EV0

address	bit 31 … bit 0
0x1A4	Multiplexed Counter 4 Prescaler

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x1A8	MXC5	MXCP5						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x1AB	MX5EV7	MX5EV6	MX5EV5	MX5EV4	MX5EV3	MX5EV2	MX5EV1	MX5EV0

address	bit 31 … bit 0
0x1AC	Multiplexed Counter 5 Prescaler

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x1B0	MXC6	MXCP6						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x1B3	MX6EV7	MX6EV6	MX6EV5	MX6EV4	MX6EV3	MX6EV2	MX6EV1	MX6EV0

address	bit 31 … bit 0
0x1B4	Multiplexed Counter 6 Prescaler

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x1B8	MXC7	MXCP7						

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x1BB	MX7EV7	MX7EV6	MX7EV5	MX7EV4	MX7EV3	MX7EV2	MX7EV1	MX7EV0

address	bit 31 … bit 0
0x1BC	Multiplexed Counter 7 Prescaler

Bit	**Function**
MXCx	Multiplexed counter output status (read-only)
MXPx	Multiplexed counter output polarity
MXxEV7	Map rising edge of multiplexed counter x to send out event trigger 7
MXxEV6	Map rising edge of multiplexed counter x to send out event trigger 6
MXxEV5	Map rising edge of multiplexed counter x to send out event trigger 5
MXxEV4	Map rising edge of multiplexed counter x to send out event trigger 4
MXxEV3	Map rising edge of multiplexed counter x to send out event trigger 3
MXxEV2	Map rising edge of multiplexed counter x to send out event trigger 2

Bit	Function
MXxEV1	Map rising edge of multiplexed counter x to send out event trigger 1
MXxEV0	Map rising edge of multiplexed counter x to send out event trigger 0

2.25.3. Transition Board Output Mapping Registers

address	bit 15	bit 0
0x480	Transition Board Output 0 Mapping ID (see Table 6 on page 21 for mapping IDs)	

address	bit 15	bit 0
0x482	Transition Board Output 1 Mapping ID	

. . .

address	bit 15	bit 0
0x41E	Transition Board Output 15 Mapping ID	

2.25.4. Front Panel Input Mapping Registers

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x500	FP0SQMK(3:0)				FP0SQEEN3:0			
								FP0IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x501	FP0DB7	FP0DB6	FP0DB5	FP0DB4	FP0DB3	FP0DB2	FP0DB1	FP0DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x502	FP0MXCR		FP0SEN1	FP0SEN0			FP0SEQ1	FP0SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x503	FP0EV7	FP0EV6	FP0EV5	FP0EV4	FP0EV3	FP0EV2	FP0EV1	FP0EV0

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x504	FP1SQMK(3:0)				FP1SQEEN3:0			
								FP1IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x505	FP1DB7	FP1DB6	FP1DB5	FP1DB4	FP1DB3	FP1DB2	FP1DB1	FP1DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x506	FP1MXCR		FP1SEN1	FP1SEN0			FP1SEQ1	FP1SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x507	FP1EV7	FP1EV6	FP1EV5	FP1EV4	FP1EV3	FP1EV2	FP1EV1	FP1EV0

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x508	FP2SQMK(3:0)				FP2SQEEN3:0			
								FP2IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x509	FP2DB7	FP2DB6	FP2DB5	FP2DB4	FP2DB3	FP2DB2	FP2DB1	FP2DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x50A	FP2MXCR		FP2SEN1	FP2SEN0			FP2SEQ1	FP2SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x50B	FP2EV7	FP2EV6	FP2EV5	FP2EV4	FP2EV3	FP2EV2	FP2EV1	FP2EV0

Bit	Function
FPxSQMKy	Map Front panel Input x to Sequence Event Mask bit y
FPxSQEENy	Map Front panel Input x to Sequence Event Enable bit y
FPxIRQ	Map Front panel Input x to External Interrupt
FPxDB7	Map Front panel Input x to Distributed Bus bit 7
FPxDB6	Map Front panel Input x to Distributed Bus bit 6
FPxDB5	Map Front panel Input x to Distributed Bus bit 5
FPxDB4	Map Front panel Input x to Distributed Bus bit 4
FPxDB3	Map Front panel Input x to Distributed Bus bit 3
FPxDB2	Map Front panel Input x to Distributed Bus bit 2
FPxDB1	Map Front panel Input x to Distributed Bus bit 1
FPxDB0	Map Front panel Input x to Distributed Bus bit 0
FPxMXCR	Map Front panel Input x to Reset Multiplexed Counters
FPxSEN1	Map Front panel Input x to Sequence External Enable 1
FPxSEN0	Map Front panel Input x to Sequence External Enable 0
FPxSEQ1	Map Front panel Input x to Sequence Trigger 1
FPxSEQ0	Map Front panel Input x to Sequence Trigger 0
FPxEV7	Map Front panel Input x to Event Trigger 7
FPxEV6	Map Front panel Input x to Event Trigger 6
FPxEV5	Map Front panel Input x to Event Trigger 5
FPxEV4	Map Front panel Input x to Event Trigger 4
FPxEV3	Map Front panel Input x to Event Trigger 3
FPxEV2	Map Front panel Input x to Event Trigger 2
FPxEV1	Map Front panel Input x to Event Trigger 1
FPxEV0	Map Front panel Input x to Event Trigger 0

2.25.5. Front Panel Input Phase Monitoring Registers

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x520	PHCLR0	DBPH0					PHSEL0	

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x522					PHFE0			

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x523					PHRE0			

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x524	PHCLR1	DBPH1					PHSEL1	

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x526					PHFE1			

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x527					PHRE1			

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x528	PHCLR2	DBPH2					PHSEL2	

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x52A					PHFE2			

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x52B					PHRE2			

Bit	Function
PHCLRx	Reset phase monitoring registers by writing ‘1’
DBPHx	Distributed bus phase sampled on rising edge of input signal
PHSELx(1:0)	Input phase select 00 - Sample input with 0ï¿½ event clock 01 - Sample input with 90ï¿½ event clock 10 - Sample input with 180ï¿½ event clock 11 - Sample input with 270ï¿½ event clock
PHFEx(3:0)	Falling edge phase monitoring register
PHREx(3:0)	Rising edge phase monitoring register

2.25.6. Universal Input Mapping Registers

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x540	UI0SQMK(3:0)				UI0SQEEN3:0			
								UI0IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x541	UI0DB7	UI0DB6	UI0DB5	UI0DB4	UI0DB3	UI0DB2	UI0DB1	UI0DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x542	UI0MXCR		UI0SEN1	UI0SEN0			UI0SEQ1	UI0SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x543	UI0EV7	UI0EV6	UI0EV5	UI0EV4	UI0EV3	UI0EV2	UI0EV1	UI0EV0

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x544	UI1SQMK(3:0)				UI1SQEEN3:0			
								UI1IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x545	UI1DB7	UI1DB6	UI1DB5	UI1DB4	UI1DB3	UI1DB2	UI1DB1	UI1DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x546	UI1MXCR		UI1SEN1	UI1SEN0			UI1SEQ1	UI1SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x547	UI1EV7	UI1EV6	UI1EV5	UI1EV4	UI1EV3	UI1EV2	UI1EV1	UI1EV0

...

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x55C	UI15SQMK(3:0)				UI15SQEEN3:0			
								UI15IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x55D	UI15DB7	UI15DB6	UI15DB5	UI15DB4	UI15DB3	UI15DB2	UI15DB1	UI15DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x55E	UI15MXCR		UI15SEN1	UI15SEN0			UI15SEQ1	UI15SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x55F	UI15EV7	UI15EV6	UI15EV5	UI15EV4	UI15EV3	UI15EV2	UI15EV1	UI15EV0

Bit	Function
UIxSQMKy	Map Universal Input x to Sequence Event Mask bit y
UIxSQEENy	Map Universal Input x to Sequence Event Enable bit y
UIxIRQ	Map Universal Input x to External Interrupt
UIxDB7	Map Universal Input x to Distributed Bus bit 7
UIxDB6	Map Universal Input x to Distributed Bus bit 6
UIxDB5	Map Universal Input x to Distributed Bus bit 5
UIxDB4	Map Universal Input x to Distributed Bus bit 4
UIxDB3	Map Universal Input x to Distributed Bus bit 3
UIxDB2	Map Universal Input x to Distributed Bus bit 2
UIxDB1	Map Universal Input x to Distributed Bus bit 1
UIxDB0	Map Universal Input x to Distributed Bus bit 0
UIxMXCR	Map Universal Input x to Reset Multiplexed Counters
UIxSEN1	Map Universal Input x to Sequence External Enable 1
UIxSEN0	Map Universal Input x to Sequence External Enable 0
UIxSEQ1	Map Universal Input x to Sequence Trigger 1

Bit	**Function**
UIxSEQ0	Map Universal Input x to Sequence Trigger 0
UIxEV7	Map Universal Input x to Event Trigger 7
UIxEV6	Map Universal Input x to Event Trigger 6
UIxEV5	Map Universal Input x to Event Trigger 5
UIxEV4	Map Universal Input x to Event Trigger 4
UIxEV3	Map Universal Input x to Event Trigger 3
UIxEV2	Map Universal Input x to Event Trigger 2
UIxEV1	Map Universal Input x to Event Trigger 1
UIxEV0	Map Universal Input x to Event Trigger 0

Note: all enabled input signals are OR'ed together. So if e.g. distributed bus bit 0 has two sources from universal input 0 and 1, if either of the inputs is active high also the distributed bus is active high.

2.25.7. Transition Board Input Mapping Registers

address	**bit 31**	**bit 30**	**bit 29**	**bit 28**	**bit 27**	**bit 26**	**bit 25**	**bit 24**
0x600	TI0SQMK(3:0)				TI0SQEEN3:0			
								TI0IRQ

address	**bit 23**	**bit 22**	**bit 21**	**bit 20**	**bit 19**	**bit 18**	**bit 17**	**bit 16**
0x601	TI0DB7	TI0DB6	TI0DB5	TI0DB4	TI0DB3	TI0DB2	TI0DB1	TI0DB0

address	**bit 15**	**bit 14**	**bit 13**	**bit 12**	**bit 11**	**bit 10**	**bit 9**	**bit 8**
0x602	TI0MXCR		TI0SEN1	TI0SEN0			TI0SEQ1	TI0SEQ0

address	**bit 7**	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
0x603	TI0EV7	TI0EV6	TI0EV5	TI0EV4	TI0EV3	TI0EV2	TI0EV1	TI0EV0

address	**bit 31**	**bit 30**	**bit 29**	**bit 28**	**bit 27**	**bit 26**	**bit 25**	**bit 24**
0x604	TI1SQMK(3:0)				TI1SQEEN3:0			
								TI1IRQ

address	**bit 23**	**bit 22**	**bit 21**	**bit 20**	**bit 19**	**bit 18**	**bit 17**	**bit 16**
0x605	TI1DB7	TI1DB6	TI1DB5	TI1DB4	TI1DB3	TI1DB2	TI1DB1	TI1DB0

address	**bit 15**	**bit 14**	**bit 13**	**bit 12**	**bit 11**	**bit 10**	**bit 9**	**bit 8**
0x606	TI1MXCR		TI1SEN1	TI1SEN0			TI1SEQ1	TI1SEQ0

address	**bit 7**	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
0x607	TI1EV7	TI1EV6	TI1EV5	TI1EV4	TI1EV3	TI1EV2	TI1EV1	TI1EV0

...

address	**bit 31**	**bit 30**	**bit 29**	**bit 28**	**bit 27**	**bit 26**	**bit 25**	**bit 24**
0x61C	TI15SQMK(3:0)				TI15SQEEN3:0			
								TI15IRQ

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x61D	TI15DB7	TI15DB6	TI15DB5	TI15DB4	TI15DB3	TI15DB2	TI15DB1	TI15DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x61E	TI15MXCR		TI15SEN1	TI15SEN0			TI15SEQ1	TI15SEQ0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x61F	TI15EV7	TI15EV6	TI15EV5	TI15EV4	TI15EV3	TI15EV2	TI15EV1	TI15EV0

Bit	Function
TIxSQMKy	Map Transition Board Input x to Sequence Event Mask bit y
TIxSQEENy	Map Transition Board Input x to Sequence Event Enable bit y
TIxIRQ	Map Transition Board Input x to External Interrupt
TIxDB7	Map Transition Board Input x to Distributed Bus bit 7
TIxDB6	Map Transition Board Input x to Distributed Bus bit 6
TIxDB5	Map Transition Board Input x to Distributed Bus bit 5
TIxDB4	Map Transition Board Input x to Distributed Bus bit 4
TIxDB3	Map Transition Board Input x to Distributed Bus bit 3
TIxDB2	Map Transition Board Input x to Distributed Bus bit 2
TIxDB1	Map Transition Board Input x to Distributed Bus bit 1
TIxDB0	Map Transition Board Input x to Distributed Bus bit 0
FPxMXCR	Map Transition Board Input x to Reset Multiplexed Counters
TIxSEN1	Map Transition Board Input x to Sequence External Enable 1
TIxSEN0	Map Transition Board Input x to Sequence External Enable 0
TIxSEQ1	Map Transition Board Input x to Sequence Trigger 1
TIxSEQ0	Map Transition Board Input x to Sequence Trigger 0
TIxEV7	Map Transition Board Input x to Event Trigger 7
TIxEV6	Map Transition Board Input x to Event Trigger 6
TIxEV5	Map Transition Board Input x to Event Trigger 5
TIxEV4	Map Transition Board Input x to Event Trigger 4
TIxEV3	Map Transition Board Input x to Event Trigger 3
TIxEV2	Map Transition Board Input x to Event Trigger 2
TIxEV1	Map Transition Board Input x to Event Trigger 1
TIxEV0	Map Transition Board Input x to Event Trigger 0

Note: all enabled input signals are OR'ed together. So if e.g. distributed bus bit 0 has two sources from universal input 0 and 1, if either of the inputs is active high also the distributed bus is active high.

2.26. Embedded Event Receivers

The VME-EVM-300 has two embedded event receivers. The downstream event receiver (EVRD) receives the event stream from port U whereas the upstream event receiver (EVRU) receives the concentrated event stream from ports 1 through 8.

The downstream event receiver (EVRD) is located in the EVG register map at offset 0x20000 through 0x2ffff and the upstream event receiver (EVRU) is located in the EVG register map at offset 0x30000 through 0x3ffff. The event receiver register map follows the description further in this document.

The event master has a number of internal signals which are connected following:

Signal destination	Signal source
EVG UNIVIN(0)	EVRD FPOUT(0)
EVG UNIVIN(1)	EVRD FPOUT(1)
EVG UNIVIN(2)	EVRD FPOUT(2)
EVG UNIVIN(3)	EVRD FPOUT(3)
EVG UNIVIN(4)	EVRD FPOUT(4)
EVG UNIVIN(5)	EVRD FPOUT(5)
EVG UNIVIN(6)	EVRD FPOUT(6)
EVG UNIVIN(7)	EVRD FPOUT(7)
EVG UNIVIN(8)	EVRU FPOUT(0)
EVG UNIVIN(9)	EVRU FPOUT(1)
EVG UNIVIN(10)	EVRU FPOUT(2)
EVG UNIVIN(11)	EVRU FPOUT(3)
EVG UNIVIN(12)	EVRU FPOUT(4)
EVG UNIVIN(13)	EVRU FPOUT(5)
EVG UNIVIN(14)	EVRU FPOUT(6)
EVG UNIVIN(15)	EVRU FPOUT(7)
EVRD FPIN(0)	EVRU FPOUT(0)
EVRD FPIN(1)	EVRU FPOUT(1)
EVRD FPIN(2)	EVRU FPOUT(2)
EVRD FPIN(3)	EVRU FPOUT(3)
EVRD FPIN(4)	EVRU FPOUT(4)
EVRD FPIN(5)	EVRU FPOUT(5)
EVRD FPIN(6)	EVRU FPOUT(6)
EVRD FPIN(7)	EVRU FPOUT(7)
EVRU FPIN(0)	EVRD FPOUT(0)
EVRU FPIN(1)	EVRD FPOUT(1)
EVRU FPIN(2)	EVRD FPOUT(2)
EVRU FPIN(3)	EVRD FPOUT(3)
EVRU FPIN(4)	EVRD FPOUT(4)
EVRU FPIN(5)	EVRD FPOUT(5)
EVRU FPIN(6)	EVRD FPOUT(6)
EVRU FPIN(7)	EVRD FPOUT(7)

Signal destination	Signal source

2.27. FCT Function Register Map

Address	Register	Type	Description
0x000	Status	UINT32	Status Register
0x004	Control	UINT32	Control Register
0x010	UpDCValue	UINT32	Upstream Data Compensation Delay Value
0x014	FIFODCValue	UINT32	Receive FIFO Data Compensation Delay Value
0x018	IntDCValue	UINT32	FCT Internal Datapath Data Compensation Delay Value
0x02C	TopologyID	UINT32	Timing Node Topology ID
0x034	RX Shutter	UINT32	RX Channel Specific Event and DBUS Shutter enable
0x034	TX Shutter	UINT32	TX Channel Specific Event and DBUS Shutter enable
0x040	Port1DCValue	UINT32	Port 1 loop delay value
0x044	Port2DCValue	UINT32	Port 2 loop delay value
0x048	Port3DCValue	UINT32	Port 3 loop delay value
0x04C	Port4DCValue	UINT32	Port 4 loop delay value
0x050	Port5DCValue	UINT32	Port 5 loop delay value
0x054	Port6DCValue	UINT32	Port 6 loop delay value
0x058	Port7DCValue	UINT32	Port 7 loop delay value
0x05C	Port8DCValue	UINT32	Port 8 loop delay value
0x1000 – 0x10FF	SFP1EEPROM		Port 1 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1100 – 0x11FF	SFP1DIAG		Port 1 SFP Transceiver diagnostics (SFP address 0xA2)
0x1200 – 0x12FF	SFP2EEPROM		Port 2 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1300 – 0x13FF	SFP2DIAG		Port 2 SFP Transceiver diagnostics (SFP address 0xA2)
0x1400 – 0x14FF	SFP3EEPROM		Port 3 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1500 – 0x15FF	SFP3DIAG		Port 3 SFP Transceiver diagnostics (SFP address 0xA2)
0x1600 – 0x16FF	SFP4EEPROM		Port 4 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1700 – 0x17FF	SFP4DIAG		Port 4 SFP Transceiver diagnostics (SFP address 0xA2)
0x1800 – 0x18FF	SFP5EEPROM		Port 5 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1900 – 0x19FF	SFP5DIAG		Port 5 SFP Transceiver diagnostics (SFP address 0xA2)
0x1A00 – 0x1AFF	SFP6EEPROM		Port 6 SFP Transceiver EEPROM contents (SFP address 0xA0)

Address	Register	Type	Description
0x1B00 – 0x1BFF	SFP6DIAG		Port 6 SFP Transceiver diagnostics (SFP address 0xA2)
0x1C00 – 0x1CFF	SFP7EEPROM		Port 7 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1D00 – 0x1DFF	SFP7DIAG		Port 7 SFP Transceiver diagnostics (SFP address 0xA2)
0x1E00 – 0x1EFF	SFP8EEPROM		Port 8 SFP Transceiver EEPROM contents (SFP address 0xA0)
0x1F00 – 0x1FFF	SFP8DIAG		Port 8 SFP Transceiver diagnostics (SFP address 0xA2)

2.27.1. Status Register

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x001	LINK8	LINK7	LINK6	LINK5	LINK4	LINK3	LINK2	LINK1

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x003	VIO8	VIO7	VIO6	VIO5	VIO4	VIO3	VIO2	VIO1

Bit	Function
LINK8	Port 8 RX Status, 1 – link up, 0 – link down
LINK7	Port 7 RX Status, 1 – link up, 0 – link down
LINK6	Port 6 RX Status, 1 – link up, 0 – link down
LINK5	Port 5 RX Status, 1 – link up, 0 – link down
LINK4	Port 4 RX Status, 1 – link up, 0 – link down
LINK3	Port 3 RX Status, 1 – link up, 0 – link down
LINK2	Port 2 RX Status, 1 – link up, 0 – link down
LINK1	Port 1 RX Status, 1 – link up, 0 – link down
VIO8	Port 8 RX Violation
VIO7	Port 7 RX Violation
VIO6	Port 6 RX Violation
VIO5	Port 5 RX Violation
VIO4	Port 4 RX Violation
VIO3	Port 3 RX Violation
VIO2	Port 2 RX Violation
VIO1	Port 1 RX Violation

2.27.2. Control Register

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x007	CLRV8	CLRV7	CLRV6	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1

Bit	Function
CLRV8	Clear RX Violation Port 8
CLRV7	Clear RX Violation Port 7
CLRV6	Clear RX Violation Port 6
CLRV5	Clear RX Violation Port 5
CLRV4	Clear RX Violation Port 4
CLRV3	Clear RX Violation Port 3
CLRV2	Clear RX Violation Port 2
CLRV1	Clear RX Violation Port 1

2.27.3. RX Shutter Register

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x035	RXDBSH8	RXDBSH7	RXDBSH6	RXDBSH5	RXDBSH4	RXDBSH3	RXDBSH2	RXDBSH1
address	**bit 7**	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
0x037	RXEVSH8	RXEVSH7	RXEVSH6	RXEVSH5	RXEVSH4	RXEVSH3	RXEVSH2	RXEVSH1

Bit	Function
RXDBSH8	When ’1’ disable RX DBUS bits from port 8
RXDBSH7	When ’1’ disable RX DBUS bits from port 7
RXDBSH6	When ’1’ disable RX DBUS bits from port 6
RXDBSH5	When ’1’ disable RX DBUS bits from port 5
RXDBSH4	When ’1’ disable RX DBUS bits from port 4
RXDBSH3	When ’1’ disable RX DBUS bits from port 3
RXDBSH2	When ’1’ disable RX DBUS bits from port 2
RXDBSH1	When ’1’ disable RX DBUS bits from port 1
RXEVSH8	When ’1’ disable RX Events from port 8
RXEVSH7	When ’1’ disable RX Events from port 7
RXEVSH6	When ’1’ disable RX Events from port 6
RXEVSH5	When ’1’ disable RX Events from port 5
RXEVSH4	When ’1’ disable RX Events from port 4
RXEVSH3	When ’1’ disable RX Events from port 3
RXEVSH2	When ’1’ disable RX Events from port 2
RXEVSH1	When ’1’ disable RX Events from port 1

2.27.4. TX Shutter Register

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x039	TXDBSH8	TXDBSH7	TXDBSH6	TXDBSH5	TXDBSH4	TXDBSH3	TXDBSH2	TXDBSH1

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x03B	TXEVSH8	TXEVSH7	TXEVSH6	TXEVSH5	TXEVSH4	TXEVSH3	TXEVSH2	TXEVSH1

Bit	Function
TXDBSH8	When ’1’ disable TX DBUS bits to port 8
TXDBSH7	When ’1’ disable TX DBUS bits to port 7
TXDBSH6	When ’1’ disable TX DBUS bits to port 6
TXDBSH5	When ’1’ disable TX DBUS bits to port 5
TXDBSH4	When ’1’ disable TX DBUS bits to port 4
TXDBSH3	When ’1’ disable TX DBUS bits to port 3
TXDBSH2	When ’1’ disable TX DBUS bits to port 2
TXDBSH1	When ’1’ disable TX DBUS bits to port 1
TXEVSH8	When ’1’ disable TX Events to port 8
TXEVSH7	When ’1’ disable TX Events to port 7
TXEVSH6	When ’1’ disable TX Events to port 6
TXEVSH5	When ’1’ disable TX Events to port 5
TXEVSH4	When ’1’ disable TX Events to port 4
TXEVSH3	When ’1’ disable TX Events to port 3
TXEVSH2	When ’1’ disable TX Events to port 2
TXEVSH1	When ’1’ disable TX Events to port 1

2.28. Firmware Version Change Log

FW Version	Date	Changes	Affected HW
0200	11.06.2015	- Prototype release	VME-EVM-300
0201	24.09.2015	- Added segmented data buffer - Fixed Port 1 TX polarity	VME-EVM-300
010202	01.10.2015	- Changed receive FIFO delay target to 00060000 - Added LED test mode (production testing) - Removed test signals from TBOUT	VME-EVM-300
010203	23.11.2015	- Added changes for running with a slower clock on fan-out.	VME-EVM-300
020203	18.12.2015	- Changes to data buffer forwarding - Changes for rate conversion forwarding, using internal div/2.	VME-EVM-300
0204	12.01.2016	- /2 rate conversion working on events, dbuf and dbits. - Improvements to delay measurement system.	VME-EVM-300
0205	13.04.2016	- Moved delay compensation segment from segment 0 to last segment in memory. - Fixed front panel TTL input order. - Fixed race condition in segmented memory buffer transmission that caused dropped software buffers.	VME-EVM-300
FB0206	23.12.2016	- Added upstream and downstream event receivers. - Changed beacon event from 0x7A to 0x7E. - Added topology ID - Added delay measurement validity information to delay compensation data	VME-EVM-300
000207	19.01.2017	- Added front panel input phase monitoring and phase select features. - Added external AC input synhronisation features.	VME-EVM-300
010207	09.02.2017	- Fixed occasional dropped out downstream and upstream data buffers/segmented data buffers.	VME-EVM-300
030207	03.05.2017	- Added RF input monitoring logic to automatically recover from lost RF signal. - Added a way to toggle distributed bus transmission phase when an external AC synchronisation clock is used.	VME-EVM-300
040207	23.05.2017	- Fixed readout of diagnostics information on single mode transceivers.	VME-EVM-300
050207	26.06.2017	- Fixed transceiver_channel to turn off receiver on first error to prevent propagation of errors up stream.	VME-EVM-300
070207	04.05.2020	- Changed VME CSR registers to readback unused bits as zero. - Fixed bus error on write to fct registers - Added external reset for MXC	VME-EVM-300

2.28.1. mTCA.4 development branch

FW Version	Date	Changes	Affected HW
070207	31.01.2019	- Moved MIRQ to 0x1C register. Moved DCDataBufControl from 0x1c to 0x30.	mTCA-EVM-300
0A0207	15.11.2019	- Fixed RX polarity of mTCA-EVM-300 ports 6 and 7. Defaults for fan-out. - Added support for mTCA backplane I/O on trigger bus.	mTCA-EVM-300
0B0207	15.11.2019	- Fixed EVRU/D to EVG signal and added pullup to MOD_SDA.	mTCA-EVM-300
0C0207	02.04.2020	- Added settings files. - Changed event clock contraint to 6 ns.	mTCA-EVM-300

2.28.2. Merging of VME and mTCA.4 development branches

FW Version	Date	Changes	Affected HW
0C0207	11.09.2020	- Merge VME tested successfully.	VME-EVM-300
0E0207	24.02.2021	- Merge mTCA.4 tested successfully.	mTCA-EVM-300
120207	21.05.2024	- Added pulse sequence repetition functionality.	mTCA-EVM-300
130207	26.11.2025	- Added port specific RX and TX Shutters for both events and DBUS bits.	mTCA-EVM-300

3. Event Receiver

Event Receivers decode timing events and signals from an optical event stream transmitted by an Event Generator. Events and signals are received at predefined rate the event clock that is usually divided down from an accelerators main RF reference. The event receivers lock to the phase event clock of the Event Generator and are thus phase locked to the RF reference. Event Receivers convert event codes transmitted by an Event Generator to hardware outputs. They can also generate software interrupts and store the event codes with globally distributed timestamps into FIFO memory to be read by a CPU.

3.1. Functional Description

After recovering the event clock the Event Receiver demultiplexes the event stream to 8-bit distributed bus data and 8-bit event codes. The distributed bus may be configured to share its bandwidth with time deterministic data transmission.

3.1.1. Event Decoding

The Event Receiver provides two mapping RAMs of 256 × 128 bits. Only one of the RAMs can be active at a time, however both RAMs may be modified at any time. The event code is applied to the address lines of the active mapping RAM. The 128-bit data programmed into a specific memory location pointed to by the event code determines what actions will be taken.

Event code	Offset	Internal functions	Pulse Triggers	'Set' Pulse	'Reset' Pulse
0x00	0x0000	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits
0x01	0x0010	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits
0x02	0x0020	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits
...	...	...	...	...	...
0xFF	0x0FF0	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits	4 bytes/32 bits

There are 32 bits reserved for internal functions which are by default mapped to the event codes shown in table . The remaining 96 bits control internal pulse generators. For each pulse generator there is one bit to trigger the pulse generator, one bit to set the pulse generator output and one bit to clear the pulse generator output.

Map bit	Default event code	Function
127	n/a	Save event in FIFO
126	n/a	Latch timestamp
125	n/a	Led event
124	n/a	Forward event from RX to TX
123	0x79	Stop event log
122	n/a	Log event
102 to 121	n/a	(Reserved)
101	0x7a	Hearbeat event

Map bit	Default event code	Function
100	0x7b	Reset Prescalers
99	0x7d	Timestamp reset event (TS counter reset)
98	0x7c	Timestamp clock event (TS counter increment)
97	0x71	Seconds shift register ‘1’
96	0x70	Seconds shift register ‘0’
80 to 95		(Reserved)
79		Trigger pulse generator 15
...		...
64		Trigger pulse generator 0
48 to 63		(Reserved)
47		Set pulse generator 15 output high
...		...
32		Set pulse generator 0 output high
16 to 31		(Reserved)
15		Reset pulse generator 15 output low
...		...
0		Reset pulse generator 0 output low

3.1.2. Event Counting

Starting with subrelease ID <TODO> the capability of counting events has been added. Each of the 255 non-null events has a 32-bit counter which increments each time the event is received (or generated) in the EVR. Counter values are available as 32-bit read-only registers (*EventCntx*).

The counters are set to 0 at device start-up, and after that, they keep incrementing with each event. There is no functionality to reset the counters, so it is up to the application to note the initial counter values. Eventually the 32-bit counters will overflow. The overflow will not be indicated by the EVR - in order to detect it, the application must keep track of the value regularly enough, depending on the expected event frequency.

3.1.3. Heartbeat Monitor

A heartbeat monitor is provided to receive heartbeat events. Event code $7A is by default set up to reset the heartbeat counter. If no heartbeat event is received the counter times out (approx. 1.6 s) and a heartbeat flag is set. The Event Receiver may be programmed to generate a heartbeat interrupt.

3.1.4. Event FIFO and Timestamp Events

The Event System provides a global timebase to attach timestamps to collected data and performed actions. The time stamping system consists of a 32-bit timestamp event counter and a 32-bit seconds counter. The timestamp event counter either counts received timestamp counter clock events or runs freely with a clock derived from the event clock. The event counter is also able to run on a clock provided on a distributed bus bit.

The event counter clock source is determined by the prescaler control register. The timestamp event counter is cleared at the next event counter rising clock edge after receiving a timestamp event counter reset event.

The seconds counter is updated serially by loading zeros and ones (see mapping register bits) into a shift register MSB first. The seconds register is updated from the shift register at the same time the timestamp event counter is reset.

The timestamp event counter and seconds counter contents may be latched into a timestamp latch. Latching is determined by the active event map RAM and may be enabled for any event code.

An event FIFO memory is implemented to store selected event codes with attached timing information. The 80-bit wide FIFO can hold up to 511 events. The recorded event is stored along with 32-bit seconds counter contents and 32-bit timestamp event counter contents at the time of reception. The event FIFO as well as the timestamp counter and latch are accessible by software.

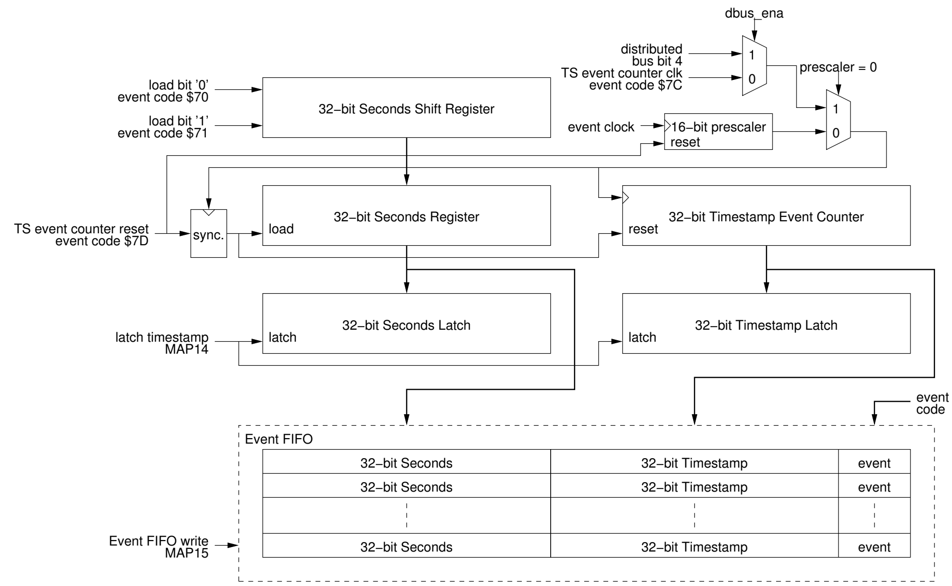

Figure 18: Event FIFO and Timestamping

3.1.5. Event Log

Up to 512 events with timestamping information can be stored in the event log. The log is implemented as a ring buffer and is accessible as a memory region. Logging events can be stopped by an event or software.

3.1.6. Distributed Bus and Data Transmission

The distributed bus is able to carry eight simultaneous signals sampled with half the event clock rate over the fibre optic transmission media. The distributed bus signals may be output on programmable front panel outputs.

The distributed bus bandwidth is shared by transmission of a configurable size data buffer to up to 2 kbytes.

3.1.7. Pulse Generators

The structure of the pulse generation logic is shown in Figure 19. Three signals from the mapping RAM control the output of the pulse: trigger, ‘set’ pulse and ‘reset’ pulse. A trigger causes the delay counter to start counting, when the end-of-count is reached the output pulse changes to the ‘set’ state and the width counter starts counting. At the end of the width count the output pulse is cleared. The mapping RAM signal ‘set’ and ‘reset’ cause the output to change state immediately without any delay.

Starting from firmware version 0200 pulse generators can also be triggered from rising edges of distributed bus signals or EVR internal prescalers.

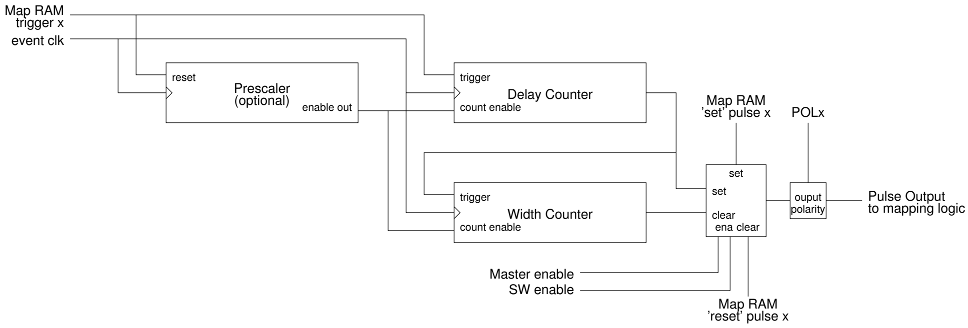

Figure 19: Pulse Generator

32 bit registers are reserved for both counters and the prescaler, however, the prescaler is not necessarily implemented for all channels and may be hard coded to 1 in case the prescaler is omitted. Software may write 0xFFFFFFFF to these registers and read out the actual width or hard-coded value of the register. For example if the width counter is limited to 16 bits a read will return 0x0000FFFF after a write of 0xFFFFFFFF.

3.1.8. Pulse Generator Gates

Depending on firmware revision/form factor a number of pulse generators are configured as event triggered gates only and can be used to mask or enable pulse generator triggers. The VME-EVR-300, PCIe-EVR-300DC and mTCA-EVR-300 have four pulse generators configured as gates, pulse generators 28 to 31 which correspond gates 0 to 3.

3.1.9. Pulse Generator Pulse Trains

Starting with subrelease ID 0x19 a capability of generating pulse trains has been added to the pulse generators. Pulse trains are enabled by specifying the number of pulse repetitions and the repeat interval. The pulse width is defined by the width of the initial pulse. When the number of repetitions is 0, the pulse generator behaves exactly the same way as in earlier firmware releases i.e. the total number of pulses is n+1 where n is the number of repetitions. Both the repeat delay (interval) and number of repetitions registers are 32 bit wide.

The pulse repeat delay must be three event clock cycles longer than the pulse width.

3.1.10. Pulse Counting

Starting with subrelease ID <TODO> the capability of counting pulses has been added. Each pulse generator has a dedicated 32-bit leading edge counter, which is available as a read-only register (*PulseCntx*). The polarity of a pulse generator is taken into account - rising edges will be counted if polarity is set to normal (PxPOL=0), and falling edges if it is set to inverted (PxPOL=1).

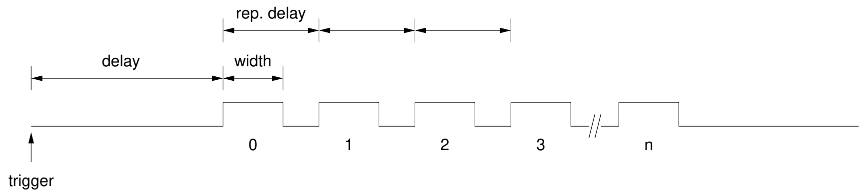

Figure 20: Pulse Generator Pulse Trains

The counters are set to 0 at device start-up, and after that, they keep incrementing with each pulse. There is no functionality to reset the counters, so it is up to the application to note the initial counter values. Eventually the 32-bit counters will overflow. The overflow will not be indicated by the EVR - in order to detect it, the application must keep track of the value regularly enough, depending on the expected pulse frequency.

3.1.11. Prescalers

The Event Receiver provides a number of programmable prescalers. The frequencies are programmable and are derived from the event clock. A special event code reset prescalers $7B causes the prescalers to be synchronously reset, so the frequency outputs will be in same phase across all event receivers.

3.1.12. Programmable Front Panel, Universal I/O and Backplane Connections

All outputs are programmable: each pulse generator output, prescaler and distributed bus bit can be mapped to any output. Starting with firmware version 0200 each output can have two sources which are logically OR'ed together. The mapping for a single source is shown in table below.

Each output has a two byte mapping register and each byte corresponds to a single source. An unused mapping source should be set to 63 (0x3f). In case of a bidirectional signal to tri-state set both bytes to 61 (0x3d).

Table 20: Output mapping values

Mapping ID	Signal
0 to n-1	Pulse generator output (number n of pulse generators depends on HW and firmware version)
n to 31	(Reserved)
32	Distributed bus bit 0 (DBUS0)
...	...
39	Distributed bus bit 7 (DBUS7)
40	Prescaler 0
41	Prescaler 1
42	Prescaler 2
43	Prescaler 3
44	Prescaler 4

Mapping ID	Signal
45	Prescaler 5
46	Prescaler 6
47	Prescaler 7
48	Flip-flop output 0
...	...
55	Flip-flop output 7
56 to 58	(Reserved)
59	Event clock output (only on PXIe-EVR-300)
60	Event clock output with 180ï¿½ phase shift (only on PXIe-EVR-300)
61	Tri-state output (for PCIe-EVR-300DC with input module populated in IFB-300's Universal I/O slot)
62	Force output high (logic 1)
63	Force output low (logic 0)

3.1.13. Flip-flop Outputs (from FW version 0E0207)

There are 8 flip-flop outputs. Each of these is using two pulse generators, one for setting the output high and the other one for resetting the output low. In the table below you can see the relationship between flip-flops and pulse generators and the output mapping IDs.

flip-flop	mapping ID	Set	Reset
0	48	Pulse gen. 0	Pulse gen. 1
1	49	Pulse gen. 2	Pulse gen. 3
2	50	Pulse gen. 4	Pulse gen. 5
3	51	Pulse gen. 6	Pulse gen. 7
4	52	Pulse gen. 8	Pulse gen. 9
5	53	Pulse gen. 10	Pulse gen. 11
6	54	Pulse gen. 12	Pulse gen. 13
7	55	Pulse gen. 14	Pulse gen. 15

3.1.14. Front Panel Universal I/O Slots

Universal I/O slots provide different types of output with exchangeable Universal I/O modules. Each module provides two outputs e.g. two TTL output, two NIM output or two optical outputs. The source for these outputs is selected with mapping registers.

3.1.15. VME-EVR-300 GTX Front Panel Outputs and mTCA-EVR TCLKA/TCLKB Clocks

The VME-EVR-300 has four GTX front panel outputs, two in Universal I/O slot UNIV6/UNIV7 and CML outputs CML0 and CML1. The GTX Outputs provide low jitter signals with special outputs. The outputs can work in different configurations: pulse mode, pattern mode and frequency mode. The difference compared to the CML output of the VME-EVR-230RF is that instead of 20 bits per event clock cycle the GTX outputs have 40 bits per event clock cycle doubling the resolution to 200 ps/bit at an event clock of 125 MHz.

The mTCA-EVR-300 TCLKA and TCLKB backplane clock operate the same way as VME-EVR-300 GTX front panel outputs. The pulse mapping is controlled through UNIV16 (TCLKA) and UNIV17 (TCLKB) mapping registers.

3.1.16. mTCA-EVR-300RF GTX Outputs

The mTCA-EVR-300RF has altogether six GTX outputs, two on the backplane, TCLKA and TCLKB, and four on the front panel: one Universal I/O slot (UNIV2/UNIV3), one SFP output, and one differential CML output.

The SFP output can be configured to generate a modulated signal that can be received by an Electron Gun trigger receiver GUN-RC-300. For GUN-RC-300 operation the GTX output has to be placed in GTX3MD mode (see GTX output control register).

The GUN-TX mode has a special inhibit function that by default is using Universal Input 0 as an inhibit signal. When the state of the inhibit signal is high the GUN-TX output is disabled. The inhibit function may be overridden with the GTXIO override bit in the main control register.

The mTCA-EVR-300RF also adds the capability of phase shifting all of the GTX outputs individually with a step resolution of 1/2560 event clock cycles. The phase shift is controlled by the GTX Phase Shift Offset Register.

3.1.17. GTX Pulse Mode

The source for these outputs is selected in a similar way than the standard outputs using mapping registers, however, the output logic monitors the state of this signal and distinguishes between state low (00), rising edge (01), high state (11) and falling edge (10). Based on the state a 40 bit pattern is sent out with a bit rate of 40 times the event clock rate.

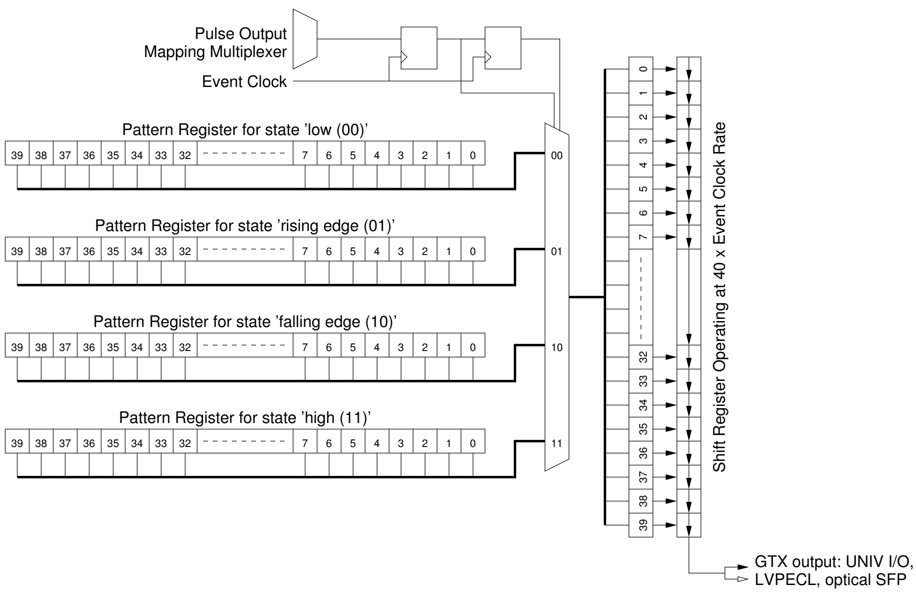

Figure 21: GTX Output

- When the source for a GTX output is low and was low one event clock cycle earlier (state low), the GTX output repeats the 40 bit pattern stored in pattern_00 register.
- When the source for a GTX output is high and was low one event clock cycle earlier (state rising), the GTX output sends out the 40 bit pattern stored in pattern_01 register.
- When the source for a GTX output is high and was high one event clock cycle earlier (state high), the GTX output repeats the 40 bit pattern stored in pattern_11 register.
- When the source for a GTX output is low and was high one event clock cycle earlier (state falling), the GTX output sends out the 40 bit pattern stored in pattern_10 register.

For an event clock of 125 MHz the duration of one single GTX output bit is 200 ps. These outputs allow for producing fine grained adjustable output pulses and clock frequencies.

3.1.18. GTX Frequency Mode

In frequency mode one can generate clocks where the clock period can be defined in steps of 1/40th part of the event clock cycle i.e. 200 ps step with an event clock of 125 MHz. There are some limitations, however:

- Clock high time and clock low time must be $\geq$ 40/40th event clock period steps
- Clock high time and clock low time must be < 65536/40th event clock period steps

The clock output can be synchronized by one of the pulse generators, distributed bus signal etc. When a rising edge of the mapped output signal is detected the frequency generator takes its output value from the trigger level bit and the counter value from the trigger position register. Thus one can adjust the phase of the synchronized clock in 1/40th steps of the event clock period.

To change the generated clock phase in respect to the trigger we can select the trigger polarity by bit CMLTL in the CML Control register and the trigger position also in the CML Control register.

3.1.19. GTX Pattern Mode

In pattern mode one can generate arbitrary bit patterns taking into account following:

- The pattern length is a multiple of 40 bits, where each bit is 1/40th of the event clock period
- Maximum length of the arbitrary pattern is 40 $\times$ 2048 bits
- A pattern can be triggered from any pulse generator, distributed bus bit etc. When triggered the pattern generator starts sending 40 bit words from the pattern memory sequentially starting from position 0. This goes on until the pattern length set by the samples register has been reached.
- If the pattern generator is in recycle mode the pattern continues immediately from position 0 of the pattern memory.
- If the pattern generator is in single pattern mode, the pattern stops and the 40 bit word from the last position of the pattern memory (2047) is sent out until the pattern generator is triggered again.

3.1.20. GTX GUN-RC-300 Mode (mTCA-EVR-300RF)

The front panel GTX SFP output can be configured to generate a modulated signal that can be received by an Electron Gun trigger receiver GUN-RC-300. The GUN-RC-300 is capable of generating pulse trains with a quarter resolution of the event clock cycle (2 ns @ 125 MHz) that is it allows triggering the gun bunch by bunch with a RF divider of four.

3.1.21. GUN-RC-300 Pulse Mode

This mode is similar to the GTX Pulse Mode except that due to the modulation used in the GUN-RC-300 the resolution is lower than in GTX Pulse Mode.

Here the 40 bit pattern is still used, however, it is required that in the 40 bit pattern five consecutive bits are of the same state that is bits 0-4 have the same state, bits 5-9 have the same state etc.

- When the source for the GTX GUN-TX output is low and was low one event clock cycle earlier (state low), the GTX output repeats the 8x5 bit pattern stored in pattern_00 register.
- When the source for the GTX GUN-TX output is high and was low one event clock cycle earlier (state rising), the GTX output sends out the 8x5 bit pattern stored in pattern_01 register.
- When the source for the GTX GUN-TX output is high and was high one event clock cycle earlier (state high), the GTX output repeats the 8x5 bit pattern stored in pattern_11 register.
- When the source for the GTX GUN-TX output is low and was high one event clock cycle earlier (state falling), the GTX output sends out the 8x5 bit pattern stored in pattern_10 register.

By default (after power up) the four first pattern registers have the following values:

pattern nr.	bit pattern
00	`00000 00000 00000 00000 00000 00000 00000 00000`
01	`11111 11111 11111 11111 11111 11111 11111 11111`
10	`00000 00000 00000 00000 00000 00000 00000 00000`
11	`11111 11111 11111 11111 11111 11111 11111 11111`

If for example the pattern for the rising edge (pattern_01) is changed to `11111 00000 11111 00000 11111 00000 11111 00000` then the pulse start is changed to four short pulses with a stable top until the end of the pulse.

Please see 4.3 for example for setting up the mTCA-EVR-300RF in GUN-RC-300 pulse mode.

3.1.22. GUN-RC-300 Pattern Mode

The pattern mode is similar to the GTX pattern mode, but like in the pulse mode the patterns is arranged as 8x5 bit patterns instead of fourty bit patterns.

- The pattern length is a multiple of 8x5 bits, where each 5 bit block is 1/8th of the event clock period
- Maximum length of the arbitrary pattern is 2048 event clock cycles
- A pattern can be triggered from the rising edge of any pulse generator, distributed bus bit etc. When triggered the pattern generator starts sending 8x5 bit words from the pattern memory sequentially starting from position 0. This goes on until the pattern length set by the samples register has been reached.

- If the pattern generator is in recycle mode the pattern continues immediately from position 0 of the pattern memory.
- If the pattern generator is in single pattern mode, the pattern stops and the 40 bit word from the last position of the pattern memory (2047) is sent out until the pattern generator is triggered again.

Common GUN-TX Mode Considerations. Common GUN-TX Mode Considerations

- The GTX outputs should not be enable before the EVG/EVR link is up
- Disconnecting any fibre connection between EVG/EVR or EVR/GUN-RC can lead to a spurious pulses at the GUN-RC
- A lost EVG RF reference can cause spurious pulses at the GUN-RC

3.1.23. Configurable Size Data Buffer

Pre-DC (Delay Compensation) event systems provided a way to to transmit configurable size data packets that may be transmitted over the event system link. The buffer transmission size is configured in the Event Generator to up to 2 kbytes. The Event Receiver is able to receive buffers of any size from 4 bytes to 2 kbytes in four byte (long word) increments.

3.1.24. Segmented Data Buffer

With the addition of delay compensation a segmented data buffer has been introduced and it can coexist with the configurable size data buffer. The segmented data buffer is divided into 16 byte segments that allow updating only part of the buffer memory with the remaining segments left untouched.

When starting a data transmission the Event Generator first sends the starting segment number that defines the starting address in the buffer. The data buffer address offset is the segment number * 16 bytes. The Event Receiver writes the received bytes into the data buffer and when transmission is complete a receive complete flag is raised for the starting segment of the packet transmission. The transmission can overlap several segments, however, the flag is raised only for the starting segment. If there is a checksum mismatch the checksum error flag for the starting segment is set. In case the receive complete flag already was set before the new data was received a segment overflow flag is set. Flags are cleared by writing a '1' to the receive flag. Each segment has a receive data counter and after completion of the transfer the receive data counter of the starting segment is updated with the actual number of bytes received in the transmission.

The procedure to receive a segmented data buffer is following:

- check that receive complete flag for received segment is set
- check that starting segment overflow flag is cleared
- read transmission size from segment receive data counter
- copy segment data from segmented data buffer memory into system RAM
- verify that starting segment overflow flag is still cleared
- clear segment receive complete flag

Starting with firmware 0205 the delay compensation logic uses the last 16 byte segment of the segmented data buffer for delay compensation data.

3.1.25. Interrupt Generation

The Event Receiver has multiple interrupt sources which all have their own enable and flag bits. The following events may be programmed to generate an interrupt:

- Receiver link state change
- Receiver violation: bit error or the loss of signal.
- Lost heartbeat: heartbeat monitor timeout.
- Write operation of an event to the event FIFO.
- Event FIFO is full.
- Data Buffer reception complete.

In addition to the events listed above an interrupt can be generated from one of the pulse generator outputs, distributed bus bits or prescalers. The pulse interrupt can be mapped in a similar way as the front panel outputs.

3.1.26. External Event Input

An external hardware input is provided to be able to take an external pulse to generate an internal event. This event will be handled as any other received event.

3.2. Programmable Reference Clock

The event receiver requires a reference clock to be able to synchronise on the incoming event stream sent by the event generator. For flexibility a programmable reference clock is provided to allow the use of the equipment in various applications with varying frequency requirements.

Please note before programming a new operating frequency with the fractional synthesizer the operating frequency (in MHz) has to be set in the UsecDivider register. This is essential as the event receiver's PLL cannot lock if it does not know the frequency range to lock to.

3.2.1. Fractional Synthesiser

The clock reference for the event receiver is generated on-board the event receiver using a fractional synthesiser. A Micrel (http://www.micrel.com) SY87739L Protocol Transparent Fractional-N Synthesiser with a reference clock of 24 MHz is used. The following table lists programming bit patterns for a few frequencies.

Event Rate	Configuration Bit Pattern	Reference Output	Precision (theoretical)
499.8 MHz/5 = 99.96 MHz	0x025B41ED	99.956 MHz	-40 ppm
50 MHz	0x009743AD	50.0 MHz	0
499.8 MHz/10 = 49.98 MHz	0x025B43AD	49.978 MHz	-40 ppm

The event receiver reference clock is required to be in ±100 ppm range of the event generator event clock.

3.3. Hardware Configuration Summary

		VME-EVR-300	mTCA-EVR-300U	mTCA-EVR-300RF	PCIe-EVR-300DC
Pulse Generators		24	24	24	24
FP TTL inputs		2	2	2	0
FP TTL outputs		0	4	0	0
FP GTX outputs		4[1]	0	4[2]	0
UNIV I/O / slots		4	2	2	16 / 8[3]
GPIO pins / slots		16 / 4	8 / 2	8 / 2	0 / 0
TB Outputs		16	10	10	0
TB Inputs		16	10	10	0
Prescalers		8 x 32 bit	8 x 32 bit	8 x 32 bit	8 x 32 bit
Pulse Generator Prescaler, Delay, Pulse Width Range (bits)	0	16, 32, 32	16, 32, 32	16, 32, 32	16, 32, 32
	1	16, 32, 32	16, 32, 32	16, 32, 32	16, 32, 32
	2	16, 32, 32	16, 32, 32	16, 32, 32	16, 32, 32
	3	16, 32, 32	16, 32, 32	16, 32, 32	16, 32, 32
	4	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	5	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	6	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	7	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	8	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	9	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	10	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	11	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	12	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	13	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	14	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	15	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	16	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	17	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	18	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	19	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	20	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	21	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	22	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16
	23	0, 32, 32	0, 32, 32	0, 32, 16	0, 32, 16

[1] One Universal I/O slot (2 outputs), 2 x CML output
[2] One Universal I/O slot (2 outputs), 1 x SFP output, 1 x CML output
[3] Universal I/O is available on the external I/O box

3.4. VME-EVR-300 Front Panel Connections

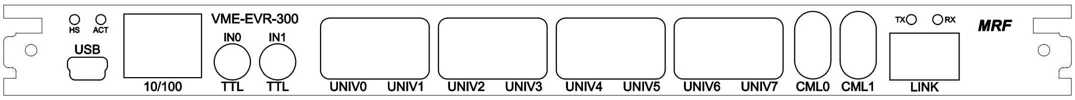

Figure 22: VME-EVR-300 Front Panel

The front panel of the Event Receiver includes the following connections and status leds:

Connector / Led	Style	Level	Description
HS	Red Led		Module Failure
HS	Blue Led		Module Powered Down
ACT	3-color Led		SAM3X Activity Led
USB	Micro-USB		SAM3X Serial port / JTAG interface
10/100	RJ45		SAM3X Ethernet Interface
IN0	LEMO	TTL (3.3V / 5V)	FPTTL0 Trigger input
IN1	LEMO	TTL (3.3V / 5V)	FPTTL1 Trigger input
UNIV0/1	Universal slot		Universal Output 0/1 with GPIO pins mapped to GPIO register bits 3–0
UNIV2/3	Universal slot		Universal Output 2/3 with GPIO pins mapped to GPIO register bits 7–4
UNIV4/5	Universal slot		Universal Output 4/5 with GPIO pins mapped to GPIO register bits 11–8
UNIV6/7	Universal slot		Universal Output 6/7 with GPIO pins mapped to GPIO register bits 15–12 The output signals come through CML/GTX logic block 0/1
CML0	LEMO EPY	CML	Mapped as Universal Output 8 The output signals come through CML/GTX logic block 2
CML1	LEMO EPY	CML	Mapped as Universal Output 9 The output signals come through CML/GTX logic block 3
Link TX (SFP)	LC	Optical 850 nm	Event link Transmit
Link RX (SFP)	LC	Optical 850 nm	Event link Receiver

3.4.1. VME TTL Input Levels

The VME-EVR-300 has two front panel TTL inputs. The inputs have a configurable input termination than can be set by a jumper. The input can be terminated with 50 ohm to ground or 220 ohm to +3.3V. The front panel inputs are 5V tolerant even when powered down.

Input specifications are following:

parameter	**value**
connector type	LEMO EPK.00.250.NTN
input impedance	50 ohm (jumper position Pull-down to GND)
input impedance	220 ohm (jumper position Pull-up to +3.3V)
V_{IH}	> 2.3 V
V_{IL}	< 1.0 V

3.5. PCIe-EVR-300DC and IFB-300 Connections

Due to its small bracket the PCIe-EVR-300DC has only a SFP transceiver and a micro-SCSI type connector to interface to the IFB-300. The cable between the PCIe-EVR-300DC and IFB-300 should be connected/disconnected only when powered down.

Connector / Led	**Style**	**Level**	**Description**
Link TX (SFP)	LC	Optical 850 nm	Event link Transmit Green: TX enable Red: Fract.syn. not locked Blue: Event out
Link RX (SFP) Next to micro-SCSI	LC	Optical 850 nm	Event link Receiver Green: link up Red: link violation detected Blue: event led

The interface board IFB-300 has eight Universal I/O slots which can be populated with various types of Universal I/O modules. If an input module is populated in any slot a jumper has to be mounted in that slot's two pin header with marking "Insert jumper for input module". Please note that if an input module is mounted the corresponding Universal Output Mapping has to be tri-stated. Refer to Table 1: Signal mapping IDs for details.

Universal Slot 0/1 signals are hard-wired to the TTLIN 0/1 signals.

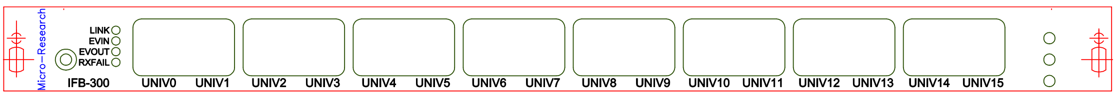

Figure 23: IFB-300 Front Panel

Connector / Led	**Style**	**Level**	**Description**
UNIV0/1	Universal slot		TTL Input / Universal I/O 0/1
UNIV2/3	Universal slot		Universal I/O 2/3
UNIV4/5	Universal slot		Universal I/O 4/5
UNIV6/7	Universal slot		Universal I/O 6/7
UNIV8/9	Universal slot		Universal I/O 8/9
UNIV10/11	Universal slot		Universal I/O 10/11

Connector / Led	Style	Level	Description
UNIV12/13	Universal slot		Universal I/O 12/13
UNIV14/15	Universal slot		Universal I/O 14/15
LINK	Green led		RX link up
EVIN	Yellow led		RX event in
EVOUT	Yellow led		RX event led (mapped)
RXFAIL	Red led		RX violation detected

3.6. PCIe-EVR-300DCS and IFB-300S Connections

Due to its small bracket the PCIe-EVR-300DCS has only a SFP transceiver and a Samtec Eye Speed I/O type connector to interface to the IFB-300S. The cable between the PCIe-EVR-300DCS and IFB-300S should be connected/disconnected only when powered down.

Connector / Led	Style	Level	Description
Link TX (SFP)	LC	Optical 850 nm	Event link Transmit Green: TX enable Red: Fract.syn. not locked Blue: Event out
Link RX (SFP) Next to Samtec connector	LC	Optical 850 nm	Event link Receiver Green: link up Red: link violation detected Blue: event led

The interface board IFB-300S has eight Universal I/O slots which can be populated with various types of Universal I/O modules. If an input module is populated in any slot a jumper has to be mounted in that slot's two pin header with marking “Insert jumper for input module”. Please note that if an input module is mounted the corresponding Universal Output Mapping has to be tri-stated. Refer to Table 1: Signal mapping IDs for details.

Universal Slot 0/1 signals are hard-wired to the TTLIN 0/1 signals.

Missing image.

Figure 24: IFB-300S Front Panel

Connector / Led	Style	Level	Description
UNIV0/1	Universal slot		TTL Input / Universal I/O 0/1
UNIV2/3	Universal slot		Universal I/O 2/3
UNIV4/5	Universal slot		Universal I/O 4/5
UNIV6/7	Universal slot		Universal I/O 6/7
UNIV8/9	Universal slot		Universal I/O 8/9
UNIV10/11	Universal slot		Universal I/O 10/11

Connector / Led	Style	Level	Description
UNIV12/13	Universal slot		Universal I/O 12/13
UNIV14/15	Universal slot		Universal I/O 14/15
LINK	Green led		RX link up
EVIN	Yellow led		RX event in
EVOUT	Yellow led		RX event led (mapped)
RXFAIL	Red led		RX violation detected

3.7. mTCA-EVR-300U Connections

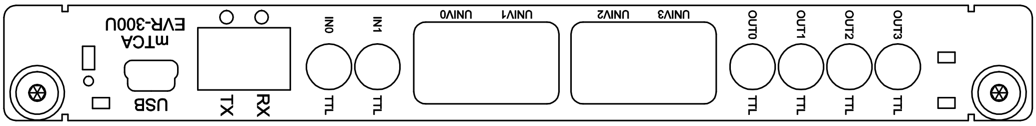

Figure 25: mTCA-EVR-300 Front Panel

Connector / Led	Style	Level	Description
USB	Micro-USB		MMC diagnostics serial port / JTAG interface
Link TX (SFP)	LC	Optical 850 nm	Event link Transmit Green: TX enable Red: Fract.syn. not locked Blue: Event out
Link RX (SFP)	LC	Optical 850 nm	Event link Receiver Green: link up Red: link violation detected Blue: event led
IN0	LEMO	TTL	FPTTL0 Trigger input
IN1	LEMO	TTL (3.3V / 5V)	FPTTL1 Trigger input
UNIV0/1	Universal slot		Universal Output 0/1 with GPIO pins mapped to GPIO register bits 3–0
UNIV2/3	Universal slot		Universal Output 2/3 with GPIO pins mapped to GPIO register bits 7–4
OUT0	LEMO	3.3V LVTTL	TTL Front panel output 0
OUT1	LEMO	3.3V LVTTL	TTL Front panel output 1
OUT2	LEMO	3.3V LVTTL	TTL Front panel output 2
OUT3	LEMO	3.3V LVTTL	TTL Front panel output 3
TCLKA	mTCA.4	LVDS	TCLKA clock on backplane This signal is driven by CML/GTX logic block 0 Mapped as Universal Output 16

Connector / Led	Style	Level	Description
TCLKB	mTCA.4	LVDS	TCLKB clock on backplane This signal is driven by CML/GTX logic block 1 Mapped as Universal Output 17
RX17	mTCA.4	MLVDS	Backplane output 0
TX17	mTCA.4	MLVDS	Backplane output 1
RX18	mTCA.4	MLVDS	Backplane output 2
TX18	mTCA.4	MLVDS	Backplane output 3
RX19	mTCA.4	MLVDS	Backplane output 4
TX19	mTCA.4	MLVDS	Backplane output 5
RX20	mTCA.4	MLVDS	Backplane output 6
TX20	mTCA.4	MLVDS	Backplane output 7

3.8. mTCA-EVR-300RF Connections

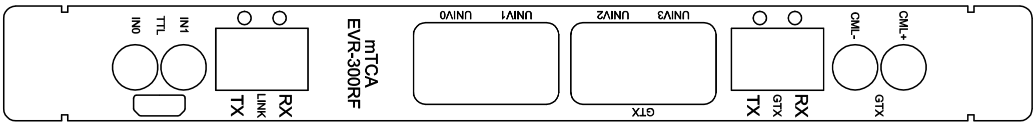

Figure 26: mTCA-EVR-300 Front Panel

Connector / Led	Style	Level	Description
USB	Micro-USB		MMC diagnostics serial port / JTAG interface
IN0	LEMO	TTL	FPTTL0 Trigger input
IN1	LEMO	TTL (3.3V / 5V)	FPTTL1 Trigger input
Link TX (SFP) Link RX (SFP)	LC LC	Optical 850 nm Optical 850 nm	Event link Transmit Green: TX enable Red: Fract.syn. not locked Blue: Event out Event link Receiver Green: link up Red: link violation detected Blue: event led
UNIV0/1	Universal slot		Universal Output 0/1 with GPIO pins mapped to GPIO register bits 3–0
UNIV2/3	Universal slot		Universal Output 2 This signal is driven by CML/GTX logic block 2 Mapped as Universal Output 18 Universal Output 3 This signal is driven by CML/GTX logic block 3 Mapped as Universal Output 19 with GPIO pins mapped to GPIO register bits 7–4

Connector / Led	Style	Level	Description
GTX SFP	SFP	Optical	Optical GTX Output for GUN-RC-300 This signal is driven by CML/GTX logic block 4 Mapped as Universal Output 20
CML-/CML+	LEMO	CML	This signal is driven by CML/GTX logic block 5 Mapped as Universal Output 21
TCLKA	mTCA.4	LVDS	TCLKA clock on backplane This signal is driven by CML/GTX logic block 0 Mapped as Universal Output 16
TCLKB	mTCA.4	LVDS	TCLKB clock on backplane This signal is driven by CML/GTX logic block 1 Mapped as Universal Output 17
RX17	mTCA.4	MLVDS	Backplane output 0
TX17	mTCA.4	MLVDS	Backplane output 1
RX18	mTCA.4	MLVDS	Backplane output 2
TX18	mTCA.4	MLVDS	Backplane output 3
RX19	mTCA.4	MLVDS	Backplane output 4
TX19	mTCA.4	MLVDS	Backplane output 5
RX20	mTCA.4	MLVDS	Backplane output 6
TX20	mTCA.4	MLVDS	Backplane output 7

3.9. mTCA-EVR-300 mTCA.4 Rear Transition Module Signals

The EVR supports driving and receiving signals to/from a RTM. The signal configuration is following:

Table 21: mTCA-EVR-300 RTM Signals

mTCA.4 J30/RP30 Pin	EVR RTM I/O Port
A3/B3	n/a
C3/D3	n/a
E3-/F3+	TBOut8
A4/B4	n/a
C4-/D4+	TBOut9
E4-/F4+	TBOut7
A5	SDA (I2C LED/GPIO control)
B5	SCL (I2C LED/GPIO control)
C5-/D5+	TBIn9
E5-/F5+	TBOut6
A6-/B6+	TBIn8
C6-/D6+	TBIn7
E6-/F6+	TBOut5

mTCA.4 J30/RP30 Pin	EVR RTM I/O Port
A7-/B7+	TBIn6
C7-/D7+	TBIn5
E7-/F7+	TBOut4
A8-/B8+	TBIn4
C8-/D8+	TBIn3
E8-/F8+	TBOut3
A9-/B9+	TBIn2
C9-/D9+	TBIn1
E9-/F9+	TBOut2
A10-/B10+	TBIn0
C10-/D10+	TBOut0
E10-/F10+	TBOut1

3.10. PCIe-EVR-300DC Firmware Upgrade

The PCIe-EVR-300DC firmware image can be upgraded with the following command after loading the driver in Linux:

```
dd if=new_image.bit of=/dev/era1
```

A power cycle is required to load the new configuration image on the PCIe-EVR-300DC.

3.11. Register Map

Address	Register	Type	Description
0x000	Status	UINT32	Status Register
0x004	Control	UINT32	Control Register
0x008	IrqFlag	UINT32	Interrupt Flag Register
0x00C	IrqEnable	UINT32	Interrupt Enable Register
0x010	PulseIrqMap	UINT32	Mapping register for pulse interrupt
0x018	SWEvent	UINT32	Software event register
0x01C	PCIIrqEnable	UINT32	PCI Interrupt Enable Register
0x020	DataBufCtrl	UINT32	Data Buffer Control and Status Register
0x024	TxDataBufCtrl	UINT32	TX Data Buffer Control and Status Register
0x028	TxSegBufCtrl	UINT32	TX Segmented Data Buffer Control and Status Register
0x02C	FWVersion	UINT32	Firmware Version Register
0x040	EvCntPresc	UINT32	Event Counter Prescaler

Address	Register	Type	Description
0x04C	UsecDivider	UINT32	Divider to get from Event Clock to 1 MHz
0x050	ClockControl	UINT32	Event Clock Control Register
0x05C	SecSR	UINT32	Seconds Shift Register
0x060	SecCounter	UINT32	Timestamp Seconds Counter
0x064	EventCounter	UINT32	Timestamp Event Counter
0x068	SecLatch	UINT32	Timestamp Seconds Counter Latch
0x06C	EvCntLatch	UINT32	Timestamp Event Counter Latch
0x070	EvFIFOSec	UINT32	Event FIFO Seconds Register
0x074	EvFIFOEvCnt	UINT32	Event FIFO Event Counter Register
0x078	EvFIFOCode	UINT16	Event FIFO Event Code Register
0x07C	LogStatus	UINT32	Event Log Status Register
0x080	FracDiv	UINT32	Micrel SY87739L Fractional Divider Configuration Word
0x090	GPIODir	UINT32	Front Panel UnivIO GPIO signal direction
0x094	GPIOIn	UINT32	Front Panel UnivIO GPIO input register
0x098	GPIOOut	UINT32	Front Panel UnivIO GPIO output register
0x0A0	SPIData	UINT32	SPI Data Register
0x0A4	SPIControl	UINT32	SPI Control Register
0x0B0	DCTarget	UINT32	Delay Compensation Target Value
0x0B4	DCRxValue	UINT32	Delay Compensation Transmission Path Delay Value
0x0B8	DCIntValue	UINT32	Delay Compensation Internal Delay Value
0x0BC	DCStatus	UINT32	Delay Compensation Status Register
0x0C0	TopologyID	UINT32	Timing Node Topology ID
0x0E0	SeqRamCtrl	UINT32	Sequence RAM Control Register
0x100	Prescaler0	UINT32	Prescaler 0 Divider
0x104	Prescaler1	UINT32	Prescaler 1 Divider
0x108	Prescaler2	UINT32	Prescaler 2 Divider
0x10C	Prescaler3	UINT32	Prescaler 3 Divider
0x110	Prescaler4	UINT32	Prescaler 4 Divider
0x114	Prescaler5	UINT32	Prescaler 5 Divider
0x118	Prescaler6	UINT32	Prescaler 6 Divider
0x11C	Prescaler7	UINT32	Prescaler 7 Divider
0x120	PrescPhase0	UINT32	Prescaler 0 Phase Offset Register
0x124	PrescPhase1	UINT32	Prescaler 1 Phase Offset Register
0x128	PrescPhase2	UINT32	Prescaler 2 Phase Offset Register
0x12C	PrescPhase3	UINT32	Prescaler 3 Phase Offset Register
0x130	PrescPhase4	UINT32	Prescaler 4 Phase Offset Register

Address	Register	Type	Description
0x134	PrescPhase5	UINT32	Prescaler 5 Phase Offset Register
0x138	PrescPhase6	UINT32	Prescaler 6 Phase Offset Register
0x13C	PrescPhase7	UINT32	Prescaler 7 Phase Offset Register
0x140	PrescTrig0	UINT32	Prescaler 0 Pulse Generator Trigger Register
0x144	PrescTrig1	UINT32	Prescaler 1 Pulse Generator Trigger Register
0x148	PrescTrig2	UINT32	Prescaler 2 Pulse Generator Trigger Register
0x14C	PrescTrig3	UINT32	Prescaler 3 Pulse Generator Trigger Register
0x150	PrescTrig4	UINT32	Prescaler 4 Pulse Generator Trigger Register
0x154	PrescTrig5	UINT32	Prescaler 5 Pulse Generator Trigger Register
0x158	PrescTrig6	UINT32	Prescaler 6 Pulse Generator Trigger Register
0x15C	PrescTrig7	UINT32	Prescaler 7 Pulse Generator Trigger Register
0x180	DBusTrig0	UINT32	DBus Bit 0 Pulse Generator Trigger Register
0x184	DBusTrig1	UINT32	DBus Bit 1 Pulse Generator Trigger Register
0x188	DBusTrig2	UINT32	DBus Bit 2 Pulse Generator Trigger Register
0x18C	DBusTrig3	UINT32	DBus Bit 3 Pulse Generator Trigger Register
0x190	DBusTrig4	UINT32	DBus Bit 4 Pulse Generator Trigger Register
0x194	DBusTrig5	UINT32	DBus Bit 5 Pulse Generator Trigger Register
0x198	DBusTrig6	UINT32	DBus Bit 6 Pulse Generator Trigger Register
0x19C	DBusTrig7	UINT32	DBus Bit 7 Pulse Generator Trigger Register
0x200	Pulse0Ctrl	UINT32	Pulse 0 Control Register
0x204	Pulse0Presc	UINT32	Pulse 0 Prescaler Register
0x208	Pulse0Delay	UINT32	Pulse 0 Delay Register
0x20C	Pulse0Width	UINT32	Pulse 0 Width Register
0x210			Pulse 1 Registers
0x220			Pulse 2 Registers
...	...	...	...
0x3F0			Pulse 31 Registers
0x400	FPOutMap0	UINT16	Front Panel Output 0 Map Register
0x402	FPOutMap1	UINT16	Front Panel Output 1 Map Register
0x404	FPOutMap2	UINT16	Front Panel Output 2 Map Register
0x406	FPOutMap3	UINT16	Front Panel Output 3 Map Register
0x408	FPOutMap4	UINT16	Front Panel Output 4 Map Register
0x40A	FPOutMap5	UINT16	Front Panel Output 5 Map Register
0x40C	FPOutMap6	UINT16	Front Panel Output 6 Map Register
0x40E	FPOutMap7	UINT16	Front Panel Output 7 Map Register
0x440	UnivOutMap0	UINT16	Front Panel Universal Output 0 Map Register
0x442	UnivOutMap1	UINT16	Front Panel Universal Output 1 Map Register

Address	Register	Type	Description
0x444	UnivOutMap2	UINT16	Front Panel Universal Output 2 Map Register
0x446	UnivOutMap3	UINT16	Front Panel Universal Output 3 Map Register
0x448	UnivOutMap4	UINT16	Front Panel Universal Output 4 Map Register
0x44A	UnivOutMap5	UINT16	Front Panel Universal Output 5 Map Register
0x44C	UnivOutMap6	UINT16	Front Panel Universal Output 6 Map Register
0x44E	UnivOutMap7	UINT16	Front Panel Universal Output 7 Map Register
0x450	UnivOutMap8	UINT16	Front Panel Universal Output 8 Map Register
0x452	UnivOutMap9	UINT16	Front Panel Universal Output 9 Map Register
0x454	UnivOutMap10	UINT16	Front Panel Universal Output 10 Map Register
0x456	UnivOutMap11	UINT16	Front Panel Universal Output 11 Map Register
0x458	UnivOutMap12	UINT16	Front Panel Universal Output 12 Map Register
0x45A	UnivOutMap13	UINT16	Front Panel Universal Output 13 Map Register
0x45C	UnivOutMap14	UINT16	Front Panel Universal Output 14 Map Register
0x45E	UnivOutMap15	UINT16	Front Panel Universal Output 15 Map Register
0x460	UnivOutMap16	UINT16	Front Panel Universal Output 16 Map Register
0x462	UnivOutMap17	UINT16	Front Panel Universal Output 17 Map Register
0x480	TBOutMap0	UINT16	Transition Board Output 0 Map Register
0x482	TBOutMap1	UINT16	Transition Board Output 1 Map Register
0x484	TBOutMap2	UINT16	Transition Board Output 2 Map Register
0x486	TBOutMap3	UINT16	Transition Board Output 3 Map Register
0x488	TBOutMap4	UINT16	Transition Board Output 4 Map Register
0x48A	TBOutMap5	UINT16	Transition Board Output 5 Map Register
0x48C	TBOutMap6	UINT16	Transition Board Output 6 Map Register
0x48E	TBOutMap7	UINT16	Transition Board Output 7 Map Register
0x490	TBOutMap8	UINT16	Transition Board Output 8 Map Register
0x492	TBOutMap9	UINT16	Transition Board Output 9 Map Register
0x494	TBOutMap10	UINT16	Transition Board Output 10 Map Register
0x496	TBOutMap11	UINT16	Transition Board Output 11 Map Register
0x498	TBOutMap12	UINT16	Transition Board Output 12 Map Register
0x49A	TBOutMap13	UINT16	Transition Board Output 13 Map Register
0x49C	TBOutMap14	UINT16	Transition Board Output 14 Map Register
0x49E	TBOutMap15	UINT16	Transition Board Output 15 Map Register
0x4A0	TBOutMap16	UINT16	Transition Board Output 16 Map Register
0x4A2	TBOutMap17	UINT16	Transition Board Output 17 Map Register
0x4A4	TBOutMap18	UINT16	Transition Board Output 18 Map Register
0x4A6	TBOutMap19	UINT16	Transition Board Output 19 Map Register
0x4A8	TBOutMap20	UINT16	Transition Board Output 20 Map Register

Address	Register	Type	Description
0x4AA	TBOutMap21	UINT16	Transition Board Output 21 Map Register
0x4AC	TBOutMap22	UINT16	Transition Board Output 22 Map Register
0x4AE	TBOutMap23	UINT16	Transition Board Output 23 Map Register
0x4B0	TBOutMap24	UINT16	Transition Board Output 24 Map Register
0x4B2	TBOutMap25	UINT16	Transition Board Output 25 Map Register
0x4B4	TBOutMap26	UINT16	Transition Board Output 26 Map Register
0x4B6	TBOutMap27	UINT16	Transition Board Output 27 Map Register
0x4B8	TBOutMap28	UINT16	Transition Board Output 28 Map Register
0x4BA	TBOutMap29	UINT16	Transition Board Output 29 Map Register
0x4BC	TBOutMap30	UINT16	Transition Board Output 30 Map Register
0x4BE	TBOutMap31	UINT16	Transition Board Output 31 Map Register
0x4C0	BPOutMap0	UINT16	Backplane Output 0 Map Register
0x4C2	BPOutMap1	UINT16	Backplane Output 1 Map Register
0x4C4	BPOutMap2	UINT16	Backplane Output 2 Map Register
0x4C6	BPOutMap3	UINT16	Backplane Output 3 Map Register
0x4C8	BPOutMap4	UINT16	Backplane Output 4 Map Register
0x4CA	BPOutMap5	UINT16	Backplane Output 5 Map Register
0x4CC	BPOutMap6	UINT16	Backplane Output 6 Map Register
0x4CE	BPOutMap7	UINT16	Backplane Output 7 Map Register
0x500	FPInMap0	UINT32	Front Panel Input 0 Mapping Register
0x504	FPInMap1	UINT32	Front Panel Input 1 Mapping Register
...	...	...	...
0x510	UnivInMap0	UINT32	Universal Input 0 Mapping Register
0x514	UnivInMap1	UINT32	Universal Input 1 Mapping Register
...	...	...	...
0x560	BPInMap0	UINT32	Backplane Input 0 Mapping Register
0x564	BPInMap1	UINT32	Backplane Input 1 Mapping Register
...	...	...	...
0x580	TBDly0	UINT32	Transition Board -DLY Module Output 0 Delay Value 0-1023
0x584	TBDly1	UINT32	Transition Board -DLY Module Output 1 Delay Value 0-1023
...	...	...	...
0x5C0	TBInMap0	UINT32	Transition Board Input 0 Mapping Register
0x5C4	TBInMap1	UINT32	Transition Board Input 1 Mapping Register
...	...	...	...
0x610	GTX0Ctrl	UINT32	GTX0 Output Control Register
0x614	GTX0HP	UINT16	GTX0 Output High Period Count

Address	Register	Type	Description
0x616	GTX0LP	UINT16	GTX0 Output Low Period Count
0x618	GTX0Samp	UINT32	GTX0 Output Number of 40 bit word patterns
0x61C	GTX0PS	UINT32	GTX0 Phase Shift Offset Register
0x630	GTX1Ctrl	UINT32	GTX1 Output Control Register
0x634	GTX1HP	UINT16	GTX1 Output High Period Count
0x636	GTX1LP	UINT16	GTX1 Output Low Period Count
0x638	GTX1Samp	UINT32	GTX1 Output Number of 40 bit word patterns
0x63C	GTX1PS	UINT32	GTX1 Phase Shift Offset Register
0x650	GTX2Ctrl	UINT32	GTX2 Output Control Register
0x654	GTX2HP	UINT16	GTX2 Output High Period Count
0x656	GTX2LP	UINT16	GTX2 Output Low Period Count
0x658	GTX2Samp	UINT32	GTX2 Output Number of 40 bit word patterns
0x65C	GTX2PS	UINT32	GTX2 Phase Shift Offset Register
0x670	GTX3Ctrl	UINT32	GTX3 Output Control Register
0x674	GTX3HP	UINT16	GTX3 Output High Period Count
0x676	GTX3LP	UINT16	GTX3 Output Low Period Count
0x678	GTX3Samp	UINT32	GTX3 Output Number of 40 bit word patterns
0x67C	GTX3PS	UINT32	GTX3 Phase Shift Offset Register
0x690	GTX4Ctrl	UINT32	GTX4 Output Control Register
0x694	GTX4HP	UINT16	GTX4 Output High Period Count
0x696	GTX4LP	UINT16	GTX4 Output Low Period Count
0x698	GTX4Samp	UINT32	GTX4 Output Number of 40 bit word patterns
0x69C	GTX4PS	UINT32	GTX4 Phase Shift Offset Register
0x6B0	GTX5Ctrl	UINT32	GTX5 Output Control Register
0x6B4	GTX5HP	UINT16	GTX5 Output High Period Count
0x6B6	GTX5LP	UINT16	GTX5 Output Low Period Count
0x6B8	GTX5Samp	UINT32	GTX5 Output Number of 40 bit word patterns
0x6BC	GTX5PS	UINT32	GTX5 Phase Shift Offset Register
0x800 – 0xFFF	DataBuf		Data Buffer Receive Memory
0x1000 – 0x17FF			Diagnostics counters
0x1800 – 0x1FFF	TxDataBuf		Data Buffer Transmit Memory
0x2000 – 0x3FFF	EventLog		512 x 16 byte position Event Log
0x4000 – 0x4FFF	MapRam1		Event Mapping RAM 1
0x5000 – 0x5FFF	MapRam2		Event Mapping RAM 2
0x6000	EventCnt0	UINT32	Event 0 Counter Register
0x6004	EventCnt1	UINT32	Event 1 Counter Register
...	...	...	...

Address	Register	Type	Description
0x63FC	EventCnt255	UINT32	Event 255 Counter Register
0x6400	PulseCnt0	UINT32	Pulse 0 Counter Register
0x6404	PulseCnt1	UINT32	Pulse 1 Counter Register
...	...	...	...
0x647C	PulseCnt31	UINT32	Pulse 31 Counter Register
0x7200	Pulse0RepCnt	UINT32	Pulse 0 Repetition Count Register
0x7204	Pulse0RepDelay	UINT32	Pulse 0 Repeat Delay Register
0x7210			Pulse 1 Rep. Registers
0x7220			Pulse 2 Rep. Registers
. . .	. . .	. . .	. . .
0x73F0			Pulse 31 Rep. Registers
0x8000 – 0x80FF	configROM		
0x8100 – 0x81FF	scratchRAM		
0x8200 – 0x82FF	SFPEEPROM		SFP Transceiver EEPROM contents (SFP address 0xA0)
0x8300 – 0x83FF	SFPDIAG		SFP Transceiver diagnostics (SFP address 0xA2)
0x8800	DataBufRXSize0	UINT32	Segmented Data Buffer Segment 0 Receive Size Register
0x8804	SDataBufRXSize0	UINT32	Segmented Data Buffer Segment 1 Receive Size Register
. . .	. . .	. . .	. . .
0x89FC	SDataBufRXSize127	UINT32	Segmented Data Buffer Segment 127 Receive Size Register
0x8F80 – 0x8F8F	SDataBufSIrqEna		Segmented Data Buffer Segment Interrupt Enable Register
0x8FA0 – 0x8FAF	SDataBufCSFlag		Segmented Data Buffer Segment Checksum Flags
0x8FC0 – 0x8FCF	SDataBufOVFlag		Segmented Data Buffer Segment Overflow Flags
0x8FE0 – 0x8FEF	SDataBufRxFlag		Segmented Data Buffer Segment Receive Flags
0x9000 – 0x97FF	SDataBufData		Segmented Data Buffer Segment Data Memory
0xA000 – 0xA7FF	SDataBufData		Segmented Data Buffer Transmit Memory
0xC000 – 0xFFFF	SeqRam		Sequence RAM
0x20000 – 0x23FFF	GTX0MEM		Pattern memory: 16k bytes GTX output 0 (VME-EVR-300) 16k bytes GTX output 0 (mTCA-EVR-300U/RF TCLKA)
0x24000 – 0x27FFF	GTX1MEM		Pattern memory: 16k bytes GTX output 1 (VME-EVR-300) 16k bytes GTX output 1 (mTCA-EVR-300U/RF TCLKB)

Address	Register	Type	Description
0x28000 – 0x2BFFF	GTX2MEM		Pattern memory: 16k bytes GTX output 2 (VME-EVR-300) 16k bytes GTX output 2 (mTCA-EVR-300RF UNIV2)
0x2C000 – 0x2FFFF	GTX3MEM		Pattern memory: 16k bytes GTX output 3 (VME-EVR-300) 16k bytes GTX output 3 (mTCA-EVR-300RF UNIV3)
0x30000 – 0x33FFF	GTX4MEM		Pattern memory: 16k bytes GTX output 4 (mTCA-EVR-300RF GTX/GUN-TX)
0x34000 – 0x37FFF	GTX5MEM		Pattern memory: 16k bytes GTX output 5 (mTCA-EVR-300RF GTX CML)

3.11.1. Status Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x000	DBUS7	DBUS6	DBUS5	DBUS4	DBUS3	DBUS2	DBUS1	DBUS0

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x001								LEGVIO

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x002								

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x003	SFPMOD	LINK	LOGSTP					

Bit	Function
DBUS7	Read status of DBUS bit 7
DBUS6	Read status of DBUS bit 6
DBUS5	Read status of DBUS bit 5
DBUS4	Read status of DBUS bit 4
DBUS3	Read status of DBUS bit 3
DBUS2	Read status of DBUS bit 2
DBUS1	Read status of DBUS bit 1
DBUS0	Read status of DBUS bit 0
LEGVIO	Legacy VIO (series 100, 200 and 230)
SFPMOD	SFP module status: ‘0’ – plugged in ‘1’ – no module installed

Bit	Function
LINK	Link status: ‘0’ – link down ‘1’ – link up
LOGSTP	Event Log stopped flag

3.11.2. Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x004	EVREN	EVFWD	TXLP	RXLP	OUTEN	SRST	LEMDE	GTXIO

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x005		DCENA						

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x006	PRPOL	TSDBUS	RSTS			LTS	MAPEN	MAPRS

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x007	LOGRS	LOGEN	LOGDIS	LOGSE	RSFIFO			

Bit	Function
EVREN	Event Receiver Master enable
EVFWD	Event forwarding enable: 0 – Events not forwarded 1 – Events received with forward bit in mapping RAM set are sent back on TX
TXLP	Transmitter loopback: 0 – Receive signal from SFP transceiver (normal operation) 1 – Loopback EVR TX into EVR RX
RXLP	Receiver loopback: 0 – Transmit signal from EVR on SFP transceiver TX 1 – Loopback SFP RX on SFP TX
OUTEN	Output enable for FPGA external components / IFB-300 (cPCI-EVRTG-300, PCIe-EVR-300, PXIe-EVR-300I) 0 – disable outputs 1 – enable outputs
SRST	Soft reset IP
LEMDE	Little endian mode (cPCI-EVR-300, PCIe-EVR-300) 0 – PCI core in big endian mode (power up default) 1 – PCI core in little endian mode
GTXIO	GUN-TX output hardware inhibit override 0 – honor hardware inhibit signal (default) 1 – inhibit override, don't care about hardware inhibit input state
DCENA	Delay compensation mode enable 0 – Delay compensation mode disable (receive FIFO depth controlled by DC Target). 1 – Delay compensation mode enable (receive FIFO depth controlled by DC Target - Datapath Delay).
PRPOL	Prescaler Polarity select 0 – Prescalers aligned on falling edge 1 – Prescalers aligned on rising edge
TSDBUS	Use timestamp counter clock on DBUS4
RSTS	Reset Timestamp. Write 1 to reset timestamp event counter and timestamp latch.
LTS	Latch Timestamp: Write 1 to latch timestamp from timestamp event counter to timestamp latch.
MAPEN	Event mapping RAM enable.
MAPRS	Mapping RAM select bit for event decoding: 0 – select mapping RAM 1 1 – select mapping RAM 2.
LOGRS	Reset Event Log. Write 1 to reset log.
LOGEN	Enable Event Log. Write 1 to (re)enable event log.
LOGDIS	Disable Event Log. Write 1 to disable event log.
LOGSE	Log Stop Event Enable.
RSFIFO	Reset Event FIFO. Write 1 to clear event FIFO.

Bit	Function

3.11.3. Interrupt Flag Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x008								

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x009				IFSOV				IFSHF

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x00A				IFSSTO				IFSSTA

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x00B	IFSEGD	IFLINK	IFDBUF	IFHW	IFEV	IFHB	IFFF	IFVIO

Bit	Function
IFSOV	Sequence RAM sequence roll over interrupt flag
IFSHF	Sequence RAM sequence halfway through interrupt flag
IFSSTO	Sequence RAM sequence stop interrupt flag
IFSSTA	Sequence RAM sequence start interrupt flag
IFSEGD	Segmented data buffer interrupt flag
IFLINK	Link state change interrupt flag
IFDBUF	Data buffer interrupt flag
IFHW	Hardware interrupt flag (mapped signal)
IFEV	Event interrupt flag
IFHB	Heartbeat interrupt flag
IFFF	Event FIFO full flag
IFVIO	Receiver violation flag

3.11.4. Interrupt Enable Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x00C	IRQEN							

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x00D				IESOV				IESHF

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x00E				IESSTO				IESSTA

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x00F	IESEGD	IELINK	IEDBUF	IEHW	IEEV	IEHB	IEFF	IEVIO

Bit	Function
IRQEN	Master interrupt enable: 0 – disable all interrupts 1 – allow interrupts
IESOV	Sequence RAM sequence roll over interrupt enable
IESHF	Sequence RAM sequence halfway through interrupt enable
IESSTO	Sequence RAM sequence stop interrupt enable
IESSTA	Sequence RAM sequence start interrupt enable
IESEGD	Segmented data buffer interrupt enable
IELINK	Link state change interrupt enable
IEDBUF	Data buffer interrupt enable
IEHW	Hardware interrupt enable (mapped signal)
IEEV	Event interrupt enable
IEHB	Heartbeat interrupt enable
IEFF	Event FIFO full interrupt enable
IEVIO	Receiver violation interrupt enable

3.11.5. Hardware Interrupt Mapping Register

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x013	Mapping ID (see Table 1 for mapping IDs)							

3.11.6. Software Event Register

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x01A							SWPEND	SWENA

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x01B	Event Code to be inserted into receive event stream							

Bit	Function
SWPEND	Event code waiting to be inserted (read-only). A new event code may be written to the event code register when this bit reads ‘0’.
SWENA	Enable software event When enabled ‘1’ a new event will be inserted into the receive event stream when event code is written to the event code register.

3.11.7. PCI Interrupt Enable Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x01c		PCIIE						

Bit	Function
PCIIE	PCI core interrupt enable (PCIe-EVR-300DC, mTCA-EVR-300) This bit is used by the low level driver to disable further interrupts before the first interrupt has been handled in user space

3.11.8. Receive Data Buffer Control and Status Register

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x022	DBRX/ DBENA	DBRDY/ DBDIS	DBCS	DBEN	RXSIZE(11:8)			

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x023	RXSIZE(7:0)							

Bit	Function
DBRX	Data Buffer Receiving (read-only)
DBENA	Set-up for Single Reception (write ‘1’ to set-up)
DBRDY	Data Buffer Transmit Complete / Interrupt Flag
DBDIS	Stop Reception (write ‘1’ to stop/disable)
DBCS	Data Buffer Checksum Error (read-only) Flag is cleared by writing ‘1’ to DBRX or DBRDY or disabling data buffer
DBEN	Data Buffer Enable Data Buffer Mode ‘1’ – Distributed bus shared with data transmission, half speed distributed bus
RXSIZE	Data Buffer Received Buffer Size (read-only)

3.11.9. Transmit Data Buffer Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x024								

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x025				TXCPT	TXRUN	TRIG	ENA	1

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x026						DTSZ(10:8)		

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x027	DTSZ(7:2)						0	0

Bit	Function
TXCPT	Data Buffer Transmission Complete
TXRUN	Data Buffer Transmission Running – set when data transmission has been triggered and has not been completed yet

Bit	Function
TRIG	Data Buffer Trigger Transmission Write ‘1’ to start transmission of data in buffer
ENA	Data Buffer Transmission enable ‘0’ – data transmission engine disabled ‘1’ – data transmission engine enabled
DTSZ(10:8)	Data Transfer size 4 bytes to 2k in four byte increments

3.11.10. Transmit Segemented Data Buffer Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x028	SADDR(7:0)							

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x029				TXCPT	TXRUN	TRIG	ENA	MODE

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x02A						DTSZ(10:8)		

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x02B	DTSZ(7:2)						0	0

Bit	Function
SADDR	Transfer Start Segment Address (16 byte segments)
TXCPT	Data Buffer Transmission Complete
TXRUN	Data Buffer Transmission Running – set when data transmission has been triggered and has not been completed yet
TRIG	Data Buffer Trigger Transmission Write ‘1’ to start transmission of data in buffer
ENA	Data Buffer Transmission enable ‘0’ – data transmission engine disabled ‘1’ – data transmission engine enabled
MODE	Distributed bus sharing mode ‘1’ – distributed bus shared with data transmission
DTSZ(10:8)	Data Transfer size 4 bytes to 2k in four byte increments

3.11.11. FPGA Firmware Version Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x02C	EVR = 0x1				Form Factor			

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x02D	Subrelease ID							

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x02E	Firmware ID							

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x02F	Revision ID							

Bit	Function
Form Factor	0 – CompactPCI 3U 1 – PMC 2 – VME64x 3 – CompactRIO 4 – CompactPCI 6U 6 – PXIe 7 – PCIe 8 – mTCA.4
Subrelease ID	For production releases the subrelease ID counts up from 00. For pre-releases this ID is used “backwards” counting down from ff i.e. when approacing release 12000207, we have prereleases 12FF0206, 12FE0206, 12FD0206 etc. in this order.
Firmware ID	00 – Modular Register Map firmware (no delay compensation) 01 – Reserved 02 – Delay Compensation firmware
Revision ID	See end of manual

3.11.12. Clock Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x050	PLLLOCK	BWSEL(2:0)				CLKMD(1:0)		

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x052							CGLOCK	

Bit	Function
PLLLOCK	Clock cleaner PLL Locked (read-only) to receiver recovered clock
BWSEL2:0	PLL Bandwidth Select (see Silicon Labs Si5317 datasheet) 000 – Si5317, BW setting HM (lowest loop bandwidth) 001 – Si5317, BW setting HL 010 – Si5317, BW setting MH 011 – Si5317, BW setting MM 100 – Si5317, BW setting ML (highest loop bandwidth)

Bit	Function
CLKMD1:0	Event clock mode 00 – Event clock synchronized to upstream EVG. Event clock continues to run with same frequency if link is lost. 01 – Event clock synchronized to local fractional synthesizer reference. 10 – Event clock synchronized to upstream EVG. Fall back to local reference if upstream link is lost. 11 – Event clock synchronized to upstream EVG. Event clock is stopped if link is lost.
CGLOCK	Micrel fractional synthesizer SY87739L locked (read-only). This serves as the reference clock for the FPGA internal transceiver and indicates that a valid configuration word has been set in the FracDiv control register.

3.11.13. Event FIFO

Note that reading the FIFO event code registers pulls the event code and timestamp/seconds value from the FIFO for access. The correct order to read an event from FIFO is to first read the event code register and after this the timestamp/seconds registers in any order. Every read access to the FIFO event register pulls a new event from the FIFO if it is not empty.

3.11.14. SY87739L Fractional Divider Configuration Word

The fractional synthesizer serves as the reference clock for the FPGA internal transceiver.

Configuration Word	Frequency with 24 MHz reference oscillator
0x0891C100	142.857 MHz
0x00DE816D	125 MHz
0x00FE816D	124.95 MHz
0x0C928166	124.9087 MHz
0x018741AD	119 MHz
0x072F01AD	114.24 MHz
0x049E81AD	106.25 MHz
0x008201AD	100 MHz
0x025B41ED	99.956 MHz
0x0187422D	89.25 MHz
0x0082822D	81 MHz
0x0106822D	80 MHz
0x019E822D	78.900 MHz
0x018742AD	71.4 MHz
0x0C9282A6	62.454 MHz

Configuration Word	Frequency with 24 MHz reference oscillator
0x009743AD	50 MHz
0x0C25B43AD	49.978 MHz
0x0176C36D	49.965 MHz

3.11.15. GPIO Pin Direction Register

address	bit 31 … bit 0
0x090	GPDIR[31:0]

Bit	Function
GPDIR	GPIO I/O Direction 0 : input 1 : output

3.11.16. SPI Configuration Flash Registers

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x0A3	SPIDATA(7:0)							

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x0A7	E	RRDY	TRDY	TMT	TOE	ROE	OE	SSO

Bit	Function
SPIDATA(7:0)	Read SPI data byte / Write SPI data byte
E	Overrun Error flag
RRDY	Receiver ready, if ‘1’ data byte waiting in SPI_DATA
TRDY	Transmitter ready, if ‘1’ SPI_DATA is ready to accept new transmit data byte
TMT	Transmitter empty, if ‘1’ data byte has been transmitted
TOE	Transmitter overrun error
ROE	Receiver overrun error
OE	Output enable for SPI pins, ‘1’ enable SPI pins
SSO	Slave select output enable for SPI slave device, ‘1’ device selected

3.11.17. Delay Compensation Status Register

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x0BE						PDVLD(2:0)		

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x0BF					DLYL	DLYS		DCLOCK

Bit	Function
PDVLD(2:0)	Path delay valid 000 – Path delay value not valid from master EVM to EVR 001 – Path delay value valid (coarse/quick acquisition) 011 – Path delay value valid (medium precision/slow acquisition)
111 – Path delay value valid (fine precision/slow acquisition)	
DLYL	Delay setting too long (delay shorter than target)
DLYS	Delay setting too short (delay longer than target)
DCLOCK	Delay fifo locked to setting/delay value

3.11.18. Sequence RAM Control Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x0E0							SQRUN	SQENA

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x0E1			SQSWT	SQSNG	SQREC	SQRES	SQDIS	SQEN

address	bit 7 … bit 0
0x0E3	SQTSEL

Bit	Function
SQRUN	Sequence RAM running flag (read-only)
SQENA	Sequence RAM enabled flag (read_only)
SQSWT	Sequence RAM software trigger, write ‘1’ to trigger
SQSNG	Sequence RAM single mode
SQREC	Sequence RAM recycle mode
SQRES	Sequence RAM reset, write ‘1’ to reset
SQDIS	Sequence RAM disable, write ‘1’ to disable
SQEN	Sequence RAM enable, write ‘1’ to enable/arm
SQTSEL	0 to n-1 – Pulse generator output n to 31 – (Reserved) 32 to 39 – Distributed bus bit 0 (DBUS0) to bit 7 (DBUS7) 40 to 47 – Prescaler 0 to Prescaler 7 48 to 58 – (Reserved) 61 – Software trigger 62 – Continuous trigger 63 – Trigger disabled

Bit	Function

3.11.19. Prescaler Pulse Trigger Registers

Each bit in the Prescaler Pulse Trigger Register corresponds to one pulse generator trigger. If for instance bit 0 is set, pulse generator 0 gets trigger on each 0 to 1 transition of the corresponding prescaler.

3.11.20. Distributed Bus Pulse Trigger Registers

Each bit in the Distributed Bus Pulse Trigger Register corresponds to one pulse generator trigger. If for instance bit 0 is set, pulse generator 0 gets trigger on each 0 to 1 transition of the corresponding distributed bus bit.

3.11.21. Pulse Generator Registers

Please note: addition pulse generator registers are located starting at offset 0x7200.

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x200	PxMASK(3:0)							

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x201	PxMENA(3:0)							

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x202							PxRME	PxSME

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x203	PxOUT	PxSWS	PxSWR	PxPOL	PxMRE	PxMSE	PxMTE	PxENA

address	bit 31	bit 0
0x204	Pulse Generator 0 Prescaler Register	

address	bit 31	bit 0
0x208	Pulse Generator 0 Delay Register	

address	bit 31	bit 0
0x20C	Pulse Generator 0 Width Register	

Bit	Function
PxMASK(3:0)	Pulse HW Mask Register 0 – HW masking disabled 1 – HW masking enabled. When corresponding gate bit is active ‘1’ pulse triggers are blocked

Bit	Function
PxMENA(3:0)	Pulse HW Enable Register 0 – HW enabling inactive 1 – HW enabling active. When corresponding gate bit is inactive ‘0’ pulse triggers are blocked
PxRME	Pulse Generator Apply mask to HW Reset
PxSME	Pulse Generator Apply mask to HW Set
PxOUT	Pulse Generator Output (read-only)
PxSWS	Pulse Generator Software Set
PxSWC	Pulse Generator Software Reset
PxPOL	Pulse Generator Output Polarity 0 – normal polarity 1 – inverted polarity
PxMRE	Pulse Generator Event Mapping RAM Reset Event Enable 0 – Reset events disabled 1 – Mapped Reset Events reset pulse generator output
PxMSE	Pulse Generator Event Mapping RAM Set Event Enable 0 – Set events disabled 1 – Mapped Set Events set pulse generator output
PxMTE	Pulse Generator Event Mapping RAM Trigger Event Enable 0 – Event Triggers disabled 1 – Mapped Trigger Events trigger pulse generator
PxENA	Pulse Generator Enable 0 – generator disabled 1 – generator enabled

3.11.22. Input Mapping Registers

The same bit mapping applies to Front Panel Inputs, Universal Inputs and Backplane Inputs.

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x500	FPIN0		EXTLV0	BCKLE0	EXTLE0	EXTED0	BCKEV0	EXTEV0

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x501	T0DB7	T0DB6	T0DB5	T0DB4	T0DB3	T0DB2	T0DB1	T0DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x502	Backward Event Code Register for front panel input 0							

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x503	External Event Code Register for front panel input 0							

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x504	FPIN1		EXTLV1	BCKLE1	EXTLE1	EXTED1	BCKEV1	EXTEV1

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x505	T1DB7	T1DB6	T1DB5	T1DB4	T1DB3	T1DB2	T1DB1	T1DB0

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x506	Backward Event Code Register for front panel input 1							

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x507	External Event Code Register for front panel input 1							

Bit	Function
FPINx	Front panel Input x state. 0 – low 1 – high
EXTLVx	Backward HW Event Level Sensitivity for input x 0 – active high 1 – active low
BCKLEx	Backward HW Event Level Trigger enable for input x 0 – disable level events 1 – enable level events, send out backward event code every 1 us when input is active (see EXTLVx for level sensitivity)
EXTLEx	External HW Event Level Trigger enable for input x 0 – disable level events 1 – enable level events, apply external event code to active mapping RAM every 1 us when input is active (see EXTLVx for level sensitivity)
EXTEDx	Backward HW Event Edge Sensitivity for input x 0 – trigger on rising edge 1 – trigger on falling edge
BCKEVx	Backward HW Event Edge Trigger Enable for input x 0 – disable backward HW event 1 – enable backward HW event, send out backward event code on detected edge of hardware input (see EXTEDx bit for edge)
EXTEVx	External HW Event Enable for input x 0 – disable external HW event 1 – enable external HW event, apply external event code to active mapping RAM on edge of hardware input
TxDB7-TxDB0	Backward distributed bus bit enable: 0 – disable distributed bus bit 1 – enable distributed bus bit control from hardware input: e.g. when TxDB7 is ‘1’ the hardware input x state is sent out on distributed bus bit 7.

3.11.23. GTX Output Control Register

address	bit 31 … bit 16
0x610	Frequency mode trigger position

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x612			GTX3RP	GTX3TP	GTX3MD	GTX2MD	GTXPH1	GTXPH0

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x613	CMLRC	CMLTL	CMLMD(1:0)			CMLRES	CMLPWD	CMLENA

Bit	Function
GTX3RP	GTX input polarity
GTX3TP	GTX output polarity
GTX3MD	GUN-TX-300 Mode (cPCI-EVRTG-300 only) 0 – CML/GTX Mode 1 – SFP output in GUN-TX-300 Mode
GTX2MD	GUN-TX-203 Mode (cPCI-EVRTG-300 only) 0 – CML/GTX Mode 1 – SFP output in GUN-TX-203 Mode
GTXPH1:0	GUN-TX-203 Trigger output phase shift (cPCI-EVRTG-300 only) 00 – no delay 01 – output pulse delayed by ï¿½ event clock period (˜2 ns) 10 – output pulse delayed by ï¿½ event clock period (˜4 ns) 11 – output pulse delayed by ï¿½ event clock period (˜6 ns)
CMLRC	CML Pattern recycle
CMLTL	CML Frequency mode trigger level
CMLMD	CML Mode Select: 00 = classic mode 01 = frequency mode 10 = pattern mode 11 = undefined
CMLRES	CML Reset 1 = reset CML output (default on EVR power up) 0 = normal operation
CMLPWD	CML Power Down 1 = CML outputs powered down (default on EVR power up) 0 = normal operation
CMLENA	CML Enable 0 = CML output disabled (default on EVR power up) 1 = CML output enabled

3.11.24. Data Buffer Segment Interrupt Enable Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x8F80	DBIE00	DBIE01	DBIE02	DBIE03	DBIE04	DBIE05	DBIE06	DBIE07

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x8F81	DBIE08	DBIE09	DBIE0A	DBIE0B	DBIE0C	DBIE0D	DBIE0E	DBIE0F

. . .

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x8F8E	DBIE70	DBIE71	DBIE72	DBIE73	DBIE74	DBIE75	DBIE76	DBIE77

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x8F8F	DBIE78	DBIE79	DBIE7A	DBIE7B	DBIE7C	DBIE7D	DBIE7E	DBIE7F

Bit	Function
DBIExx	Data Buffer Segment (16-byte segments) Interrupt Enable: 0 – Interrupt for segment disabled 1 – Interrupt for segment enabled An interrupt will occur when the segment's receive flag is active. To enable Data Buffer interrupts the IEDBUF bit in the Interrupt Enable Register has to be set.

3.11.25. Data Buffer Checksum Flag Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x8FA0	DBCS00	DBCS01	DBCS02	DBCS03	DBCS04	DBCS05	DBCS06	DBCS07

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x8FA1	DBCS08	DBCS09	DBCS0A	DBCS0B	DBCS0C	DBCS0D	DBCS0E	DBCS0F

...

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x8FAE	DBCS70	DBCS71	DBCS72	DBCS73	DBCS74	DBCS75	DBCS76	DBCS77

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x8FAF	DBCS78	DBCS79	DBCS7A	DBCS7B	DBCS7C	DBCS7D	DBCS7E	DBCS7F

Bit	Function
DBCSxx	Data Buffer Segment (16-byte segments) Checksum Flag: 0 – Checksum OK 1 – Checksum error This flag is cleared by writing a '1' into the segment's DBRXxx bit in the DataBufRxFlag register.

3.11.26. Data Buffer Overflow Flag Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x8FC0	DBOV00	DBOV01	DBOV02	DBOV03	DBOV04	DBOV05	DBOV06	DBOV07

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x8FC1	DBOV08	DBOV09	DBOV0A	DBOV0B	DBOV0C	DBOV0D	DBOV0E	DBOV0F

...

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x8FCE	DBOV70	DBOV71	DBOV72	DBOV73	DBOV74	DBOV75	DBOV76	DBOV77

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x8FCF	DBOV78	DBOV79	DBOV7A	DBOV7B	DBOV7C	DBOV7D	DBOV7E	DBOV7F

Bit	Function
DBOVxx	Data Buffer Segment (16-byte segments) Overflow Flag: 0 – No overflow condition 1 – Overflow: a new packet has been received before the DBRX flag for this segment was cleared This flag is cleared by writing a '1' into the segment's DBRXxx bit in the DataBufRxFlag register.

3.11.27. Data Buffer Receive Flag Register

address	bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
0x8FE0	DBRX00	DBRX01	DBRX02	DBRX03	DBRX04	DBRX05	DBRX06	DBRX07

address	bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
0x8FE1	DBRX08	DBRX09	DBRX0A	DBRX0B	DBRX0C	DBRX0D	DBRX0E	DBRX0F

. . .

address	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
0x8FEE	DBRX70	DBRX71	DBRX72	DBRX73	DBRX74	DBRX75	DBRX76	DBRX77

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0x8FEF	DBRX78	DBRX79	DBRX7A	DBRX7B	DBRX7C	DBRX7D	DBRX7E	DBRX7F

Bit	Function
DBRXxx	Data Buffer Segment (16-byte segments) Receive Flag: 0 – No packet received 1 – Data packet received in this segment This flag is cleared by writing a '1' into the segment's DBRXxx bit.

3.11.28. SFP Module EEPROM and Diagnostics

Small Form Factor Pluggable (SFP) transceiver modules provide a means to identify the module by accessing an EEPROM. As an advanced feature some modules also support reading dynamic information including module temperature, receive and transmit power levels etc. from the module. The EVR gives access to all of this information through a memory window of 2×256 bytes. The first 256 bytes consist of the EEPROM values and the rest of the advanced values.

Byte #	Field size	Notes	Value
		BASE ID FIELDS	
0	1	Type of serial transceiver	0x03 = SFP transceiver
1	1	Extended identifier of type serial transceiver	0x04 = serial ID module definition
2	1	Code for connector type	0x07 = LC

Byte #	Field size	Notes	Value
3 – 10	8	Code for electronic compatibility or optical compatibility	
11	1	Code for serial encoding algorithm	
12	1	Nominal bit rate, units of 100 MBits/sec	
13	1	Reserved	
14	1	Link length supported for 9/125 μm fiber, units of km	
15	1	Link length supported for 9/125 μm fiber, units of 100 m	
16	1	Link length supported for 50/125 μm fiber, units of 10 m	
17	1	Link length supported for 62.5/125 μm fiber, units of 10 m	
18	1	Link length supported for copper, units of meters	
19	1	Reserved	
20 – 35	16	SFP transceiver vendor name (ASCII)	
36	1	Reserved	
37 – 39	3	SFP transceiver vendor IEEE company ID	
40 – 55	16	Part number provided by SFP transceiver vendor (ASCII)	
56 – 59	4	Revision level for part number provided by vendor (ASCII)	
60 – 62	3	Reserved	
63	1	Check code for Base ID Fields	
		EXTENDED ID FIELDS	
64 – 65	2	Indicated which optional SFP signals are implemented	
66	1	Upper bit rate margin, units of %	
67	1	Lower bit rate margin, units of %	
68 – 83	16	Serial number provided by vendor (ASCII)	
84 – 91	8	Vendor's manufacturing date code	
92 – 94	3	Reserved	
95	1	Check code for the Extended ID Fields	
		VENDOR SPECIFIC ID FIELDS	
96 – 127	32	Vendor specific data	
128 – 255		Reserved	
		ENHANCED FEATURE SET MEMORY	
256 – 257	2	Temp H Alarm	Signed twos complement integer in increments of 1/256 ï¿½C

Byte #	Field size	Notes	Value
258 – 259	2	Temp L Alarm	Signed twos complement integer in increments of 1/256 ï¿½C
260 – 261	2	Temp H Warning	Signed twos complement integer in increments of 1/256 ï¿½C
262 – 263	2	Temp L Warning	Signed twos complement integer in increments of 1/256 ï¿½C
264 – 265	2	VCC H Alarm	Supply voltage decoded as unsigned integer in increments of 100 µV
266 – 267	2	VCC L Alarm	Supply voltage decoded as unsigned integer in increments of 100 µV
268 – 269	2	VCC H Warning	Supply voltage decoded as unsigned integer in increments of 100 µV
270 – 271	2	VCC L Warning	Supply voltage decoded as unsigned integer in increments of 100 µV
272 – 273	2	Tx Bias H Alarm	Laser bias current decoded as unsigned integer in increment of 2 µA
274 – 275	2	Tx Bias L Alarm	Laser bias current decoded as unsigned integer in increment of 2 µA
276 – 277	2	Tx Bias H Warning	Laser bias current decoded as unsigned integer in increment of 2 µA
278 – 279	2	Tx Bias L Warning	Laser bias current decoded as unsigned integer in increment of 2 µA
280 – 281	2	Tx Power H Alarm	Transmitter average optical power decoded as unsigned integer in increments of 0.1 µW
282 – 283	2	Tx Power L Alarm	Transmitter average optical power decoded as unsigned integer in increments of 0.1 µW
284 – 285	2	Tx Power H Warning	Transmitter average optical power decoded as unsigned integer in increments of 0.1 µW
286 – 287	2	Tx Power L Warning	Transmitter average optical power decoded as unsigned integer in increments of 0.1 µW
288 – 289	2	Rx Power H Alarm	Receiver average optical power decoded as unsigned integer in increments of 0.1 µW
290 – 291	2	Rx Power L Alarm	Receiver average optical power decoded as unsigned integer in increments of 0.1 µW
292 – 293	2	Rx Power H Warning	Receiver average optical power decoded as unsigned integer in increments of 0.1 µW
294 – 295	2	Rx Power L Warning	Receiver average optical power decoded as unsigned integer in increments of 0.1 µW

Byte #	Field size	Notes	Value
296 – 311	16	Reserved	
312 – 350		External Calibration Constants	
351	1	Checksum for Bytes 256 – 350	
352 – 353	2	Real Time Temperature	Signed twos complement integer in increments of 1/256 ï¿½C
354 – 355	2	Real Time VCC Power SupplyVoltage	Supply voltage decoded as unsigned integer in increments of 100 µV
356 – 357	2	Real Time Tx Bias Current	Laser bias current decoded as unsigned integer in increment of 2 µA
358 – 359	2	Real Time Tx Power	Transmitter average optical power decoded as unsigned integer in increments of 0.1 µW
360 – 361	2	Real Time Rx Power	Receiver average optical power decoded as unsigned integer in increments of 0.1 µW
362 – 365	4	Reserved	
366	1	Status/Control	bit 7: TX_DISABLE State bit 6 – 3: Reserved bit 2: TX_FAULT State bit 1: RX_LOS State bit 0: Data Ready (Bar)
367	1	Reserved	
368	1	Alarm Flags	bit 7: Temp High Alarm bit 6: Temp Low Alarm bit 5: VCC High Alarm bit 4: VCC Low Alarm bit 3: Tx Bias High Alarm bit 2: Tx Bias Low Alarm bit 1: Tx Power High Alarm bit 0: Tx Power Low Alarm
369	1	Alarm Flags cont.	bit 7: Rx Power High Alarm bit 6: Rx Power Low Alarm bit 5 – 0: Reserved
370 – 371	2	Reserved	
372	1	Warning Flags	bit 7: Temp High Warning bit 6: Temp Low Warning bit 5: VCC High Warning bit 4: VCC Low Warning bit 3: Tx Bias High Warning bit 2: Tx Bias Low Warning bit 1: Tx Power High Warning bit 0: Tx Power Low Warning

Byte #	Field size	Notes	Value
373	1	Warning Flags cont.	bit 7: Rx Power High Warning bit 6: Rx Power Low Warning bit 5 – 0: Reserved
374 – 511		Reserved/Vendor Specific	

3.12. Firmware Version Change Log

FW Version	Date	Changes	Affected HW
0200	11.06.2015	- Prototype release	VME-EVR-300
0201	24.09.2015	- Added segmented data buffer block status flags - Changed delay compensation FIFO depth from 2k to 4k event cycles - Added DCM modulation to improve jitter performance	VME-EVR-300
0203	12.01.2016	- Delay compensation amendments, non-GTX outputs are compensated properly	VME-EVR-300
0204	25.01.2016	- First release for PCIe-EVR-300DC - Fixed segmented data buffer flag writes	all
0204	03.02.2016	- Fixed initial values of GTX outputs - GTX output aligment	VME-EVR-300
0205	07.04.2016	- Changed PCIe-EVR-300DC class code to 0x118000. - Moved delay compensation data from first segment to last segment. - Fixed dual output mapping for transition board outputs. - Added backplane signals to mTCA-EVR. - Added delay compensation disabled mode to be able to use DC capable EVRs with pre-DC EVG and fan-outs.	all
0206	12.08.2016	- Relocated segmented data buffer to new address location. - Replaced earlier data buffer in its original position (maintaining compatibility with 230 series protocol). - Changed segmented data buffer protocol to use K28.2 as a start symbol	all
0207	30.08.2016	- Added stand-alone capability: using its internal reference the EVR can now operate as a stand-alone pulse generator without event link. - EVR can operate as a simple EVG by forwarding internal events - Added software event capability - Added one EVG type sequencer	all

FW Version	Date	Changes	Affected HW
030207	23.12.2016	- Changed beacon event code from 0x7a to 0x7e. - Added status bits for delay compensation path delay value validity. - Added register for topology ID.	all
040207	09.1.2017	- Repaired "trigger allways" problem with triggering sequencer with pulse generator 19. - Added mapping 61 for sequencer software triggering.	all
050207	19.1.2017	- Fixed running on internal reference for VME-EVR-300.	VME-EVR-300
060207	9.2.2017	- Added configurability to handling a lost event clock: continue, stop, fallback to reference clock. - Further fix to running on internal reference for VME-EVR-300.	VME-EVR-300
070207	6.4.2017	- Fixed CML/GTX operation in stand-alone mode without receiver event stream. - Fixed mapping of TCLKA/TCLKB backplane clocks on mTCA-EVR-300.	mTCA-EVR-300
080207	7.8.2017	- PCIe AXI to OPB bridge fix for overloapping read/write operation during block transfers. - Added pullup to MODU_SDA and MODU_DEF0.	PCIe-EVR-300DC
090207	27.2.2018	- Changes to get design built on Vivado 2017.4	All
0A0207	18.9.2018	- Changed number of external inputs to 16.	PCIe-EVR-300DC
0D0207	20.5.2019	- Added programmable phase shift to prescalers.	mTCA-EVR-300
0E0207	2.7.2019	- Fix to event FIFO. - Added flip-flop outputs.	mTCA-EVR-300
0F0207	3.3.2020	- Added support for RTM on mTCA-EVR-300.	mTCA-EVR-300
100207	9.4.2021	- Allow disabling receive databuf mode.	mTCA-EVR-300
110207	2.6.2021	- Added support for mTCA-EVR-300RF	mTCA-EVR-300RF
130207	14.2.2022	- 32 bit pulse width for HW mask pulse generators	VME-EVR-300
140207	11.5.2022	- Added pulse generator masking capability for HW set and reset	mTCA-EVR-300DC
150207	15.6.2022	- Added VME-UTB-64x support for VME-EVR-300	VME-EVR-300
160207	6.10.2022	- Changed pulse with and delay to 32 bits on all pulse generators.	mTCA-EVR-300DC and VME-EVR-300
170207	6.10.2022	- Use LOS from SFP transceiver to reset receiver.	mTCA-EVR-300DC
180207	17.01.2023	- Added support for new mTCA-EVR-300DC hardware with improved backplane triggers	mTCA-EVR-300DC
190207	27.05.2023	- Added support pulse trains for pulse generators	mTCA-EVR-300DC
200207	26.06.2023	- Fixed pulse trains issues with single width pulses	mTCA-EVR-300DC

4. Examples

4.1. Setting Up an Event System with Delay Compensation

In this example we are setting up a test system consisting of two VME-EVM-300 boards and two VME-EVR-300 boards. The first EVM (EVM1) is configured as the master and the seconds EVM (EVM2) as a fan-out. One EVR (EVR1) will be connected to the master (EVM1) and the other EVR (EVR2) to the fan-out (EVM2). The example setup is represented in figure 27.

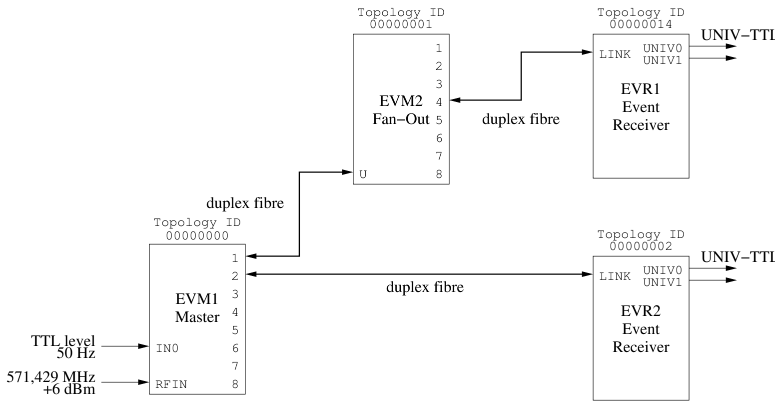

Figure 27: Example Setup

4.1.1. Initializing Master EVG

First we need to configure the master EVG to use the external RF input reference clock divided by four. After changing the clock source we need to reload the fractional synthesizer control word to force an internal reset.

The next step is to tell the top EVG that it is the master EVG and enable its beacon generator and delay compensation master responsibilities. Please note that the highest EVG in the system has to be the system master and if delay compensation is used it also has to have the beacon generator enabled. Only one EVG/EVM can be the system master and only one EVG is allowed to have the beacon generator enabled.

API calls	**Register access**
EvgSetRFInput(evm1, 1, 3);	*(evm1+0x50) = 0xc1030000;
EvgSetFracDiv(evm1, 0x0891c100);	*(evm1+0x80) = 0x0891c100;
EvgSystemMasterEnable(evm1, 1);	*(evm1+0x04) = 0xe0c00000;
EvgBeaconEnable(evm1, 1);	
EvgEnable(evm1, 1);	

4.1.2. Initializing VME-EVM-300 as Fan-Out

The downstream EVM has to be configured to use the upstream EVG/EVM clock as its event clock and after this we need to (re)load the fractional synthesizer control word.

We enable the EVM and please note that both the system master bit and beacon generator bit are disabled.

API calls	Register access
EvgSetRFInput(evm2, 4, 0x0c);	*(evm2+0x50) = 0xc40c0000;
EvgSetFracDiv(evm2, 0x0891c100);	*(evm2+0x80) = 0x0891c100;
EvgEnable(evm2, 1);	*(evm2+0x04) = 0xe0000000;

4.1.3. Initializing VME-EVR-300

We start with setting the fractional synthesizer operating frequency (reference for event clock) so that the EVR can lock to the received event stream.

The delay compensation logic measures/calculates a path delay from the master EVM/EVG to the EVR which consist of internal delays and fibre delays. The EVR has a receive FIFO and it adjusts the delay of this FIFO based on a target delay value and the actual path delay value. The delay value is a 32 bit value with a 16 bit integer part and a 16 bit fractional part. The integer part represents the delay in event clock cycles i.e. a value of 0x00010000 corresponds to an actual delay of one event clock cycle which at this examples rate is 7 ns.

In this example we set the target delay to 0x02100000 which is 3.696 μs.

API calls	Register access
EvrSetFracDiv(evr1, 0x0891c100);	*(evr1+0x80) = 0x0891c100;
EvrSetTargetDelay(evr1, 0x02100000);	*(evr1+0xb0) = 0x02100000;
EvrGetViolation(evr1, 1);	*(evr1+0x08) = 0x00000001;
EvrDCEnable(evr1, 1);	*(evr1+0x04) = 0x80400000;
EvrEnable(evr1, 1);	

The datapath delay value can be read from the EVR DCRxValue register at offset 0x0b8. For the example above with 2 m fiber patches the measured datapath delay value shows 0x0032cff0 (355.686 ns) for EVR1 and 0x00125eea (128.595 ns) for EVR2.

4.1.4. Generating an Event from AC input

A 50 Hz TTL level square wave signal is provided to the IN0 input on EVM1. We setup the input AC input divider to divide by 5, set the AC input logic to trigger event trigger 0 and we configure event trigger 0 to send out event code 0x01.

API calls	Register access
EvgSetACInput(evm1, 0, 0, 5, 0);	*(evm1+0x10) = 0x00000500;
EvgSetACMap(evm1, 0);	*(evm1+0x14) = 0x00000001;
EvgSetTriggerEvent(evm1, 0, 0x01, 1);	*(evm1+0x100) = 0x00000101;

4.1.5. Receiving an Event and Generating an Output Pulse

To generate a pulse on a received event code in the EVR we need to setup the mapping RAM to trigger a pulse generator on an event and setup the pulse generator. We also need to map the pulse generator to the actual hardware output.

API calls	Register access
EvrSetPulseMap(evr1, 0, 0x01, 0, -1, -1);	*(evr1+0x4014) = 0x00000001;
EvrSetPulseParams(evr1, 0, 0, 0, 1000);	*(evr1+0x20C) = 0x000003e8;
EvrSetPulseProperties(evr1, 0, 0, 0, 0, 1, 1);	*(evr1+0x200) = 0x00000003;
EvrSetUnivOutMap(evr1, 0, 0x3f00);	*(evr1+0x440) = 0x3f003f3f;
EvrMapRamEnable(evr1, 0, 1);	*(evr1+0x04) = 0x88400200;
EvrOutputEnable(evr1, 1);	

Now we should see a 7 μs pulse on EVR1 UNIV0 output with a rate of 10 Hz. If we configure EVR2 the same way as the EVR1 in this example we should see a similar pulse on its UNIV0 output aligned with the output of EVR1.

4.2. Event Receiver Standalone Operation

Starting from firmware version 0207 capability to use the EVR as a stand-alone unit has been added. Functionality includes:

- Using the internal fractional synthesizer clock as a reference clock
- Generating internal events by software
- Generating internal event by one EVG type sequencer
- Generating internal event by external signals
- Internal events may be sent out on the TX link by setting the FWD bit for each event in the active mapping RAM

The example code below has been written for the mTCA-EVR-300, but with minor changes (remapping the outputs) it can be used for other form factors as well.

```
int evr_sa(volatile struct MrfErRegs *pEr)
{
  int i;

  EvrEnable(pEr, 1);
  if (!EvrGetEnable(pEr))
    {
      printf(ERROR_TEXT "Could not enable EVR!\n");
      return -1;
    }

  EvrSetIntClkMode(pEr, 1);

  /* Build configuration for EVR map RAMS */

  {
    int ram,code;

    /* Setup MAP ram:
       event code 0x01 to 0x04 trigger pulse generators 0 through 3 */
    ram = 0;
    for (i = 0; i < 4; i++)
      {
        code = 1+i;
        EvrSetLedEvent(pEr, ram, code, 1);
        /* Pulse Triggers start at code 1 */
        EvrSetPulseMap(pEr, ram, code, i, -1, -1);
      }

    /* Setup pulse generators and front panel TTL outputs */
    for (i = 0; i < 4; i++)
      {
 EvrSetPulseParams(pEr, i, 1, 100, 100);
 EvrSetPulseProperties(pEr, i, 0, 0, 0, 1, 1);
 EvrSetFPOutMap(pEr, i, 0x3f00 | i);
      }
```

```
  }

  /* Setup Prescaler 0 */
  EvrSetPrescaler(pEr, 0, 0x07ffff);

  /* Write some RAM events */
  EvrSetSeqRamEvent(pEr, 0, 0, 0, 1);
  EvrSetSeqRamEvent(pEr, 0, 1, 0x001ff, 2);
  EvrSetSeqRamEvent(pEr, 0, 2, 0x002ff, 3);
  EvrSetSeqRamEvent(pEr, 0, 3, 0x003ff, 4);
  EvrSetSeqRamEvent(pEr, 0, 4, 0x04000, 0x7f);
  /* Setup sequence RAM to trigger from prescaler 0 */
  EvrSeqRamControl(pEr, 0, 1, 0, 0, 0, C_EVR_SIGNAL_MAP_PRESC+0);

  EvrMapRamEnable(pEr, 0, 1);

  EvrOutputEnable(pEr, 1);

  return 0;
}
```

4.3. mTCA-EVR-300RF GUN-TX/GUN-RC-300 Pulse Mode Example

This example is using the shell wrapper functions in the mrf-linux-api available on `https://github.com/jpietari/mrf-linux-api`.

```
./EvrSetEventFrequency /dev/era3 124.95
./EvrGetViolation /dev/era3 1
./EvrEnable /dev/era3 1
./EvrMapRamEnable /dev/era3 0 1
./EvrGetViolation /dev/era3 1
# Setup Map RAM0 to trigger pulse generator 0 on event 2
./EvrSetPulseMap /dev/era3 0 2 0 -1 -1
# Setup 10 event clock cycle long pulse on pulse generator 0
./EvrSetPulseParams /dev/era3 0 0 0 10
./EvrSetPulseProperties /dev/era3 0 0 0 0 1 1
# Map Pulse genrator 0 to UNIV0
./EvrSetUnivOutMap /dev/era3 1 0x3f00
# Map Pulse genrator 0 also to SFP GTX
# SFP GTX is on Universal Output 20
./EvrSetUnivOutMap /dev/era3 20 0x3f00
# Map Pulse genrator 0 also to CML GTX
# CML GTX is on Universal Output 21
./EvrSetUnivOutMap /dev/era3 21 0x3f00

# Override inhibit signal on UNIVIN0
./EvrSetGunTxInhibitOverride /dev/era3 1
# Set SFP GTX into GUN-RC-300 mode
./EvrSetCMLMode /dev/era3 4 0x3800
# Enable SFP GTX output
./EvrCMLEnable /dev/era3 4 1
# For testing purposes it is possible to look at the GTX modulation
# on the CML outputs:
```

```
#./EvrSetCMLMode /dev/era3 5 0x3800
./EvrCMLEnable /dev/era3 5 1
# Set Phase offset 0-2559, steps of 8 ns/2560 = 3.125 ps
./EvrSetCMLPhaseOffset /dev/era3 4 1000}
```