**Micro-Research Finland Oy**
Välitalontie 83 C, FI-00660 Helsinki, Finland



# Event Generator (EVG-200)

# Technical Reference

# Firmware Version E306

## Contents

# Introduction

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.

## *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.

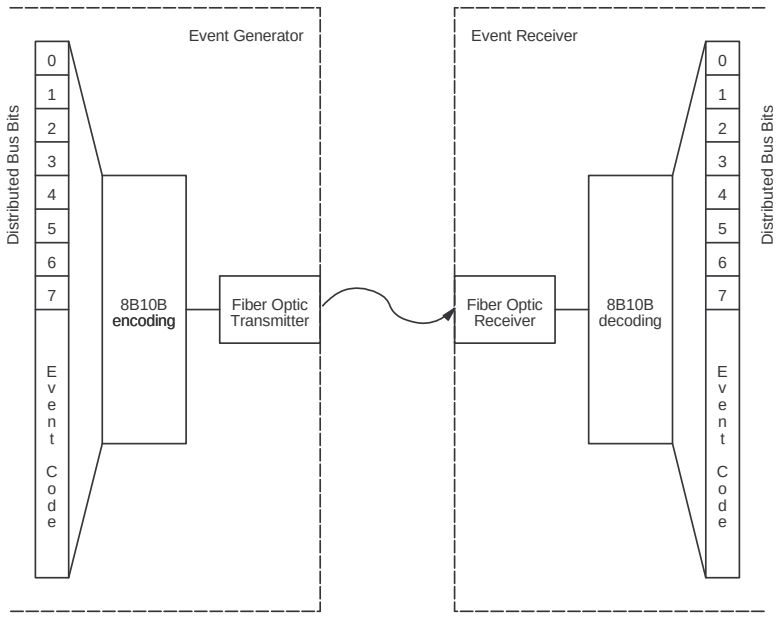


**Figure 1: Event Frame**

## 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 is the serial bit stream.

| 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 |
| 0x71 | Seconds ‘1’ | - | Shift in ‘1’ to LSB of Seconds Shift Register |
| 0x72 – 0x79 | - | User Defined | User Defined |
| 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 |
| 0x7F | End of Sequence | Stop Sequence | - |

### Distributed Bus and Data Transmission

The distributed bus allows transmission of eight simultaneous signals with the event clock rate time resolution (8 ns at 125 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.

In latest firmware versions the distributed bus bandwidth may be shared by transmission of a configurable size data buffer to up to 2 kbytes. When data transmission is enabled the distributed bus bandwidth is halved. The remaining bandwidth is reserved for transmitting data with a speed up to 62.5 Mbytes/s (event clock rate divided by two).

## *Event Sources*

### 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.

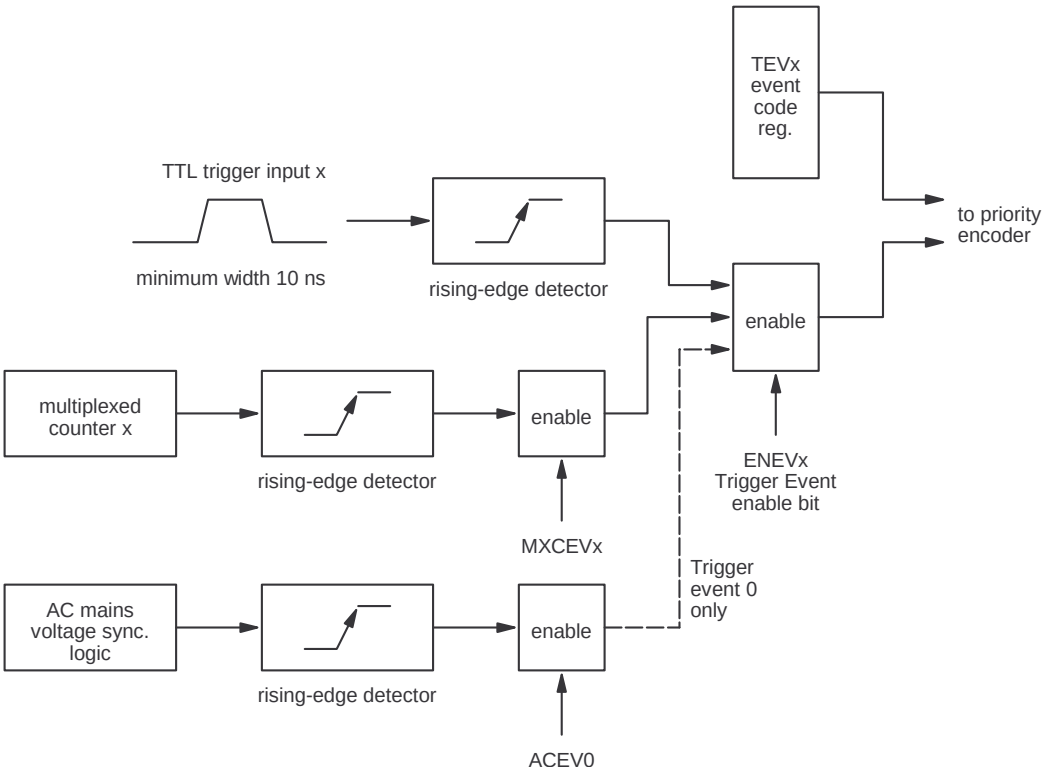


**Figure 2: Trigger Events**

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

## 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.

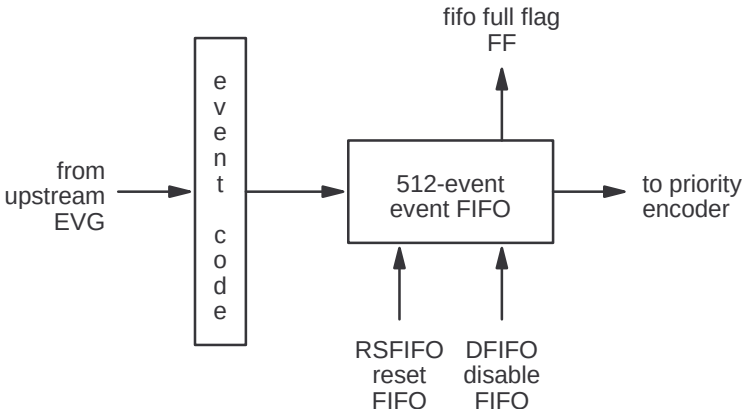


**Figure 3: Upstream Event FIFO**

## 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. The 8-bit event 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.

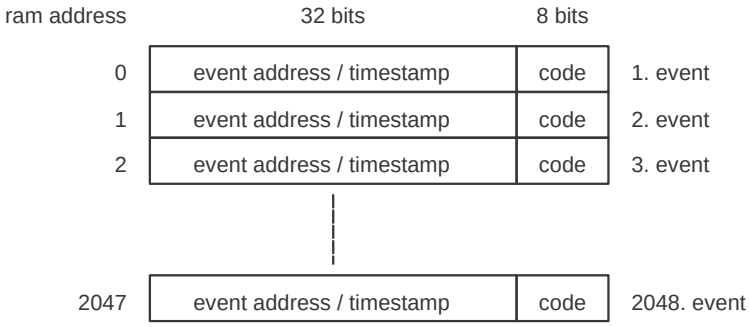


**Figure 4: Sequencer RAM Structure**

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 RAMs are addressed indirectly: there are separate registers for address and data.

The sequencer clock may be selected to be applied externally or down converted directly from the event clock with a 16-bit prescaler. The clock is selected by a 16-bit prescaler. When the prescaler value is zero an externally supplied clock is used. The sequencers are capable of operating at full event clock speed to up to 125 MHz.

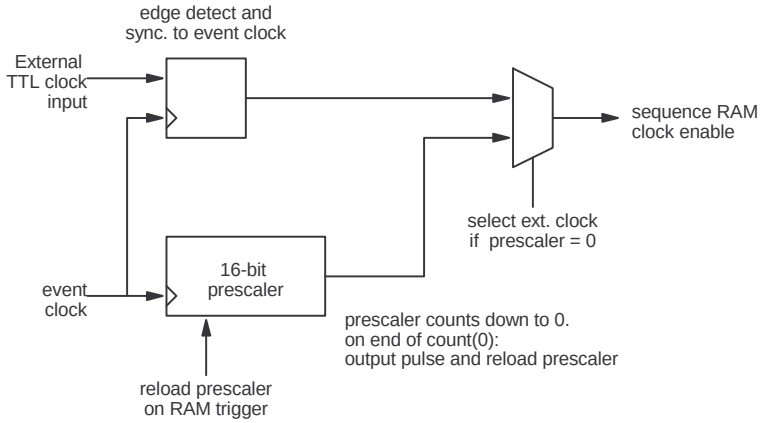


**Figure 5: Sequencer Clock**

Both sequencers have their own prescalers, but by setting the CMODE configuration bit sequence RAM 2 may be forced to use the same clock, trigger and reset signals as sequence RAM 1.

The Sequencers may be triggered from several sources including hardware triggering from external TTL input, software triggering by VME access, triggering on a multiplexed counter output or AC mains voltage synchronization logic output.

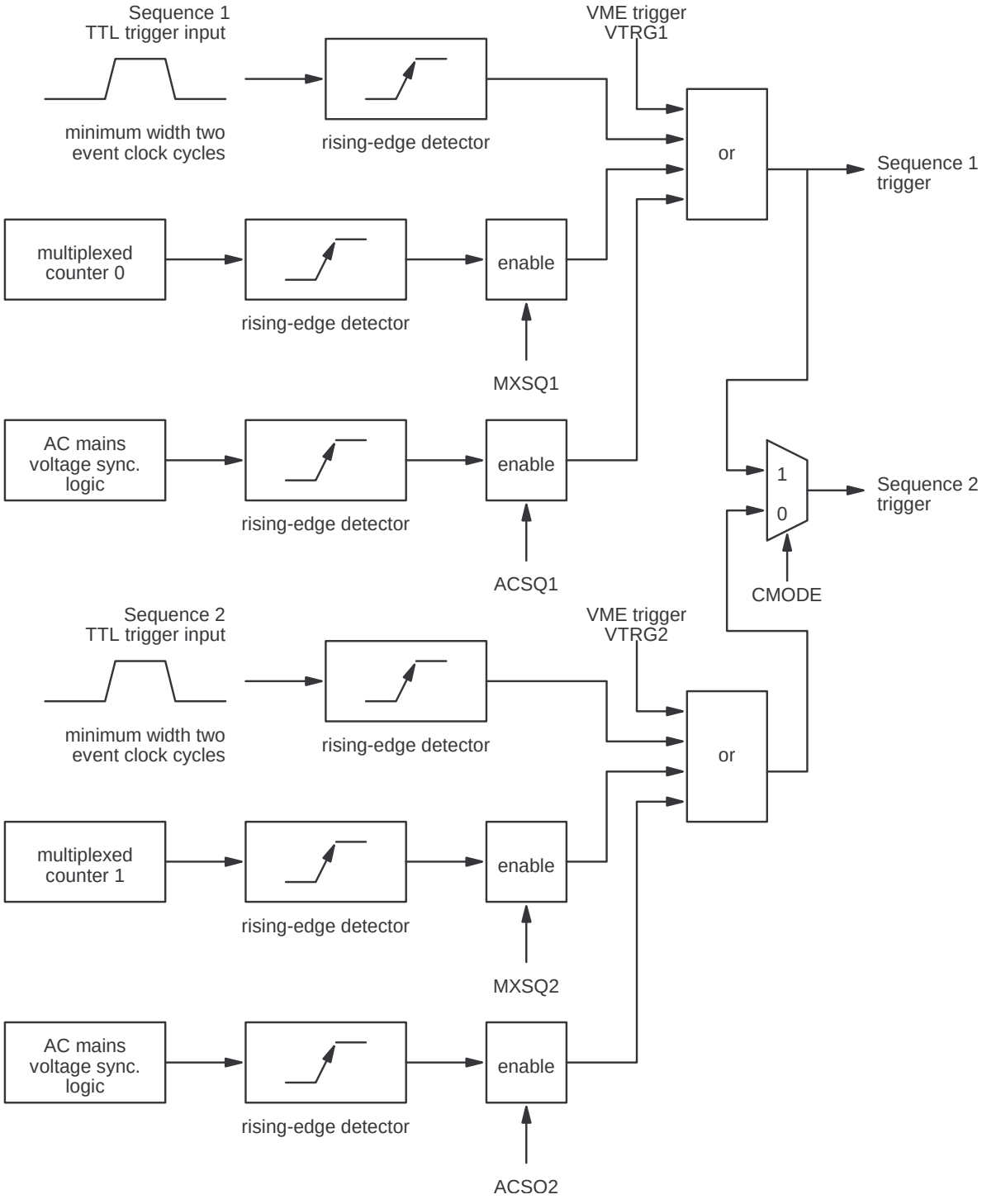


**Figure 6: Sequencer Triggering**

The sequencers are enabled by setting bit ENSQx in the Event Enable Register. The RAMs may be disabled any time by clearing the corresponding bits. Disabling sequence RAMs does not reset the RAM address and timestamp registers. By setting the bit SEQx 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 counters starts counting. The counter value is compared to the event address of the next event in the RAM table. When the counter value matches the timestamp in the RAM table, the attached event code is transmitted. The time offset 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. The end sequence code resets the sequencer RAM table address and timestamp register and depending on configuration bits, disables the sequencer (single sequence, SSEQx=1) or restarts the sequence either immediately (recycle sequence, RCYLx=1) or waits for a new trigger (RCYLx=0).

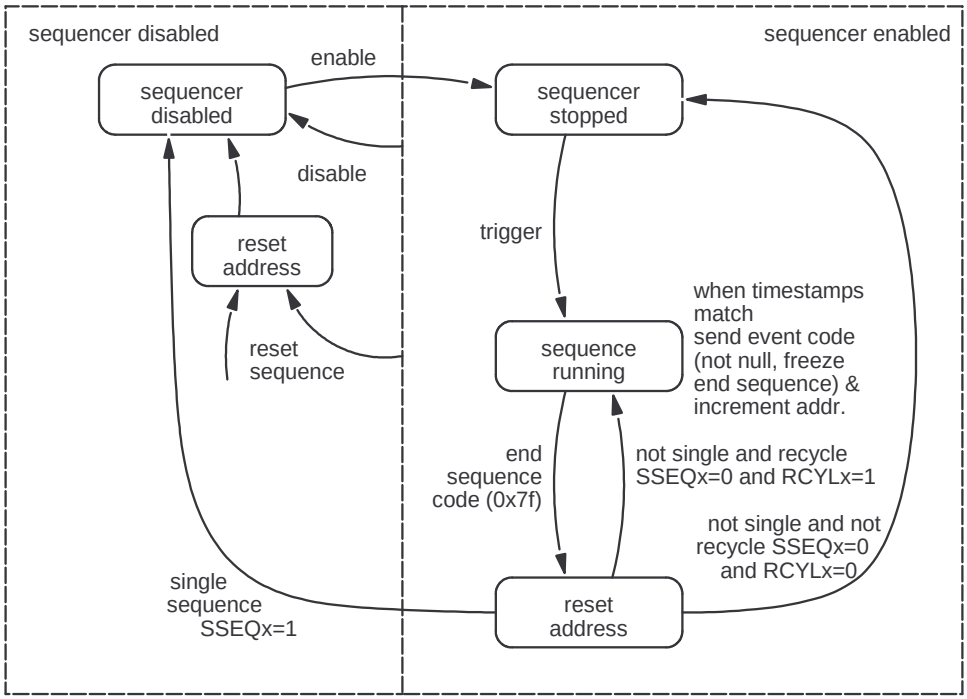


**Figure 7: Sequencer Control**

## *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.

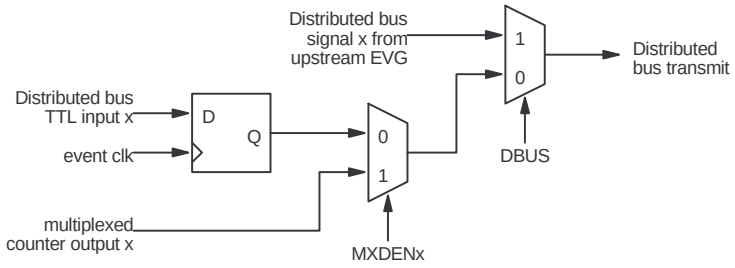


**Figure 8: Distributed Bus**

## Timestamping Inputs

Staring 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.

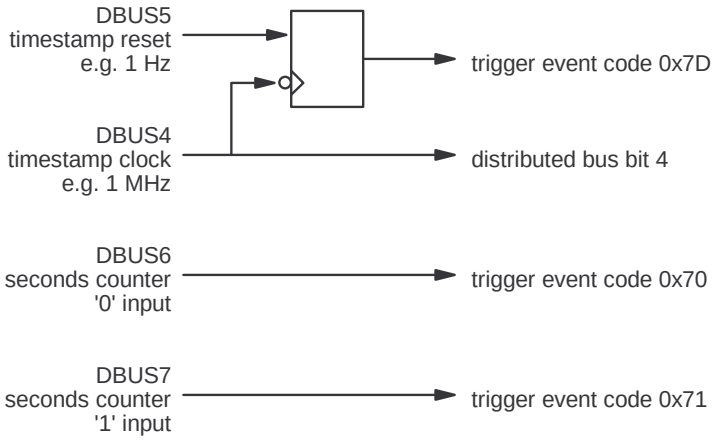


**Figure 9: Timestamping Inputs**

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 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.

## *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.

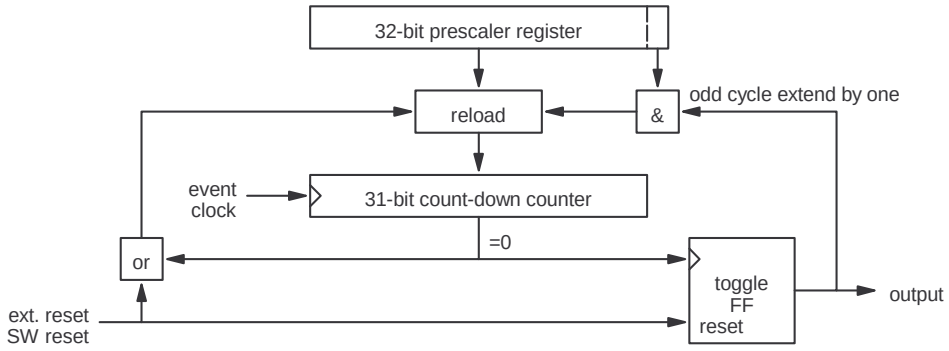


**Figure 10: Multiplexed Counter**

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.

| Prescaler value | Duty Cycle | Frequency at 125 MHz Event Clock |
|---|---|---|
| 0, 1 not allowed | undefined | undefined |
| 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.

## *Configurable Size Data Buffer*

Starting from firmware version E305 transmission of a configurable size data buffer over the event system link is possible. The buffer size can be programmed in four byte increments (long words) from 4 bytes to 2048 bytes.

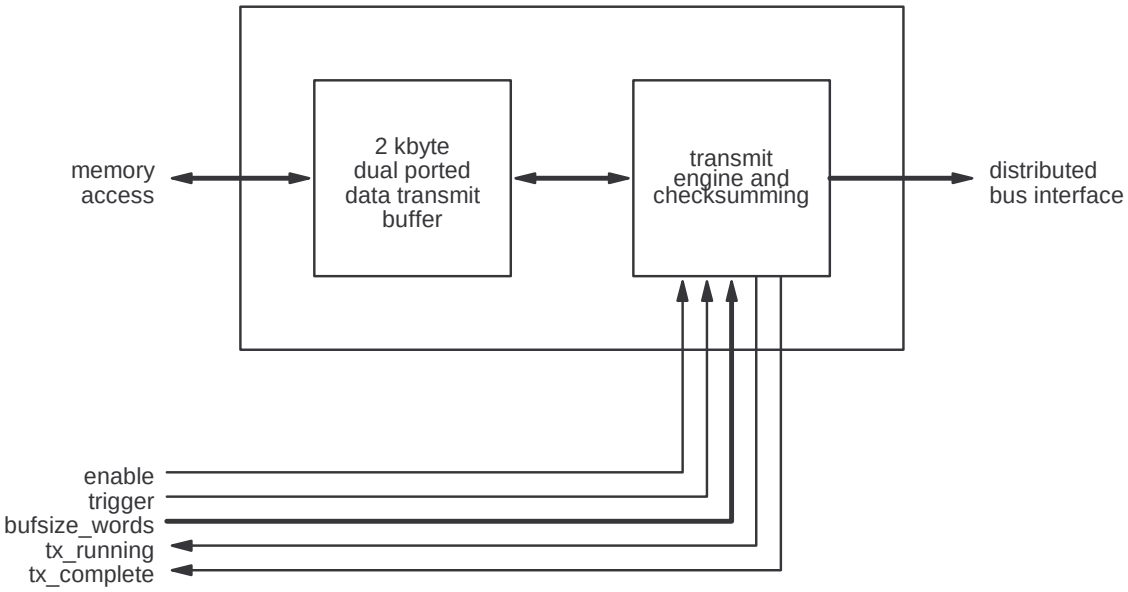


**Figure 11: Configurable size transmit data buffer**

When the EVG is configured for data transmission (*mode* = 1 in data buffer control register) the bandwidth of the distributed bus is shared with data transmission: half of the bandwidth remains for the distributed bus and the other half is reserved for data transmission.

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 trigger 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 1: Data Transmission Protocol**

| **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 (LSB) |
| Dxx.x | Checksum(MSB) |

## AC Line Synchronisation

The Event Generator provides synchronization to the mains voltage frequency. The mains voltage frequency is 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 synchronized to the event clock or the output of multiplexed counter 7.

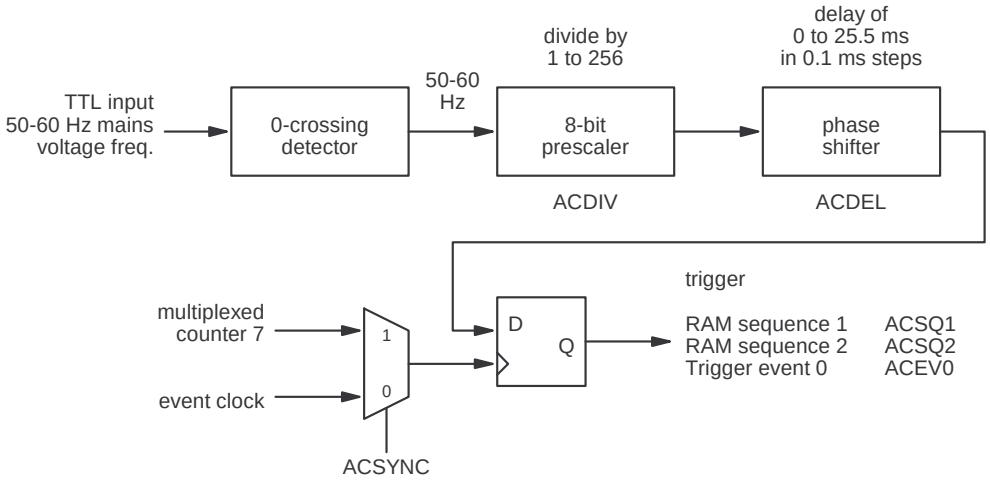


**Figure 12: AC Input**

## Event Clock

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.

| | Event Clock | Bit Rate |
|---|---|---|
| Minimum | 50 MHz | 1.0 Gb/s |
| Maximum | 125 MHz | 2.5 Gb/s |

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

## 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 around +10 dBm can be used.

| Divider | RF Input Frequency | Event Clock | Bit Rate |
|---|---|---|---|
| ÷ 4 | 200 MHz – 500 MHz | 50 MHz – 125 MHz | 1.0 Gb/s – 2.5 Gb/s |
| ÷ 5 | 250 MHz – 625 MHz | 50 MHz – 125 MHz | 1.0 Gb/s – 2.5 Gb/s |
| ÷ 6 | 300 MHz – 900 MHz | 50 MHz – 125 MHz | 1.0 Gb/s – 2.5 Gb/s |
| ÷ 8 | 400 MHz – 1 GHz | 50 MHz – 125 MHz | 1.0 Gb/s – 2.5 Gb/s |
| ÷ 10 | 500 MHz – 1.25 GHz | 50 MHz – 125 MHz | 1.0 Gb/s – 2.5 Gb/s |

| | | | |
|---|---|---|---|
| ÷ 12 | 600 MHz – 1.3 GHz *) | 50 MHz – 125 MHz | 1.0 Gb/s – 2.5 Gb/s |

*) Range limited by MC100EP139 maximum frequency for ÷ 6

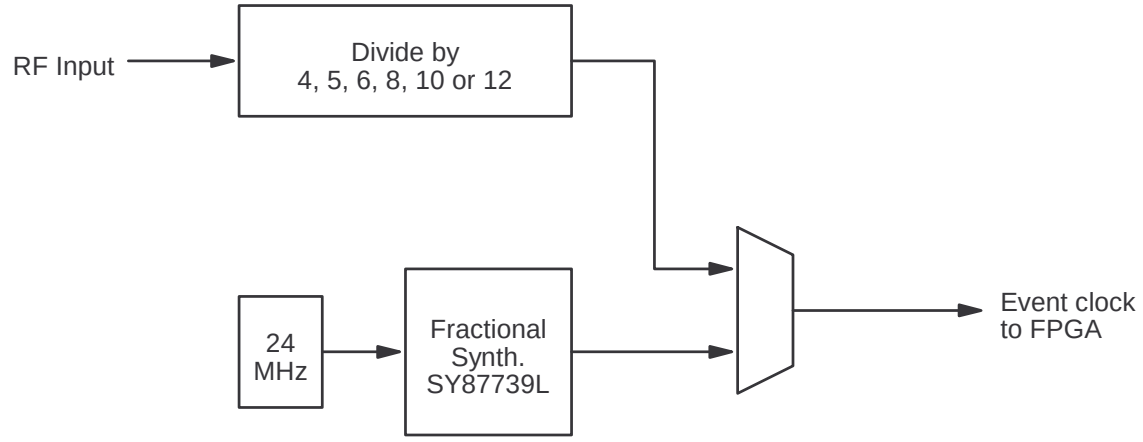


**Figure 13: Event Clock Generation**

## Fractional Synthesiser

For laboratory testing purposes the event clock may be generated on-board the event generator 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/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 |

## Non-Volatile Storage for Frequency Configuration

The reference clock setting and delay line initialisation values are stored in non-volatile memory inside the IP2022 microcontroller. The part to part differences of the delay lines require tuning of the delays when a new operating frequency is selected.

# Connections

## *Front Panel Connections*

The front panel of the Event Generator is shown in Figure 14.

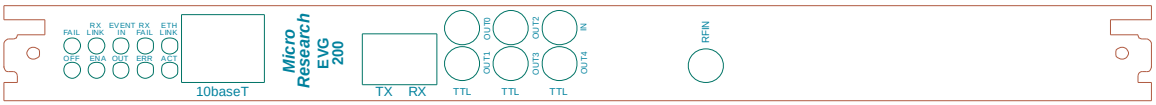


**Figure 14: Event Generator Front Panel**

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

| Connector / Led | Style | Level | Description |
|---|---|---|---|
| FAIL | Red Led | | Module Failure |
| OFF | Blue Led | | Module Powered Down |
| RX LINK | Green Led | | Receiver Link Signal OK |
| ENA | Green Led | | Event Generator Enabled |
| EVENT IN | Yellow Led | | Incoming Event (RX) |
| EVENT OUT | Yellow Led | | Outgoing Event (TX) |
| RX FAIL | Red Led | | Receiver Violation |
| ERR | Red Led | | SY87739L reference not locked |
| ETH LINK | Green Led | | 10baseT Activity Led |
| ACT | Yellow Led | | Ubicom IP2022 Active (Flashing) |
| 10baseT | RJ45 | 10baseT | 10baseT Ethernet Connection |
| TX | LC | optical | Transmit Optical Output (TX) |
| RX | LC | optical | Receiver Optical Input (RX) |
| OUT0 | LEMO-EPY | TTL | Multiplexed Counter 0 Output |
| OUT1 | LEMO-EPY | TTL | Multiplexed Counter 1 Output |
| OUT2 | LEMO-EPY | TTL | Multiplexed Counter 7 Output |
| OUT3 | LEMO-EPY | TTL | Injection Trigger |
| OUT4 | LEMO-EPY | TTL | Multiplexed Counter 2 Output |
| IN | LEMO-EPY | TTL | 50-60 Hz Line Frequency |
| RFIN | LEMO-EPL | RF +10 dBm | RF Reference Input |

## *VME P2 User I/O Pin Configuration*

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

| 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 | external event trigger input 0 |
| C2 | external event trigger input 1 |
| C3 | external event trigger input 2 |
| C4 | external event trigger input 3 |
| C5 | external event trigger input 4 |
| C6 | external event trigger input 5 |
| C7 | external event trigger input 6 |
| C8 | external event trigger input 7 |
| C9 | sequence RAM 1 clock |
| C10 | sequence RAM 1 trigger |
| C11 | sequence RAM 1 reset |
| C12 | sequence RAM 2 clock |
| C13 | sequence RAM 2 trigger |
| C14 | sequence RAM 2 reset |
| C15 | (reserved input) |
| C16 | (reserved input) |
| C17 | multiplexed counter 0 reset |
| C18 | multiplexed counter 1 reset |
| C19 | multiplexed counter 2 reset |
| C20 | multiplexed counter 3 reset |
| C21 | multiplexed counter 4 reset |
| C22 | multiplexed counter 5 reset |
| C23 | multiplexed counter 6 reset |
| C24 | multiplexed counter 7 reset (main external reset) |
| C25 | distributed bus input 0 |
| C26 | distributed bus input 1 |
| C27 | distributed bus input 2 |
| C28 | distributed bus input 3 |
| C29 | distributed bus input 4 / timestamp clock |
| C30 | distributed bus input 5 / timestamp reset input |
| C31 | distributed bus input 6 / timestamp seconds ‘0’ input |
| C32 | distributed bus input 7 / timestamp seconds ‘1’ input |

# Programming Details

## *CR/CSR Support*

The 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.

| CR address | Register | EVG |
|---|---|---|
| 0x27, 0x2B, 0x2F | Manufacturer's ID (IEEE OUI) | 0x000EB2 |
| 0x33, 0x37, 0x3B, 0x3F | Board ID | 0x454700C8 |

For convenience functions are provided to locate VME64x capable boards in the VME crate.

```
STATUS vmeCRFindBoard(int slot, UINT32 ieee_oui, UINT32 board_id,
                      int *p_slot);
```

To locate the first Event Generator in the crate starting from slot 1, the function has to be called following:

```
#include "vme64x_cr.h"
int slot = 1;
int slot_evg;
vmeCRFindBoard(slot, 0x000EB2, 0x344354C8, &slot_evg);
or
vmeCRFindBoard(slot, MRF_IEEE_OUI, MRF_4CHTIM_BID, &slot_evg);
```

If this function returns `OK`, an Event Generator board was found in slot `slot_evg`.

## *Event Generator Function 0 Registers*

The Event Generator specific register are accessed via Function 0 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. For convenience a function to perform this is provided:

```
STATUS vmeCSRWriteADER(int slot, int func, UINT32 ader);
```

To configure Function 0 of an Event Generator board in slot 3 to respond to A16 accesses at the address range 0x1800-0x1FFF the function has to be called with following values:

```
vmeCSRWriteADER(3, 0, 0x18A4);
```

ADER contents are composed of the address mask and address modifier, the above is the same as:

```
vmeCSRWriteADER(3, 0, (slot << 11) | (VME_AM_SUP_SHORT_IO << 2));
```

To get the memory mapped pointer to the configured Function 0 registers on the Event Generator board the following VxWorks function has to be called:

```
MrfEvgStruct *pEvg;
sysBusToLocalAdrs(VME_AM_SUP_SHORT_IO, (char *) (slot << 11),
                  (void *) pEvg);
```

**Note:** using the data transmission capability requires reserving more than 4 kbytes for function 0 i.e. use of addressing mode A24 is suggested, following:

```
vmeCSRWriteADER(3, 0, (slot << 19) | (VME_AM_STD_USR_DATA << 2));
MrfEvgStruct *pEvg;
sysBusToLocalAdrs(VME_AM_STD_USR_DATA, (char *) (slot << 19),
                  (void *) pEvg);
```

## Register Map

| Address Offset | Register | Type | Description |
|---|---|---|---|
| 0x000 | Control | UINT16 | Control/Status Register |
| 0x002 | EventEnable | UINT16 | Event Enable Register |
| 0x004 | SWEvent | UINT16 | Software Event |
| 0x006 | (reserved) | UINT16 | Reserved |
| 0x008 | (reserved) | UINT16 | Reserved |
| 0x00A | (reserved) | UINT16 | Reserved |
| 0x00C | (reserved) | UINT16 | Reserved |
| 0x00E | EventMap0 | UINT16 | Event 0 Mapping Register |
| 0x010 | EventMap1 | UINT16 | Event 1 Mapping Register |
| 0x012 | EventMap2 | UINT16 | Event 2 Mapping Register |
| 0x014 | EventMap3 | UINT16 | Event 3 Mapping Register |
| 0x016 | EventMap4 | UINT16 | Event 4 Mapping Register |
| 0x018 | EventMap5 | UINT16 | Event 5 Mapping Register |
| 0x01A | EventMap6 | UINT16 | Event 6 Mapping Register |
| 0x01C | EventMap7 | UINT16 | Event 7 Mapping Register |
| 0x01E | MXCEnable | UINT16 | Multiplexed Counter Enable Register |
| 0x020 | (reserved) | UINT16 | Reserved |
| 0x022 | (reserved) | UINT16 | Reserved |
| 0x024 | Seq1ClockSel | UINT16 | Sequencer 1 Clock Select Register |
| 0x026 | Seq2ClockSel | UINT16 | Sequencer 2 Clock Select Register |
| 0x028 | ACEnable | UINT16 | AC Sync. Enable Register |
| 0x02A | MXCControl | UINT16 | Multiplexed Counter Control Register |
| 0x02C | MXCPrescaler | UINT16 | Multiplexed Counter Prescaler Register |
| 0x02E | FirmwareVersion | UINT16 | Firmware Version Number |
| 0x030 | (reserved) | UINT32 | Reserved |
| 0x034 | (reserved) | UINT32 | Reserved |

| | | | |
|---|---|---|---|
| 0x038 | (reserved) | UINT32 | Reserved |
| 0x04C | (reserved) | UINT32 | Reserved |
| 0x040 | RFControl | UINT16 | RF Clock Select Register |
| 0x042 | MXCPolarity | UINT16 | Multiplexed Counter Reset Polarity |
| 0x044 | Seq1Addr | UINT16 | Sequencer 1 RAM Address Register |
| 0x046 | Seq1Code | UINT16 | Sequencer 1 RAM Event Code |
| 0x048 | Seq1Time | UINT32 | Sequencer 1 RAM Timestamp |
| 0x04C | Seq1Pos | UINT32 | Sequencer 1 Sequence Current Time |
| 0x050 | Seq2Addr | UINT16 | Sequencer 2 RAM Address Register |
| 0x052 | Seq2Code | UINT16 | Sequencer 2 RAM Event Code |
| 0x054 | Seq2Time | UINT32 | Sequencer 2 RAM Timestamp |
| 0x058 | Seq2Pos | UINT32 | Sequencer 2 Sequence Current Time |
| 0x05C | EvanControl | UINT16 | Event Analyser Control Register |
| 0x05E | EvanEvent | UINT16 | Event Analyser Distributed Bus and Event Code Register |
| 0x060 | EvanTimeHigh | UINT32 | Event Analyser Time Counter (bits 63 – 32) |
| 0x064 | EvanTimeLow | UINT32 | Event Analyser Time Counter (bits 31 – 0) |
| 0x068 | uSecDivider | UINT16 | Divider to get from Event Clock to 1 MHz |
| 0x06A | DataBufControl | UINT16 | Data Buffer Control Register |
| 0x06C | DataBufSize | UINT16 | Data Buffer Size Register |
| 0x06E | DBusEvents | UINT16 | Distributed Bus Event Enable Register |
| 0x070 | (reserved) | UINT32 | Reserved |
| 0x074 | (reserved) | UINT32 | Reserved |
| 0x078 | (reserved) | UINT32 | Reserved |
| 0x07C | (reserved) | UINT32 | Reserved |
| 0x080 | FracDiv | UINT32 | SY87739L Fractional Divider Configuration Word |
| 0x084 | (reserved) | UINT32 | Reserved |
| 0x088 | RxDelay | UINT32 | Receive Delay (Controlled by PPC) |
| 0x08C | TxDelay | UINT32 | Transmit Delay (Controlled by PPC) |
| 0x090 | ADICtrl | UINT32 | ADN2812 Control Word |
| 0x094 | FbTxFrac | UINT32 | Transmit Feedback Fraction |
| 0x098 | (reserved) | UINT32 | Reserved |
| 0x09C | RxDelayInit | UINT32 | Receive Init Delay |
| 0x0A0 | TxDelayInit | UINT32 | Transmit Init Delay |
| 0x0A4 | (reserved) | UINT32 | Reserved |
| 0x0A8 | (reserved) | UINT32 | Reserved |
| 0x0AC | (reserved) | UINT32 | Reserved |
| 0x0B0 | (reserved) | UINT32 | Reserved |
| 0x0B4 | (reserved) | UINT32 | Reserved |
| 0x0B8 | (reserved) | UINT32 | Reserved |

| | | | |
|---|---|---|---|
| 0x0BC | (reserved) | UINT32 | Reserved |
| 0x0C0 | (reserved) | UINT32 | Reserved |
| 0x0C4 | (reserved) | UINT32 | Reserved |
| 0x0C8 | (reserved) | UINT32 | Reserved |
| 0x0CC | (reserved) | UINT32 | Reserved |
| 0x0D0 | (reserved) | UINT32 | Reserved |
| 0x0D4 | (reserved) | UINT32 | Reserved |
| 0x0D8 | (reserved) | UINT32 | Reserved |
| 0x0DC | (reserved) | UINT32 | Reserved |
| 0x0E0 | (reserved) | UINT32 | Reserved |
| 0x0E4 | (reserved) | UINT32 | Reserved |
| 0x0E8 | (reserved) | UINT32 | Reserved |
| 0x0EC | (reserved) | UINT32 | Reserved |
| 0x0F0 | (reserved) | UINT32 | Reserved |
| 0x0F4 | (reserved) | UINT32 | Reserved |
| 0x0F8 | (reserved) | UINT32 | Reserved |
| 0x0FC | (reserved) | UINT32 | Reserved |
| 0x100-0x7FF | (reserved) | | Reserved |
| 0x800-0xFFF | DataBuf | | Data Buffer Transmit Memory |

## Control and Status Register

| address | bit 15 | bit 14 | bit 13 | bit 12 | bit 11 | bit 10 | bit 9 | bit 8 |
|---|---|---|---|---|---|---|---|---|
| 0x000 | MSDIS | FF | RSFIFO | DFIFO | ERRLD | | | VTRG1 |

| Bit | Function |
|---|---|
| MSDIS | Master Disable, write 1 to disable EVG |
| FF | FIFO full flag, write 1 to clear. |
| RSFIFO | Reset FIFO, write 1 to reset FIFO. The FIFO has to be disabled prior to resetting. |
| DFIFO | Disable FIFO/receiver, write 1 to disable upstream receiver, this disabled reception and retransmission of event codes from an upstream EVG. |
| ERRLD | Controls the front panel ERROR-led when the upstream receiver is disabled. |
| VTRG1 | Software trigger sequence RAM 1, Write 1 to trigger. |

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x001 | VTRG2 | RCYL1 | RCYL2 | | | SEQ1 | SEQ2 | RXVIO |

| Bit | Function |
|---|---|
| VTRG2 | Software trigger sequence RAM 2, Write 1 to trigger. |
| RCYL1 | Sequence RAM 1 recycle mode select |
| RCYL1 | Sequence RAM 1 recycle mode select |
| SEQ1 | Stop and Reset Sequence RAM 1 |
| SEQ2 | Stop and Reset Sequence RAM 2 |
| RXVIO | Receiver violation flag, write 1 to clear flag |

## Event Enable Register

| address | bit 15 | bit 14 | bit 13 | bit 12 | bit 11 | bit 10 | bit 9 | bit 8 |
|---|---|---|---|---|---|---|---|---|
| 0x002 | DBUS | | SSEQ1 | SSEQ2 | CMODE | ENEV7 | ENEV6 | ENEV5 |

| Bit | Function |
|---|---|
| DBUS | When set, distributed bus state of upstream EVG is forwarded |
| SSEQ1 | Sequence RAM 1 Single Sequence Mode |
| SSEQ2 | Sequence RAM 2 Single Sequence Mode |
| CMODE | Select RAM 1 control signals for RAM 2 |
| ENEVx | Master enable for trigger event x. |

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x003 | ENEV4 | ENEV3 | ENEV2 | ENEV1 | ENEV0 | ENSQ1 | ENSQ2 | ENVME |

| Bit | Function |
|---|---|
| ENEVx | Master enable for trigger event x. |
| ENSQ1 | Sequence RAM 1 Enable/Disable Sequence |
| ENSQ2 | Sequence RAM 2 Enable/Disable Sequence |
| ENVME | Enable VME Events |

## Software Event Register

| address | bit 7 | bit 0 |
|---|---|---|
| 0x005 | Software event register (write only) | |

## Distributed Bus Data Register

| address | bit 7 | bit 0 |
|---|---|---|
| 0x005 | Distributed bus data register (read only) | |

## Trigger Event 0 Mapping Register

| address | bit 7 | bit 0 |
|---|---|---|
| 0x00f | Trigger Event 0 Mapping Registers | |

## Trigger Event 1 Mapping Register

| address | bit 7 | bit 0 |
|---|---|---|
| 0x011 | Trigger Event 1 Mapping Registers | |

## Trigger Event 2 Mapping Register

| address | bit 7 | bit 0 |
|---|---|---|
| 0x013 | Trigger Event 2 Mapping Registers | |

## Trigger Event 3 Mapping Register

| address | bit 7 | bit 0 |
|---|---|---|
| 0x015 | Trigger Event 3 Mapping Registers | |

## Trigger Event 4 Mapping Register

| address | bit 7 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x017 | Trigger Event 4 Mapping Registers | | | | | | | |

## Trigger Event 5 Mapping Register

| address | bit 7 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x019 | Trigger Event 5 Mapping Registers | | | | | | | |

## Trigger Event 6 Mapping Register

| address | bit 7 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x01b | Trigger Event 6 Mapping Registers | | | | | | | |

## Trigger Event 7 Mapping Register

| address | bit 7 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x01d | Trigger Event 7 Mapping Registers | | | | | | | |

## Multiplexed Counter Enable Register

| address | bit 15 | bit 14 | bit 13 | bit 12 | bit 11 | bit 10 | bit 9 | bit 8 |
|---|---|---|---|---|---|---|---|---|
| 0x01e | MXDB7 | MXDB6 | MXDB5 | MXDB4 | MXDB3 | MXDB2 | MXDB1 | MXDB0 |

| Bit | Function |
|---|---|
| MXDBx | When set, map multiplexed counter output x to distributed bus bit x. |

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x01f | MXEV7 | MXEV6 | MXEV5 | MXEV4 | MXEV3 | MXEV2 | MXEV1 | MXEV0 |

| Bit | Function |
|---|---|
| MXEVx | Enable multiplexed counter output x to generate trigger events. |

## Sequence RAM 1 Clock Select Register

| address | bit 15 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x024 | Sequence RAM 1 Clock Select Register | | | | | | | |

## Sequence RAM 2 Clock Select Register

| address | bit 15 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x026 | Sequence RAM 2 Clock Select Register | | | | | | | |

## AC Input Control Register

| address | bit 15 | bit 14 | bit 13 | bit 12 | bit 11 | bit 10 | bit 9 | bit 8 |
|---|---|---|---|---|---|---|---|---|
| 0x028 | ACSQ2 | ACSQ1 | ACEV0 | ACSYNC | | | | DLYSEL |

| Bit | Function |
|---|---|
| ACSQ1 | Enable AC mains voltage synchronization logic output to trigger sequence RAM 1. (Note! Sequence RAM 2 triggered if CMODE is 1) |
| ACSQ2 | Enable AC mains voltage synchronization logic output to trigger |

| | |
|---|---|
| | sequence RAM 2. (Note! Only when CMODE is 0) |
| ACEV0 | Enable AC mains voltage synchronization logic output to generate trigger event 0. |
| ACSYNC | Synchronization select (0 = event clock, 1 = multiplexed counter 7 output) |
| DLYSEL | AC data register select bit, 0 for divider, 1 for delay. |

| address | bit 7 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x029 | AC Input Divider / Delay Register | | | | | | | |

## Multiplexed Counter Control Register

| address | bit 15 | bit 14 | bit 13 | bit 12 | bit 11 | bit 10 | bit 9 | bit 8 |
|---|---|---|---|---|---|---|---|---|
| 0x02A | MXRS7 | MXRS6 | MXRS5 | MXRS4 | MXRS3 | MXRS2 | MXRS1 | MXRS0 |

| Bit | Function |
|---|---|
| MXRSx | Write 1 to reset multiplexed counter x. |

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x02B | MXSQ2 | MXSQ1 | | | MXHSEL | MXSEL2 | MXSEL1 | MXSEL0 |

| Bit | Function |
|---|---|
| MXSQ1 | Enable multiplexed counter 0 output to trigger sequence RAM 1. (Note! Sequence RAM 2 triggered if CMODE is 1) |
| MXSQ2 | Enable multiplexed counter 1 output to trigger sequence RAM 2. (Note! Only when CMODE is 0) |
| MXHSEL | Multiplexed counter select prescaler high word |
| MXSEL2-0 | Multiplexed counter select register |

## Multiplexed Counter Prescaler Register

| address | bit 15 | | | | | | | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x02C | Multiplexed Counter Prescaler Register | | | | | | | |

## RF Clock Select Register

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x041 | TRSEL | GSEL | TXSEL1 | TXSEL0 | RFSEL1 | RFSEL0 | RFDIV1 | RFDIV0 |

| Bits | Function |
|---|---|
| TRSEL | Transceiver Select:<br>0 – Small Form Factor Pluggable Transceiver (up to 2.5 Gbit/s)<br>1 – 1x9 duplex SC Transceiver (up to 1.25 Gbit/s) |
| GSEL | Gigabit clock source select:<br>0 – Lock Gigabit clock to RF / reference clock<br>1 – Lock Gigabit clock to Upstream EVG |
| TXSEL1-0 | Transmit Data Select:<br>00 – Data transmitted through ADN2812<br>01 – Data transmitted directly from MGT<br>10 – Data transmitted synchronized to bit clock |

| | |
|---|---|
| | 11 – Loop back received data |
| RFSEL1-0 | Event Clock Source Select:<br>00 – RF input divided by 8, 10 or 12<br>01 – SY87729L fractional synthesizer output<br>10 – LVPECL oscillator<br>11 – RF input divided by 4, 5 or 6 |
| RFDIV1-0 | RF input divider select:<br>00 – Divide by 4 or 8<br>01 – Divide by 6 or 12<br>1x – Divide by 5 or 10 |

## MXC Polarity Select Register

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x043 | MXCP7 | MXCP6 | MXCP5 | MXCP4 | MXCP3 | MXCP2 | MXCP1 | MXCP0 |

| Bits | Function |
|---|---|
| MXCPx | Multiplexed Counter Reset State<br>0 – falling edged aligned<br>1 – rising edges aligned |

## Sequencer 1 RAM Address Register

| address | bit 15 | bit 10 | bit 9 | bit 0 |
|---|---|---|---|---|
| 0x044 | | | Sequencer 1 RAM address register | |

## Sequencer 1 RAM Event Code Register

| address | bit 15 | bit 8 | bit 7 | bit 0 |
|---|---|---|---|---|
| 0x046 | | | Sequencer 1 RAM Event Code Register | |

## Sequencer 1 RAM Event Timestamp Register

| address | bit 31 | bit 0 |
|---|---|---|
| 0x048 | Sequencer 1 RAM Event Timestamp Register | |

## Sequencer 2 RAM Address Register

| address | bit 15 | bit 10 | bit 9 | bit 0 |
|---|---|---|---|---|
| 0x050 | | | Sequencer 2 RAM address register | |

## Sequencer 2 RAM Event Code Register

| address | bit 15 | bit 8 | bit 7 | bit 0 |
|---|---|---|---|---|
| 0x052 | | | Sequencer 2 RAM Event Code Register | |

## Sequencer 2 RAM Event Timestamp Register

| address | bit 31 | bit 0 |
|---|---|---|
| 0x054 | Sequencer 2 RAM Event Timestamp Register | |

## Event Analyser Control Register

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x05D | | | | EVANE | EVARS | EVAOF | EVAEN | EVACR |

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

## Event Analyser Data Register

| address | bit 15 | bit 8 | bit 7 | bit 0 |
|---|---|---|---|---|
| 0x05E | (reserved) | | Event Code | |

## Event Analyser Counter Registers

| address | bit 31 | bit 0 |
|---|---|---|
| 0x060 | Event Analyser Counter Register (bits 63 – 32) | |

| address | bit 31 | bit 0 |
|---|---|---|
| 0x064 | Event Analyser Counter Register (bits 31 – 0) | |

## Data Buffer Control Register

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x06B | | | | TXCPT | TXRUN | TRIG | ENA | MODE |

| Bits | 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<br>Write ‘1’ to start transmission of data in buffer |
| ENA | Data Buffer Transmission enable<br>‘0’ – data transmission engine disabled |

| | |
|---|---|
| | '1' – data transmission engine enabled |
| MODE | Distributed bus sharing mode |
| | '0' – distributed bus not shared with data transmission |
| | '1' – distributed bus shared with data transmission |

### Distributed Bus Event Enable Register

| address | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0x06F | DBEV7 | DBEV6 | DBEV5 | | | | | |

| Bits | 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 |

## Network Interface

A 10baseT network interface is provided to upgrade the FPGA firmware and set up boot options. It is also possible to control the module over the network interface.

### *Changing the IP Address of the Module*

The IP address of the module may be changed by sending an ICMP echo request packet with the modules MAC (Media Access Control) address and the IP address the module should respond to. This method is called ARP/PING. Both IP addresses have to be on the same subnet with the subnet mask programmed into the module (defaults to 255.255.255.0).

#### Linux

To set the IP address of an Event Generator module to 192.168.1.32 on a Linux machine (as root):

```
# /sbin/arp –s 192.168.1.32 00:0E:B2:00:00:16
# ping 192.168.1.32
```

Now the board should respond to the echo request with echo replies.

#### Windows

To set the IP address of an Event Generator module to 192.168.1.32 on a Windows machine (in command prompt):

```
C:\> arp –s 192.168.1.32 00-0E-B2-00-00-16

C:\> ping 192.168.1.32
```

Now the board should respond to the echo request with echo replies.

### *Using Telnet to Configure Module*

To connect to the configuration utility of the module issue the following command:

```
telnet 192.168.1.32 23
```

The latter parameter is the telnet port number and is required in Linux to prevent negotiation of telnet parameters which the telnet server of the module is not capable of.

The telnet server responds to the following commands:

| Command | Description |
|---|---|
| b | Show/change boot parameters, IP address etc. |
| h / ? | Show Help |
| i | Read & show dynamic configuration values from FPGA |
| m <address> [<data>] | Read/Write FPGA CR/CSR, Function 0 |
| r | Reset Board |
| s | Save boot configuration & dynamic configuration values into non-volatile memory |
| u | Update IP2022 software |
| q | Quit Telnet |

## Boot Configuration (command b)

Command b displays the current boot configuration parameters of the module. The parameter may be changed by giving a new parameter value. The following parameters are displayed:

| Parameter | Description |
|---|---|
| IP address | IP address of module |
| Subnet mask | Subnet mask of module |
| Default GW | Default gateway |
| V2P IP address | (not used) |
| FPGA mode | FPGA configuration mode<br>0 – FPGA is not configured after power up<br>1 – FPGA configured from internal Flash memory<br>2 – FPGA is configured from FTP server |
| FTP server | FTP server IP address where configuration bit file resides |
| Username | FTP server username |
| Password | FTP server password |
| FTP Filename | FTP server configuration file name |
| Flash Filename | Configuration file name on internal flash |
| UDP Port | UDP server port for FPGA data access |

Note that after changing parameters the parameters have to be saved to internal flash by issuing the Save boot configuration (s) command. The changes are applied only after resetting the module using the reset command or hardware reset/power sequencing.

# *Upgrading FPGA Configuration File*

When the FPGA configuration file resides in internal flash memory a new file system image has to be downloaded to the module. This is done using TFTP protocol:

## Linux

In Linux use e.g. interactive tftp:

```
$ tftp 192.168.1.32
tftp> bin
tftp> put filesystem.bin /
tftp> quit
```

### Windows

In Windows command prompt issue the following command:

```
C:\> tftp –i 192.168.1.32 PUT filesystem.bin /
```

Now the FPGA configuration file has been upgraded and the new configuration is loaded after next reset/power sequencing.

**Note!** Due to the UDP protocol it is recommended to verify (read back and compare) the filesystem image before restarting the module. This is done following:

### Linux

In Linux use e.g. interactive tftp:

```
$ tftp 192.168.1.32
tftp> bin
tftp> get / verify.bin
tftp> quit
$ diff filesystem.bin verify.bin
$
```

If files differ you should get following message:

```
Binary files filesystem.bin and verify.bin differ
```

### Windows

In Windows command prompt issue the following command:

```
C:\> tftp –i 192.168.1.32 GET / verify.bin
C:\> fc /b filesystem.bin verify.bin
Comparing files filesystem.bin and verify.bin
FC: no differences encountered
```

## Event Generator Transition Board (EVG-TB)

The event generator transition board delivers eight external event trigger signals, eight distributed bus signals and an external reset signal to the Event Generator via the P2 connector. LEMO connector style EPY.00.250.NTN will be used in the transition board front panel. The transition module provides a status led to indicate that the board is in operation.

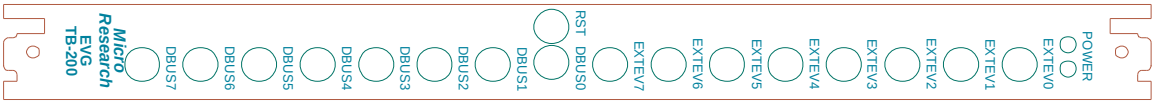


**Figure 15: Event Generator Transition Board (EVG-TB) Front Panel**

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

| Connector / Led | Style | Level | Description |
|---|---|---|---|
| POWER | Green Led | | Transition Board Powered Up |
| EXTEV0 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 0 |
| EXTEV1 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 1 |
| EXTEV2 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 2 |
| EXTEV3 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 3 |
| EXTEV4 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 4 |
| EXTEV5 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 5 |
| EXTEV6 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 6 |
| EXTEV7 | LEMO-EPL | TTL 50 ohm | External Event Trigger Input 7 |
| RST | LEMO-EPY | TTL 50 ohm | External MXC7 Reset Input |
| DBUS0 | LEMO-EPY | TTL 50 ohm | External Distributed Bus Input 0 |
| DBUS1 | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 1 |
| DBUS2 | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 2 |
| DBUS3 | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 3 |
| DBUS4 | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 4 |
| DBUS5 / EVCRS | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 5 / Event Counter Reset Input |
| DBUS6 / SEC0 | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 6 / Seconds ‘0’ Input |
| DBUS7 / SEC1 | LEMO-EPL | TTL 50 ohm | External Distributed Bus Input 7 / Seconds ‘1’ Input |