STD-EVE Hardware Manual

Version 1.0

February/2017

Beam Diagnostics Group (DIG)

Brazilian Synchrotron Light Laboratory (LNLS)

Brazilian Center for Research in Energy and Materials (CNPEM)

About this manual

This manual is intended for people who need information about the STD-EVE hardware. Information about the timing system structure and operation, firmware, or software can be found in the corresponding manuals.

Contents

1 Hardware Specification

STD-EVE is a 19 inches 1U module.
110/220V 50/60Hz AC power supply.

Figure 1: STD-EVE

Table 1: STD-EVE front panel connectors

Connector	Type	Description / Specification
UPLINK	LC Duplex	Fiber for uplink
INH	LEMO EPL.00.250.NTN	Interlock input TTL level 50ohm input impedance
RF OUT	LEMO EPL.00.250.NTN RF	Recovery clock output Square waveform -3dBm (0.23V peak)
OUT0 - OUT7	LEMO EPL.00.250.NTN	3.3V LVTTL level

Table 2: STD-EVE front panel LEDs

LED	Type	Description / Specification
EVG	Green LED	Always Off
EVR	Green LED	Always On
FOUT	Green LED	Always Off
FPGA	Green LED	FPGA downloaded
INH	Red LED	Interlock input activated
ENA	Green LED	STD-EVE enabled
EVT	Yellow LED	(Blink) Event code received
LINK	Green LED	Uplink established

Table 3: STD-EVE rear panel connectors

Connector	**Type**	**Description / Specification**
ETHERNET	RJ45	10/100Mbit Ethernet port
RST	Button	Reset Ethernet

Table 4: STD-EVE rear panel LEDs

LED	**Type**	**Description / Specification**
PWR	Green LED	Power on

2 STD-EVE Hardware Functions

The STD-EVE module is an *Event Receiver* (front panel green LED *EVR* is always on). In order to distinguish it from the STD-EVO/EVR (STD-EVO configured as EVR), it is going to be referred to as EVE. The role of an *Event Receiver* is to convert the data frames broadcast by the *Event Generator* (EVG) into trigger and clock signals that can be transmitted to devices in the accelerator. The STD-EVE module has two inputs in the front panel, which are: *UPLINK*, and *INH*. The signal associated with each input is described below.

The *UPLINK* input is connected to one of the EVG's SFP outputs (generally with fiber optic cables). This input receives data frames broadcast by the EVG containing timing information to be converted into triggers and clocks.

The *INH* input is an interlock active-low input.

The Timing System data frame is composed of two parts of 8 bits each: the event code, which is converted by the EVE into a trigger, and the *Distributed Bus* (*DBUS*), which carries information of 8 different clocks (see figure 2). The parallel frequency for transmission of data frames by the EVG is defined by the EVG's *event clock*, which is a submultiple of the RF frequency. In addition to obtaining the event code and *Distributed Bus* clocks, the *Event Receiver* uses the received data frames to recover the *event clock* and lock itself to the EVG's reference clock.

Event code	*DBus*							
8 bits	7	6	5	4	3	2	1	0

Figure 2: Timing System Data Frame

The EVE has 8 electrical outputs in the front panel (OUT0 - OUT7). Each OUTx can be configured to output one of the *Distributed Bus* clocks, or transmit one of the configured triggers.

The EVE triggers are configured through 24 independent channels (OTP channels). Each OTP channel can be configured to generate a trigger in response to the reception of a given event. The OTP channel also has other parameters, which define the characteristics of the output trigger, i.e., delay, width, polarity, and number of pulses.

The *Event Receiver* has timestamping functinality. It has a timestamp register containing the current UTC timestamp, which is incremented through special event codes, or the *Distributed Bus* bit 6 (DBUS 6) clock, depending on the configuration. The timestamp register contains a subsecond counter as well. This counter is updated by the *event clock* recovered from the *UPLINK*. The module has also a timestamp

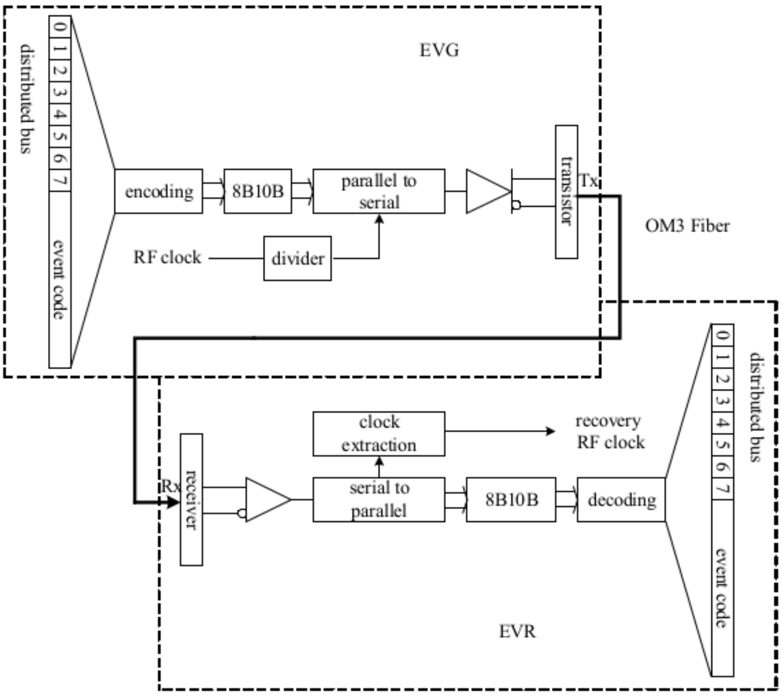
FIFO for storing event codes and the timestamp in which they were received. The *OTP channels* have a parameter which tells whether the event being monitored should be stored upon reception.

Details of the EVE submodules are provided later in this section.

2.0.1 Clock Recovery

The *Event Generator* broadcasts the event code and *Distributed Bus* data frame every event clock period, even when there is no event scheduled, in which case a dummy event with code 0 is broadcast. The data frame is broadcast using 8b10b encoding (see figure 3). The *Event Receivers* use the data frames that arrive from the *UPLINK* to recover the *event clock*.

Figure 3: Timing system encoding-decoding scheme



2.0.2 Trigger Generation

The EVE has 24 *OTP* channels, each of which can be configured to generate a trigger upon the reception of a given event. The OTP channels parameters are configured in the OTPx Register. The *EVT* parameter specifies the event code responsible for generating the trigger. The *DELAY* parameter specifies the delay between the event reception and the output trigger in *event clock* period units. The *WIDTH* parameter specifies the width of the pulse generated in *event clock* period units. The *POL* parameter specifies the pulse polarity. The *PULSES* parameter specifies the number of pulses to be generated in response to the event reception. When *PULSES* is greater than one, the trigger generated is in fact a pulse train. The pulse train period is 2 times *WIDTH* and the duty cycle is 0.5.

The EVE's *Timestamp FIFO* can store the instant when some event is received. Each *OTP* channel has a *TIME* parameter that enables the timestamping function for the event code being monitored.

2.0.3 FP Ouputs

The FP outputs correspond to the OUT0 - OUT7 outputs in the EVE front panel. The OUT0 - OUT7 outputs can be configured in the OUTx Register. The *OUTx* signal source can be mapped to one of the *OTP* channels, or any of the *Distributed Bus* clocks. The source selection is defined by the *SEL* parameter.

The *ITL* parameter enables/disables the interlock function for the corresponding *OUT* channel. When the *OUTx* channel has ITL *enabled*, the channel is inhibited whenever the EVE's interlock input (*INH*) is active (active-low signal).

The OUTx signal has an associated RF delay (*RFDLY*) and fine delay (*FINEDLY*). The RF delay resolution is $\frac{1}{20}$ of *event clock* period. When the value of *RFDLY* is set to 31, the RF delay of the OUTx output is set by the EVG's *Event Sequencer* special event codes 0x40 - 0x53, which set the delay value from 0 to 19 respectively. The RF delay special event codes can be written to the EVG's *Sequence RAM*, in order to give event timestamps enough resolution to, for example, allow single bucket injection. The fine delay resolution is 5ps, and may be used for fine adjustment of the trigger timing.

2.0.4 RF Ouput

The EVE's *RF OUT* output in the front panel is able to provide a square waveform with frequency multiple of the recovered *event clock*. The *RF OUT* can be configured to output x1, x2, x4, x5, and x10 the *event clock*. The *RF OUT* settings are defined in the RF Output Register.

2.0.5 Timestamp

The EVE has timestamping function, which allows it to register when events of interest are received. To this end, the module maintains a timestamp register containing the 64-bit Timing System timestamp, which comprises a 32-bit *UTC* timestamp and a 32-bit *SUBSECOND* timestamp. The EVG sets the EVE's timestamp (generally once), and then increments it every second. The timestamp settings and status can be accessed in the Timestamp Register.

The 32-bit *UTC* timestamp stores the number of seconds passed since some epoch. This value can only be modified by a timestamp broadcast. When the EVG broadcasts its UTC timestamp, the EVE first receives the special event code 0x74. The event 0x74 resets the *SUBSECOND* counter and tells the EVE to treat the next 4 events as UTC information, instead of actual event codes. The UTC information received then replaces the current 4-byte *UTC* value.

The *SUBSECOND* counter is incremented by the recovered *event clock*, providing *event clock* period resolution to the timestamp.

The *TIMESRC* field specifies the PPS source, i.e., the signal which is going to increment the EVE's UTC. When *TIMESRC* is configured as *EVENT*, the *UTC* field is incremented by the special event 0x73, which is sent at the start of every second. When the *TIMESRC* is configured as *DBUS*, the *UTC* field is incremented by the *DBUS 6* clock. When *TIMESRC* is *INTERNAL*, the internal oscillator provides the increment signal. Once a PPS is received, the *SUBSECOND* counter is reset and the *UTC* timestamp is incremented.

In addition to the timestamp special events already mentioned, there are also event code 0x71, which resets the *SUBSECOND* timestamp, and event code 0x72, which resets the *UTC* timestamp.

Timestamp FIFO The EVE has a *Timestamp FIFO*, also referred to as *Timestamp Log*, capable of storing 16384 sets of event code and timestamp (32-bit *UTC* and 32-bit *SUBSECOND*). The purpose of the *Timestamp Log* is to store the instant when relevant events are received. The Timestamp Log Register is used to read and command the *Timestamp FIFO*. Each OTP channel has a timestamping setting, defining whether the event code mapped to that channel should be stored along with its timestamp once received.

The *Timestamp FIFO* operates as a circular buffer, replacing the oldest timestamp value by the newest, in the case that the FIFO is full and a new timestamp is stored. The *EMPTY* and *FULL* flags indicate when the FIFO is empty, or full respectively.

Writing to the *PULL* field pulls the *Timestamp Log* oldest value. Reading *PULL* has no effect. The pull action updates the *Timestamp Log*'s *UTC*, *SUBSECOND*, and *EVENT* fields with the UTC, subsecond, and event code obtained from the *Timestamp FIFO*. The pulled information is erased from the Log. The procedure for reading the *Timestamp Log* consists of writing to *PULL* and reading the Log's *UTC*, *SUBSECOND*, and *EVENT* fields.

The *LOGCOUNT* field indicates how many sets of event code and timestamp are stored in the *Timestamp FIFO*. The *RSTLOG* field resets the *Timestamp FIFO*, i.e., clear all stored values. The *STOPLOG* field stops the timestamping function while set. Both *RSTLOG* and *STOPLOG* return their current value when read.

2.0.6 Registers

Address	Register Name	Description
0	Control and Status Register	General settings and status. Enable/disable module, *UPLINK* status, and interlock input status.
1-16	OTPx Register	OTPx settings. Specifies OTPx trigger delay, width, polarity, number of pulses, and mapped event. Also enables/disables timestamping option.
17-24	OUTx Register	OUTx settings. Specifies OUTx output signal source, RF delay, and fine delay. Also enables/disables the interlock function for the channel being configured.
25	RF Output Register	RF ouput (*RF OUT*) settings. Specify the RF output multiplier.
51	Timestamp Register	Timestamp information and settings. Define PPS signal source, reading of *UTC* and *SUB-SECOND* values.
52	Timestamp Log Register	Timestamp Log settings. *Timestamp FIFO* reading and commanding.
62	Firmware Version Register	12-byte code for current firmware version.
63	Configuration Register	Main STD-EVO configuration options. Alive counter, and *Clock mode* setting.

Control and Status Register [0]

RegC

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
LINK	INHS						

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
							EVREN

EVREN	Enable/Disable EVE. Disabling the EVE disables all of its outputs. 0 Disable 1 Enable
INHS	Interlock input status. The interlock input status only affects the OUTx outpus whose interlock function is enabled. 0 Disserted 1 Asserted
LINK	*UPLINK* status. 0 Unlink 1 Link

OTPx Register [1-16]

OTP0 - OTP15: Address 1 - 16

OTP16 - OTP23: Address 25 - 32

RegA

Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24
EN	POL	TIME					

Bit 23	Bit 16
PULSES	

Bit 15	Bit 8
PULSES	

Bit 7	Bit 0
EVT	

RegB

Bit 31	Bit 0
DELAY	

RegC

Bit 31	Bit 0
WIDTH	

EN	Enable/Disable OTPx. Disabling the OTPx channel does not disable its timestamping function. 0 Disable 1 Enable
POL	OTPx polarity. Specify the polarity of the OTPx output trigger. 0 Normal 1 Inverted
TIME	Enable/Disable OTPx timestamping. When enabled, once the mapped event is received, the corresponding code and timestamp are stored in the *Timestamp FIFO*. 0 Timestamping 1 Not timestamping
PULSES	OTPx number of pulses. Specify the number of pulses to be generated. When the number of pulses is greater than 1, the output is a pulse train with duty cycle 0.5.
EVT	OTPx mapped event. Specify the event to generate the trigger.
DELAY	OTPx delay. Specify the delay between the reception of the mapped event and the trigger output.
WIDTH	OTPx width. Specify the width of the trigger/pulse train pulse(s).

OUTx Register [17-24]

RegA

Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24
ITL							

Bit 23							Bit 16

Bit 15							Bit 8

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	SEL						

RegB

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
						FINEDLY	

Bit 7							Bit 0
FINEDLY							

RegC

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			RFDLY				

ITL — Enable/Disable OUTx interlock function. When the interlock function is enabled, the OUTx will be inhibited while the EVE interlock input (*INH*) is active.

- 0 Disable
- 1 Enable

SEL — OUTx source selection. Specifies the source of the OUTx signal from one of the *OTP* channels or *Distributed Bus* clocks.

- 0x10 - 0x1F OTP0 - OTP15
- 0x20 - 0x27 Dbus0 - Dbus7
- 0x30 - 0x37 OTP16 - OTP23

FINEDLY — OUTx fine delay. The fine delay between the event code reception and the OUTx trigger. The fine delay unit is 5ps.

RFDLY — OUTx RF delay. The RF delay between the event code reception and the OUTx trigger. The RF delay unit is $\frac{1}{20}$ of *event clock* period. When *RFDLY* is set to 31, the delay is defined by *Event Sequencer* event codes 0x40 - 0x53, which set the RF delay to 0 - 19, respectively.

RF Output Register [25]

RegC

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
							CMSEL

CMSEL — RF output multiplier. Configuration of the *RF OUT* output clock frequency, which is a multiple of the recovered *event clock*.

- 0 Disable
- 1 x1
- 2 x2
- 3 x4
- 4 x5
- 5 x10

Timestamp Register [51]

RegA

Bit 31 … Bit 0
UTC

RegB

Bit 31	Bit 0
SUBSECOND	

RegC

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						TIMESRC	

UTC The timestamp UTC field, which is a 32-bit counter to store the number of seconds passed since some epoch. The counter is incremented by the source defined in *TIMESRC*. The UTC can only be modified by EVG timestamp broadcasts or PPS signal.

SUBSECOND The timestamp subsecond field, which is a 32-bit counter to store the subsecond portion of the Timing System timestamp. SUBSECOND is incremented by the recovered *event clock*, and thus, it has *event clock* period resolution. The SUBSECOND is reset whenever a PPS signal or UTC broadcast is received.

TIMESRC The Pulse-per-second signal source, which is responsible for incrementing the timestamp UTC. The PPS signal can be obtained from the clock transmitted by the *Distributed Bus* bit 6, from the special event code 0x73, which is broadcast by the EVG at the start of every second, or from the internal oscillator.

- 0 Idle
- 1 DBUS (DBUS6)
- 2 Event
- 3 Internal

Timestamp Log Register [52]

RegA

Bit 31	Bit 0
UTC	

RegB

Bit 31	Bit 0
SUBSECOND	

RegC

Bit 31	Bit 24
EVENT	

Bit 23							Bit 16
LOGCOUNT							

Bit 15							Bit 8
LOGCOUNT							

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STOPLOG	RSTLOG	PULL				FULL	EMPTY

UTC — Last UTC timestamp pulled from the *Timestamp FIFO*.

SUBSECOND — Last subsecond timestamp pulled from the *Timestamp FIFO*.

EVENT — Last event code pulled from the *Timestamp FIFO*.

LOGCOUNT — Number of sets of event code and timestamp in the *Timestamp FIFO*.

STOPLOG — Stop log function. When enabled, the timestamping function is stopped.

- 0 Disable
- 1 Enable

RSTLOG — Reset log function. When enabled, the *Timestamp FIFO* is cleared.

- 0 Disable
- 1 Enable

PULL — Pull timestamp from *Timestamp FIFO*. Writing to *PULL* moves the oldest set of event code and timestamp from the *Timestamp FIFO* to the *UTC*, *SUBSECOND*, and *EVENT* fields of the *Timestamp Log Register*, where it is available for reading.

FULL — *Timestamp FIFO* full flag. The full flag is set while LOGCOUNT is equal to 16384. While *RSTLOG* is set, both *FULL* and *EMPTY* flags stay in 1.

- 0 FIFO is not full
- 1 FIFO is full

EMPTY — *Timestamp FIFO* empty. The empty flag is set while LOGCOUNT is 0. While *RSTLOG* is set, both *FULL* and *EMPTY* flags stay in 1.

- 0 FIFO is not empty
- 1 FIFO is empty

> When the *UPLINK* signal is lost, *STOPLOG* is automatically set to 1. In this circumstance, *STOPLOG* can only be disabled after a *Timestamp Log* reset (*RSTLOG* set to 1).

Firmware Version Register [62]

RegA

Bit 31	Bit 0
FRMVERSION	

RegB

Bit 31	Bit 0
FRMVERSION	

RegC

Bit 31	Bit 0
FRMVERSION	

FRMVERSION The STD-EVE current firmware version, which is represented by the first 12 characters of the firmware commit hash.

Configuration Register [63]

RegA

Bit 31	Bit 0
ALIVE	

RegB

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			CLKMODE				

RegC

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	1	0	0	0	0	0

ALIVE The alive counter is incremented by the internal oscillator. It starts once the STD-EVE module completes the initialization.

CLKMODE Clock mode. Set according to *UPLINK* event clock frequency.

11	60MHz - 62.5MHz
12	63MHz - 77MHz
13	77.5MHz - 91.5MHz
14	92MHz - 106MHz
15	106MHz - 120.5MHz
16	121MHz - 135MHz

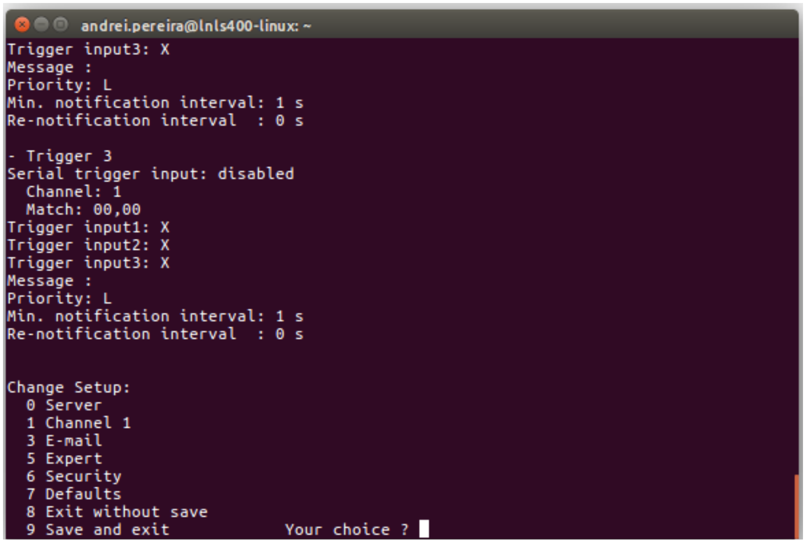
3 Ethernet Network Interface

- 10/100Mbit Ethernet interface.
- UDP protocol by default.
- DHCP client by default.

3.1 Network configuration

In order to modify the module's network configurations, e.g., the IP address, the Telnet program command *telnet <IP address> 9999* can be used. Another option is to use the web browser, and type the module's IP address directly into the address bar, which will open the configuration page.

Figure 4: Setup using Telnet



3.2 Register Read/Write

Both read and write operations use the same structure for the UDP data frame, which is the following:

Byte 0	Byte 1 Byte 4	Byte 5 Byte 8	Byte 9 Byte 12
Command	RegA	RegB	RegC

The *RegA*, *RegB*, and *RegC* sections in the UDP data frame correspond to the 32-bit register sections of same name in each register. These sections carry the information to be written/read. The operation selection and register are specified by the *Command* byte of the UDP data frame.

Read (request) In order to read a register, the UDP data frame *Command* byte must agree with the following rule:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Address					

The first bits, from left to right, must be 1 and 0 respectively, followed by the register address.

Read (response) The response to a read request has a different *Command* byte, which is represented below:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	1	Address					

The first bits, from left to right, must be 1 and 1 respectively, followed by the register address.

Write In order to write to a register, the UDP data frame *Command* byte must agree with the following rule:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	1	Address					

The first bits, from left to right, must be 0 and 1 respectively, followed by the register address.