GTFE

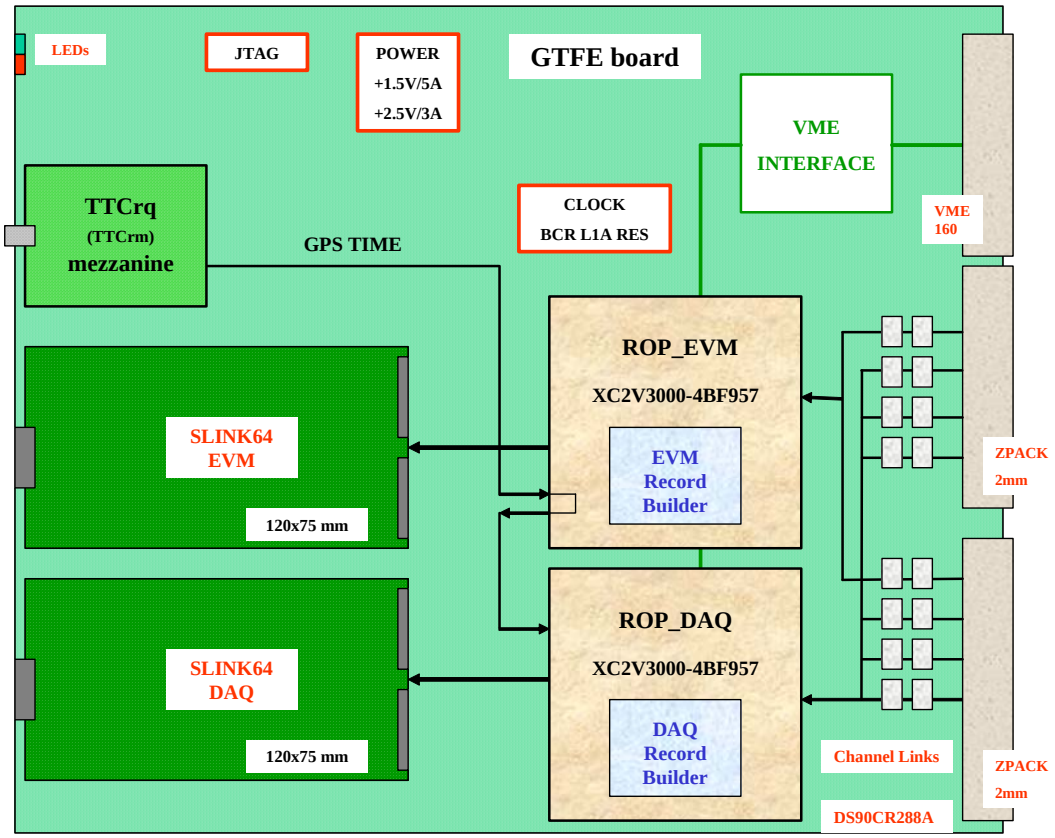
Global Trigger Front End board for Global Trigger for CMS

B. Arnold, H. Bergauer, M. Eichberger, K. Kastner, B. Neuherz, M. Padrta, T. Schreiner, J. Strauss,
A. Taurok

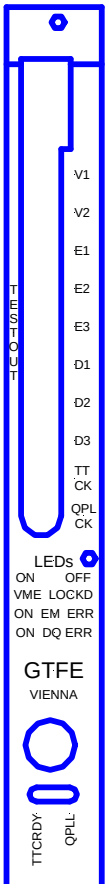
Firmware Versions:
DAQ chip: V102A (6/2009)
EVM chip: V101B (1/2010)
Text corrected
6-Jul-10

Table of contents:

1 GTFE Module



The Global Trigger Front End board (GTFE) receives event data from the FDL, TCS, GMT and all PSB boards and sends it over two SLINK mezzanine boards to the CMS data acquisition system (DAQ).
The TTCrq mezzanine board receives data periodically via an optical fiber from the LHC Beam Synchronous Timing System (BST) including GPS time, beam type and other run parameters.
The ROP_EVM chip merges event data of the TCS and FDL, adds BST information and sends the event record to the CMS Event Manager.
The ROP_DAQ chip merges the event data of the GMT, all PSB boards and also from the FDL board and sends the event record to the CMS_DAQ like any other subdetector.
The VME64 and the VME chip provide the interface using VME protocol to control and monitoring software.
All chips receive the common 40 MHz clock system via the backplane from the Timing board (TIM).
Several LEMO test connectors on the front panel allow to monitor internal signals of the FPGA chips. The internal signals are selected by software programmable TESTMASKS.



1.1 LEDS on Front Panel

1st row: LEDs for board
Green: ON GTFE board is active
Red: OFF GTFE board is not active
2nd row: LEDs
Green: VME is illuminated during VME access
Green: LOCKED The DAQ and EVM chips are locked to the on board clock
3rd row: LEDs for EM:
Green: ON Slink64 for EVM is on and ready to accept event records
(=SLINK is 'NOT FULL' AND 'NOT DOWN')
Red: ERR a not ignored error is active in the EVM chip
4th row: LEDs for DQ:
Green: ON Slink64 for DAQ is on and ready to accept event records.
(=SLINK is 'NOT FULL' AND 'NOT DOWN')
Red: ERR a not ignored error is active in the DAQ chip
LEDS of TTCrq mezzanine board:
Left red LED: TTCRDY TTCrx chip is locked
Right red LED: QPLL QPLL chip is locked

1.2 Test Outputs

V1, V2 Testpoints for VME chip: See description of VME_CHIP_GTFE.
E1, E2, E3 Testpoints of EVM chip: See registers for EVM: TESTMASK1,2,3
D1, D2, D3 Testpoints of DAQ chip: See registers for DAQ: TESTMASK1,2,3
TT CK Clock signal from TTCrx chip on the TTCrq module.
QPL CK Clock signal from QPLL chip on the TTCrq module.

If both 40 MHz clock signals can be seen then the optical link to the BST system is working.

2 VME64 and VME chip of GTFE-9U-card

See descriptions of VME64 and VME chip.

3 ROP_DAQ chip

The ROP_DAQ chip contains a SIM_SPY memory used either to monitor event data being sent to the SLINK or to send simulation events. The SIM_SPY memory is implemented as a 32kx16 Dual Port Memory(DPM). On one side VME accesses the SIM_SPY memory as 32k x 16 bit. On the other side the memory is configured as 8k x 64 bit memory to spy data going to the SLINK64 or to send simulation data to the SLINK64.

3.1 Firmware Updates

Version 102A event_length for Trailer word corrected
Version 1029 …No software change required.
EVENT RECORD in Spy memory corrected
gtfe_simspy_ctrl: spy_full flag should now appear
event_length_calculation corrected
Version 0027: New register to mix 3- and 5-BC records
BB10 0046 DAQ_5BX_REGISTER w/r
Automatic EVENT LENGTH CALCULATION!!
➔ EVENT_LENGTH_15_0, EVENT_LENGTH_31_16 not used anymore

Version 0026: VersionNr., Chip-ID, Power-up values in package file
.../rop_types/hdl/rop_pkg.vhd
'l1a_dlyed' with clk40 as in EVM chip V0018
'vme_en2' added to make 'vme_dis_rd'
Version 0025: Increment signals for monitoring counters corrected.
Version 0024: Monitoring counters added
Corrected chan_recvr module: without error in empty counter
Cmd_pulse(13):= REFRESH_DAQ_MON_COUNTER

Version 0023: Flags of inactive channels (boards) are masked out.
Version 0022: IGNORE ERRORS changed and 004C…power-upvalue
Version 0021: too_many_L1A in chain logic → sets busy flag
Version 0020: ROC: l1res at begin + end of event only, TESTMASK2,3 signal positions changed.
Version 001D: ROC: begin of GMT logic changed to be like psb-logic,
End_of_event logic simplified
Version 001C: New fifo16to64_4k with corrected empty error, ROC: wait 1023 for GMT,
cmd_reg(7)= 0 allows L1Reset to reset S-Link
TESTMASK2 &3: status bits to EVM inserted.
Version 001B: Reset Slink by VME only, not by L1Reset
Version 001A: ###### run_rop = '1' ###### l1a_latency register removed
Version 0019: l1a_latency register + l1a_delay added!! (l1a_dlyed -->evnr_check, bcnr_fifo)
fifo_16to64_4k corrected
Version 0018: chan_recvr : 11x fifo_1k =44ramb,
gmt: fifo_4k =16 ramb ==> 60ramb; simspy+bcnr_fifo =33ramb
60+33= 93 <96ramb(max)
Version 0017: ROC : GMT logic simplified (send 1bxdata, then check Slink)
Version 0016: daq_evnr_check: changed to 32 bit counters
Version 0015: daq_status_encoder: sl_full(inkl.ignore bit) included

Version 0014: chan_recvr_gmt: Empty logic for GMT per event, not per word
Version 0013: Send GT-Status to EVM chip: GPS 56-->48 bits, ROC,daq_evnr_check modified
Version 0012: ROC: length for GMT extended to 3x17, 5x17 w64
Version 0011: chlink_ok register, chan_receiver_gmt, ROC:res_roc_vme
Version 0010: corrected simspyctrl avoids 'spurious' events when switching from spy to sim mode
Version V000F: 20.3.07: cmd_pulse(15)=res_runff_vme, daq_status(11):= run_ff
ROC changed
Version V000A: S64_NWEN on DAQ_TEST(0), new SM-States (300x)
Version: V0009
Show **rd_fifo** (11:0) on testpoints DAQ_TEST(11:0),
Show **wr_fifo** (11:0) on testpoints DAQ_TEST(11:0),
FIFO logic corrected

3.2 General information

Configuration file: Size for XC2V3000 chip

- Bit file 1,3 Mbytes
- 3 MCS files: 1,5 + 1,5 + 1,0 Mbytes
- SVF file (JTAG): 10,2 Mbytes

Required RAM blocks: XC2V2000 (56 RAMB), XC2V3000 (96RAMB)

11 Board_receiver FIFOs each 2 RAMB ➔ 22 RAMB to contain >20 3bx_events
1 board receiver FIFOs for GMT 4 RAMB ➔ 4 RAMB to contain >20 3bx_events
1 bcnr FIFO ➔ 1 RAMB
SIMSPY for SLINK: (32k x 16) ➔ 32 RAMB to contain ~40 3bx_events
total ➔➔ 59 RAMB

Size of 3bx event = 199 W64 ~800 W16

3.3 DAQ chip overview VME Addresses

A31-A24: = 'BB' = base address
A23-20: = 0001 …DAQ chip
A23-20: = 0010 …EVM chip

23-20	19-16	15-12	11	10	9	8	7	6	5	4	3	2	1	0
0001	0000	0000	0	Register space (1kx16 readback)										
0001	0000	0000	1	STATUS REGISTERS										
0001	0001	SIM SPY MEMORY 32kx16bits												

3.4 DAQ_SIM_SPY MEMORY

BB01 0000 DAQ_SIM_SPY_MEMORY (32kx16)

The SIM_SPY memory is implemented as Dual Port Memory(DPM). On one side VME accesses the SIM_SPY memory as 32k x 16 bit. On the other side the memory is configured as 8k x 64 bit memory to spy data going to the SLINK64 or to send simulation data to the SLINK64.

VME side 16 bits	← →	SLINK side 64 bits
Addr 0: bits 15-0	← →	Adr0: bits 15-0
Addr 1: bits 15-0	← →	Adr0: bits 31 -16

Addr 2: bits 15-0	← →	Adr0: bits 47 - 32
Addr 3: bits 15-0	← →	Adr0: bits 63 - 48
Addr 4 bits 15-0	← →	Adr1 bits 15-0
Addr 5 bits 15-0	← →	Adr1 bits 31 -16
Addr 6 bits 15-0	← →	Adr1 bits 47 - 32
Addr 7 bits 15-0	← →	Adr1 bits 63 - 48

3.5 Register overview

Setup registers

BB10 0000 BCRES_DELAY w/r

BB10 0002 SHOW_SLINK_BITS w/r // mask register to show S-Link64 bits

BB10 0004 DAQ_CMD_REG w/r // sim_mode, enable bits,
// set SLINK mode etc.

BB10 0006 MAX_BC_NUMBER w/r //=orbit length -1 = **0DEB**

BB10 0008 INITIAL_CRC w/r // bits 15_0
// initial CRC value =**FFFF**

BB10 000A CMS_HEADER19_4 w/r //CMS HeaderWord1(19:4)
bits15:4:= GT-identifier in DAQ software: 813 =32D(hex)
bits 3:0:= FORMAT_VERSION currently =0000 ➔ ***default value = 32D0***

BB10 000C EVENT_LENGTH_15_0 w/r // bits 15_0 : 24 bit length

BB10 000E EVENT_LENGTH_31_16 w/r // bits 31_16 : 24 bit length
// optional: Hardware Adder using ACTIVE_BOARDS

BB10 0010 BOARD_ID /r
// bits 15:8: Board_ID of GTFE = VME slot number= 17dec=11hex
// bits 7:0: length in bx =3 or 5 ➔ ***default value = 1103***

BB10 0012 SETUP_VERSION_15_0 w/r // bits 15_0 //GFTE HeaderWord1

BB10 0014 SETUP_VERSION_31_16 w/r // bits 31_16 //GFTE HeaderWord1

BB10 0016 ACTIVE_BOARDS w/r //GFTE HeaderWord2

BB10 0018 SPY_FULL_LIMIT w/r // bits 15_0 // Spy control word

BB10 001A TEST_MASK1 w/r // bits 15_0

BB10 001C TEST_MASK2 w/r // bits 15_0

BB10 001E TEST_MASK3 w/r // bits 15_0

Status words read only

BB10 0020 DAQ_CHIP_IDL -/r // bits 15_0

BB10 0022 DAQ_CHIP_IDH -/r // bits 31_16

BB10 0024 DAQ_VERSION_NR -/r // Firmware Version nr.

BB10 0026 DAQ_STATUS -/r // General status with SLINK

BB10 0028 CHAN_FIFOS_FULL -/r // FULL flags of all channels

BB10 002A CHAN_FIFOS_WARN50 -/r // WARN50 flags of all channels

BB10 002C CHAN_FIFOS_EMPTY -/r // EMPTY flags of all channels

BB10 002E CHAN_FIFOS_WARN75 -/r // WARN75 flags of all channels

BB10 0030 CHAN_LINK_BAD_CODE -/r // Bad Header Code flags

BB10 0032 CHAN_LINK_DATA_LOST -/r // ..of all channels

BB10 0034 CHAN_LINK_OK -/r // ..of all channels

BB10 0036 STATE_MACHINE_STATUS -/r // actual status of ROP-sm

BB10 0038 SIM_SPY_ADDRESS -/r // actual value of spy address

BB10 003A xxxxx -/r //

BB10 003C xxxxx -/r //

BB10 003E xxxxx -/r //

Write registers

BB10 0040	CMD_PULSE	w/	//
BB10 0042	IGNORE_ERR_FOR_TCS	w/r	// bits 15_0
BB10 0044	DAQ_CMD_REG2	w/r	// bits 15_0
BB10 0046	DAQ_5BX_REGISTER	w/r	// 1= 5bc record, 0=3bc record
BB10 0048 -5E	free		

Monitoring counters

BB10 0100	CNTR_WAITING_FOR_FDL_DAQ_L	-/r
BB10 0102	CNTR_WAITING_FOR_FDL_DAQ_H	-/r
BB10 0104	CNTR_FIFO_FULL_FDL_DAQ_L	-/r
BB10 0106	CNTR_FIFO_FULL_FDL_DAQ_H	-/r
BB10 0108	CNTR_FIFO_WARN75_FDL_DAQ_L	-/r
BB10 010A	CNTR_FIFO_WARN75_FDL_DAQ_H	-/r
BB10 010C	CNTR_FIFO_WARN50_FDL_DAQ_L	-/r
BB10 010E	CNTR_FIFO_WARN50_FDL_DAQ_H	-/r
BB10 0110	CNTR_WAITING_FOR_PSB9_L	-/r
BB10 0112	CNTR_WAITING_FOR_PSB9_H	-/r
BB10 0114	CNTR_FIFO_FULL_PSB9_L	-/r
BB10 0116	CNTR_FIFO_FULL_PSB9_H	-/r
BB10 0118	CNTR_FIFO_WARN75_PSB9_L	-/r
BB10 011A	CNTR_FIFO_WARN75_PSB9_H	-/r
BB10 011C	CNTR_FIFO_WARN50_PSB9_L	-/r
BB10 011E	CNTR_FIFO_WARN50_PSB9_H	-/r
BB10 0120	CNTR_WAITING_FOR_PSB13_L	-/r
BB10 0122	CNTR_WAITING_FOR_PSB13_H	-/r
BB10 0124	CNTR_FIFO_FULL_PSB13_L	-/r
BB10 0126	CNTR_FIFO_FULL_PSB13_H	-/r
BB10 0128	CNTR_FIFO_WARN75_PSB13_L	-/r
BB10 012A	CNTR_FIFO_WARN75_PSB13_H	-/r
BB10 012C	CNTR_FIFO_WARN50_PSB13_L	-/r
BB10 012E	CNTR_FIFO_WARN50_PSB13_H	-/r
BB10 0130	CNTR_WAITING_FOR_PSB14_L	-/r
BB10 0132	CNTR_WAITING_FOR_PSB14_H	-/r
BB10 0134	CNTR_FIFO_FULL_PSB14_L	-/r
BB10 0136	CNTR_FIFO_FULL_PSB14_H	-/r
BB10 0138	CNTR_FIFO_WARN75_PSB14_L	-/r
BB10 013A	CNTR_FIFO_WARN75_PSB14_H	-/r
BB10 013C	CNTR_FIFO_WARN50_PSB14_L	-/r
BB10 013E	CNTR_FIFO_WARN50_PSB14_H	-/r
BB10 0140	CNTR_WAITING_FOR_PSB15_L	-/r
BB10 0142	CNTR_WAITING_FOR_PSB15_H	-/r
BB10 0144	CNTR_FIFO_FULL_PSB15_L	-/r
BB10 0146	CNTR_FIFO_FULL_PSB15_H	-/r
BB10 0148	CNTR_FIFO_WARN75_PSB15_L	-/r
BB10 014A	CNTR_FIFO_WARN75_PSB15_H	-/r
BB10 014C	CNTR_FIFO_WARN50_PSB15_L	-/r

BB10 014E	CNTR_FIFO_WARN50_PSB15_H	-/r
BB10 0150	CNTR_WAITING_FOR_PSB19_L	-/r
BB10 0152	CNTR_WAITING_FOR_PSB19_H	-/r
BB10 0154	CNTR_FIFO_FULL_PSB19_L	-/r
BB10 0156	CNTR_FIFO_FULL_PSB19_H	-/r
BB10 0158	CNTR_FIFO_WARN75_PSB19_L	-/r
BB10 015A	CNTR_FIFO_WARN75_PSB19_H	-/r
BB10 015C	CNTR_FIFO_WARN50_PSB19_L	-/r
BB10 015E	CNTR_FIFO_WARN50_PSB19_H	-/r
BB10 0160	CNTR_WAITING_FOR_PSB20_L	-/r
BB10 0162	CNTR_WAITING_FOR_PSB20_H	-/r
BB10 0164	CNTR_FIFO_FULL_PSB20_L	-/r
BB10 0166	CNTR_FIFO_FULL_PSB20_H	-/r
BB10 0168	CNTR_FIFO_WARN75_PSB20_L	-/r
BB10 016A	CNTR_FIFO_WARN75_PSB20_H	-/r
BB10 016C	CNTR_FIFO_WARN50_PSB20_L	-/r
BB10 016E	CNTR_FIFO_WARN50_PSB20_H	-/r
BB10 0170	CNTR_WAITING_FOR_PSB21_L	-/r
BB10 0172	CNTR_WAITING_FOR_PSB21_H	-/r
BB10 0174	CNTR_FIFO_FULL_PSB21_L	-/r
BB10 0176	CNTR_FIFO_FULL_PSB21_H	-/r
BB10 0178	CNTR_FIFO_WARN75_PSB21_L	-/r
BB10 017A	CNTR_FIFO_WARN75_PSB21_H	-/r
BB10 017C	CNTR_FIFO_WARN50_PSB21_L	-/r
BB10 017E	CNTR_FIFO_WARN50_PSB21_H	-/r
BB10 0180	CNTR_WAITING_FOR_GMT_L	-/r
BB10 0182	CNTR_WAITING_FOR_GMT_H	-/r
BB10 0184	CNTR_FIFO_FULL_GMT_L	-/r
BB10 0186	CNTR_FIFO_FULL_GMT_H	-/r
BB10 0188	CNTR_FIFO_WARN75_GMT_L	-/r
BB10 018A	CNTR_FIFO_WARN75_GMT_H	-/r
BB10 018C	CNTR_FIFO_WARN50_GMT_L	-/r
BB10 018E	CNTR_FIFO_WARN50_GMT_H	-/r
BB19 0190	DAQ_MONITORING_TIME _L	-/r
BB19 0192	DAQ_MONITORING_TIME _H	-/r
BB19 0194	CNTR_EVENTS_TO_DAQ_SLINK_L	-/r
BB19 0196	CNTR_EVENTS_TO_DAQ_SLINK_H	-/r
BB19 0198	CNTR_FFF_RECORDS_DAQ_L	-/r
BB19 019A	CNTR_FFF_RECORDS_ DAQ_H	-/r
BB19 019C	CNTR_DAQ_SLINK_DOWN_L	-/r
BB19 019E	CNTR_DAQ_SLINK_DOWN_H	-/r
BB10 01A0	CNTR_DAQ_SLINK_FULL_L	-/r
BB10 01A2	CNTR_DAQ_SLINK_FULL_H	-/r
BB10 01A4	CNTR_WAITING_FOR_DAQ_SLINK_L	-/r
BB10 01A6	CNTR_WAITING_FOR_DAQ_SLINK_H	-/r

BB10 01A8
BB10 01AA
BB10 01AC
BB10 01AE

3.6 CMS Header & Trailer word

	63-60	59-56	55- 32	31 - 20	19 - 8	7 -4	3	2	1	0
	BOE_1	Evt_typ	LV1_id (24) Event-Nr	BX_id(12)	Source_id (12)	FOV	H	x	$	$
	16 + 16 + 16 + 16 bits data words *//record is of constant length*									
	EOE_1	xxxx	Event_length (24)	CRC(16); xxxx; Event_stat(8)			T	R	$	$

HEADER word:
Bit 63-60: BOE = "0101"=5 Begin of event (fixed)
Bit 59-56: Event type // from TCS, see also table below ????
Bit 55-32: Event Nr // reference event number since last L1Reset,
//**1st EVNR=1 after L1Reset,ResEvnr,HardRes**
Bit 31-20: BX-id // reference BC number
Bit19-8: Source_id = 813 =32D =GT-identifier //**programmable register bits**
Bit7-4: FOV =000...DAQ-Version of Format **// programmable register bits**
Bit3: H= 0...last header word; =1 another header word follows
Bit2,1,0: =000...undefined or used by S-Link hardware
TRAILER word:
Bit 63-60: EOE = "1010" =A End of event (fixed)
Bit 59-56: 0000 // undefined
Bit 55-32: Event Length // **length in W64 including header+trailer word**
Bit 31-16: CRC // **calculated including last word with crc=0000.**
Bit 15-12: 0000 // undefined
Bit 11- 8: Eventstatus 4 bits // **still undefined, probably =sync-error bits of event building.**
Bit 7- 4: TTS 4 bits //current values of TTS bits ????
Bit 3: T=0 last trailer word; =1 other trailer word follows
Bit 2: R=0
Bit1: x=0 ...reserved bit
Bit0: $=0 ... used by S-Link hardware

Evt_type	*name*	*comment*
0001	Physics trigger	FinOR
0010	Calibration trigger	Calibration cycle
0011	Test trigger	Test cycle
0100	Technical trigger	Technical Trigger
0101	Simulated event	Reserved for DAQ
0110	Traced event	Reserved for DAQ
1111	Error	
other	undefined	

3.7 Register descriptions

3.7.1 CMD_PULSE

BB10 0040 CMD_PULSE w/ //
Bit 15: res_runff_vme // resets RUN_FF and stops ROC-state machine (V000F)
Bit 14: ---
Bit 13: REFRESH_DAQ_MON_COUNTERS // moves counter contents into registers and clears the monitoring counters.
Bit 12: ---
Bit 11: L1A_vme // simulate a L1A pulse for a standalone test
Bit 10: res_roc_vme // reset ROC state machine by VME
Bit 9: res_slink_vme // reset SLINK by VME: ~800 ns pulse
Bit 8: stop_spying_simulating
Bit 7: start_spying_ simulating
Bit 6: stop_rop_vme
Bit 5: start_rop_vme
Bit 4: res_bc_error // clears BC ERROR flag and also front panel LED
Bit 3: ResEvnr_vme
Bit 2: ResOrbnr_vme
Bit 1: BCRes_vme
Bit 0: L1Res_vme
//ROC = **R**ead**o**ut **C**ontroller is implemented as finite state machine

3.7.2 IGNORE_ERR_FOR_TCS

BB10 0042 IGNORE_ERR_FOR_TCS w/r // bits 15_0
Proposed value for tests and first data taking runs: **X"004C"** (=power-up value of V0022)
If one of the ignore bits is set then the respective status code is not transmitted to the TCS Trigger Control board (via the EVM chip).
If any of the not ignored errors is active then the ERR_LED is illuminated.
Bit15-10 not used
Bit9 := 1 ignore ROC_BUSY FLAG → busy
Bit8 := 1 ignore ROC_ERROR FLAG → error
Bit7 := 1 ignore SLINK DOWN FLAG → error
Bit6 := 1 ignore BC_ERROR → error
Bit5 := 1 ignore FIFO WARN50 FLAGS → warning //since V0022
Bit4 := 1 ignore FIFO WARN75 FLAGS → busy //since V0022
Bit3 := 1 ignore DATA_LOST FLAGS → not included in out_of_sync to TCS
Bit2 := 1 ignore BAD_HEADER FLAGS → not included in error to TCS
Bit1 := 1 ignore TOO_MANY_L1A_IN_CHAIN → busy //since V0022
Bit0 := 1 ignore FIFO FULL FLAGS → out_of_sync

3.7.3 DAQ_CMD_REG2

BB10 0044 DAQ_CMD_REG2 w/r // bits 15_0
Bit15-1 not used
Bit0 := 1 RESET_SLINK, =0 release RESET_SLINK signal
This option can be used if the Slink requires a reset signal for more than 800 ns.

3.7.4 DAQ_5BX_REGISTER

BB10 0046 DAQ_5BX_REGISTER w/r // bits 15_0
For each board select the record length separately.
Bit = 1 … 5 BC record from this board expected
Bit = 0 … 3 BC record from this board expected (default)

BIT#	BOARD	SLOT# in crate	Remark	
0	FDL	10	Final Decision board	
1	PSB_0	9	Techn.Triggers for FDL	
2	PSB_1	13	Calo data for GTL	
3	PSB_2	14	Calo data for GTL	
4	PSB_3	15	Calo data for GTL	
5	PSB_4	19	M/Q bits for GMT	
6	PSB_5	20	M/Q bits for GMT	
7	PSB_6	21	M/Q bits for GMT	
8	GMT	18	Global Muon Trigger	
9	*GMT_spare*	*18*	*Not used*	
10	*TCS_M*	*9*	*Not used*	
11	TIM	16	Timing board	
12-15	*0 0 0 0 0*	*Not used*	*Nothing assigned*	

3.7.5 BCRES_DELAY

BB10 0000 BCRES_DELAY w/r

--UNIT= ½ BC (12.5 ns)

15 - 12	11 - 8	7 - 4	3 - 0
Delay C	Delay B	Delay A	Delay= 0…3

Total Delay = Delay C + Delay B + Delay A + (0…3)

-- DELAY =0 ➔ 0000 0000 0000 0000
-- DELAY =1 ➔ 0000 0000 0000 0001
-- DELAY =2 ➔ 0000 0000 0000 0010
-- DELAY =3 ➔ 0000 0000 0000 0011
-- DELAY =C+B+A+3 ➔ CCCC BBBB AAAA 0011

-- **For DELAY <4 the bits 15-4 have to be =0 !!**
-- Bits 3,2 are always =0; are not decoded

3.7.6 SHOW_SLINK_BITS

BB10 0002 SHOW_SLINK_BITS w/r

Signals for the frontpanel LEMO connectors go through 2 FF (2x12.5ns) inside the FPGA and are then delayed by 50 Ohm drivers for about 5-6 ns. Signals for the DAQ_TEST(i) testpoints go through 1 FF (12.5ns) inside the FPGA and then immediately to the test point pins.
➔ Signals on DAQ_TEST(i) test-points appear about 18.5 ns before the signals on the frontpanel LEMO connectors if they are generated at the same time.
Mask register to show S-Link64 bits.
If a mask bit =1 then 15 signals are switched to 16 DAQ_TEST points.
If all mask bits = 0 then all DAQ_TEST points are switched off (0V).
Bit 0 : =1 Show S64_D(15:0) on testpoints DAQ_TEST(15:0)
Bit 1 : =1 Show S64_D(31:16) on testpoints DAQ_TEST(15:0)
Bit 2 : =1 Show S64_D(47:32) on testpoints DAQ_TEST(15:0)
Bit 3 : =1 Show S64_D(63:48) on testpoints DAQ_TEST(15:0)

Bit 4 : =1 Show S64_NWEN on DAQ_TEST(0) // to S-Link64
Show S64_NCTRL on DAQ_TEST(1) // to S-Link64
Show S64_NRESET on DAQ_TEST(2) // to S-Link64

Show S64_NTEST on DAQ_TEST(3) // to S-Link64
Show S64_DW(0) on DAQ_TEST(4) // to S-Link64
Show S64_DW(1) on DAQ_TEST(5) // to S-Link64
Show S64_DOWN on DAQ_TEST(6) // from S-Link64
Show S64_FULL on DAQ_TEST(7) // from S-Link64
Show S64_LRL(3:0) on DAQ_TEST(11:8) // from S-Link64
Show S64_RESV(2:0) on DAQ_TEST(14:12) // from S-Link64
Show SEL_HEADER on DAQ_TEST(15) // begin of record

Bit 5 : =1 Show REC_CLK(11:0) on testpoints DAQ_TEST(11:0),
DAQ_TEST(15:12) =0000

Bit 6 : =1 Show **rd_bc_fifo** on testpoint DAQ_TEST(15)
Show **000** on testpoints DAQ_TEST(15:12)
Show **rd_fifo** (11:0) on testpoints DAQ_TEST(11:0),
Bit 7 : =1 Show **0000** on testpoints DAQ_TEST(15:12)
Show **wr_fifo** (11:0) on testpoints DAQ_TEST(11:0),

3.7.7 DAQ_CMD_REG

BB10 0004 DAQ_CMD_REG w/r // sim_mode, enable bits
Bit 15 : DAQ_EVENT_TYPE 3 // Event type bit3 defined by software
Bit 14 : DAQ_EVENT_TYPE 2 // Event type bit2 defined by software
Bit 13 : DAQ_EVENT_TYPE 1 // Event type bit1 defined by software
Bit 12 : DAQ_EVENT_TYPE 0 // Event type bit0 defined by software
Bit 11: ONE_EVENT_MODE
= 1: Write only one event into the sim/spy memory overwriting the previous event.
= 0: Accumulate events until sim/spy memory becomes full
Bit 10: not used since V0027; use DAQ_RECORD_LENGTH register instead
~~**LENGTH_5BX // Record Length for 5bx per Event expected**~~
Bit 9: IGNORE_SLINK_DOWN // 1= ignore that the S-Link is down to send data
// whether the SLINK is connected or not
Bit 8: IGNORE _SLINK_FULL // ignore that the S-Link is full to send data
// whether the SLINK is full or not
Bit 7 : RES_SLINK_WITHOUT_RESYNC (V001C)
// 1= reset by VME only; 0= Resync and VME will reset the S-LINK chip
Bit 6 : SPY_EVERY_TICK
= 1: During an event transfer even when not writing into the S-Link the data on the S-Link bus are stored in the sim/spy memory.
= 0: Only valid words going to the S-Link are written into the sim/spy memory.
Bit 5: SIM_MODE ….simulation mode
=1: Data are transmitted from the SIM/SPY memory to the S-Link64 and data from Channel Links are ignored. One event only can be transmitted.
Bit 4: SLINK_TEST_MODE default: =0
Bit 3 : DEBUG_CHANLINKS … is valid for all channels
=0: standard data taking mode
=1: All Channel Link data are recorded, except when
header = “5”➔ IDLE code or
header = “F”➔ End of record
Bit 2 : EN_ROBUS // enable ROBUS signals from TIM board
Bit 1 & 0:
00 ➔ DAQ_ROP DISCONNECTED for TCS
01 ➔ DAQ_ROP BUSY_FOR_TCS

10 ➔ DAQ_ROP READY_FOR_TCS
11 ➔ DAQ_ROP DISCONNECTED for TCS

3.7.8 MAX_BC_NUMBER

BB10 0006 MAX_BC_NUMBER w/r //=orbit length -1
Default value := 0deb = 3564-1

3.7.9 INITIAL_CRC

BB10 0008 INITIAL_CRC w/r // bits 15_0 // initial CRC value
Default: “FFFF”

3.7.10 CMS_HEADER19_4 - CMS HeaderWord1

BB10 000A CMS_HEADER19_4 w/r //CMS HeaderWord1(19:4)
bits15:4:= GT-identifier in DAQ software: 813 =32D(hex)
bits 3:0:= FORMAT_VERSION currently =0000 ➔ **default value = 32D0**

3.7.11 EVENT_LENGTH

V0027: Replaced by automatic EVENT LENGTH CALCULATOR !!

BB10 000C **EVENT_LENGTH_15_0 w/r // bits 15_0 : 24 bit length**
BB10 000E **EVENT_LENGTH_31_16 w/r // bits 31_16 : 24 bit length**
See GT_Readout_format.pdf on GlobalTrigger Webpage for latest update.!!!

Calculate Event Length according to register bits of
ACTIVE BOARDS and EVENT_LENGTH

3 Bunch-crossings per event for all boards: **202(=00CA) W64** =
1 HEADER + 2 GTFE + 21 FDL + 18 PSB_slot9 + 18 PSB_ slot13 + 18 PSB_ slot14 + 18 PSB_ slot15 + **51** GMT + 18 PSB_ slot 19 + 18 PSB_ slot 20 + 18 PSB_ slot 21 + 1 TRAILER
5 Bunch-crossings per event for all boards: **334 (=014E)W64** =
1 HEADER + 2 GTFE + 35 FDL + 30 PSB_ slot9 + 30 PSB_ slot13 + 30 PSB_ slot14 + 30 PSB_ slot15 + **85** GMT + 30 PSB_ slot19 + 30 PSB_ slot20 + 30 PSB_ slot21 + 1 TRAILER
3-5 Bunch-crossings per event mixed: GMT and FDL with 5bc/event, all PSB with 3bc/event
1 HEADER + 2 GTFE + 35 FDL + **85** GMT + 18 x7 PSBs = **249(=00F9) W64**

3 bx / 80MHz : S-LINK64 transfer time per event: 202 x 12.5 ns ~ 2525 ns = 2.5 us
5 bx / 80MHz: S-LINK64 transfer time per event: 334 x 12.5 ns ~ 4175 ns = 4.2 us

3 bx / 40 MHz Channel Link transfer times:
FDL: 21 W64 x 4 = 84 W16 ➔ *25ns ➔ 2.1 us
PSB: 18 W64 x 4 = 72 W16 ➔ *25ns ➔ 1.8 us
GMT: **51** W64 x 4 =192 W16 ➔ *25ns ➔ 4.8 us
5 bx / 40 MHz Channel Link transfer times:
FDL: 35 W64 x 4 = 140 W16 ➔ *25ns ➔ 3.5 us
PSB: 30 W64 x 4 = 120 W16 ➔ *25ns ➔ 3.0 us
GMT: **85** W64 x 4 = 320 W16 ➔ *25ns ➔ 8.0 us

3.7.12 GFTE HEADER WORD_1

3.7.12.1 BOARD_ID

BB10 0010 BOARD_ID w/r

bits 15:8: Board_ID of GTFE = VME slot number= 17dec=11hex

bits 7:0: length in bx =3 or 5 ➔ **default value = 1103**

3.7.12.2 SETUP_VERSION 31-0

BB10 0012 SETUP_VERSION_15_0 w/r // bits 15_0 //GFTE HeaderWord1

BB10 0014 SETUP_VERSION_31_16 w/r // bits 31_16 //GFTE HeaderWord1

➔ **default values = 0000 0000 …to be defined by DAQ group**

3.7.13 ACTIVE_BOARDS

BB10 0016 ACTIVE_BOARDS w/r //GFTE HeaderWord2

If bit xx =1 then the board is included in the event record.

GTFE board is always included. GTFE record length in DAQ chip = 2 W64 (3 in EVM chip)

BIT#	BOARD	SLOT# in crate	Remark	Record_length (W64)
0	FDL	10	Final Decision board	3*7=21 (5*7=35)
1	PSB_0	9	Techn.Triggers for FDL	3*6=18 (5*6=30)
2	PSB_1	13	Calo data for GTL	3*6=18 (5*6=30)
3	PSB_2	14	Calo data for GTL	3*6=18 (5*6=30)
4	PSB_3	15	Calo data for GTL	3*6=18 (5*6=30)
5	PSB_4	19	M/Q bits for GMT	3*6=18 (5*6=30)
6	PSB_5	20	M/Q bits for GMT	3*6=18 (5*6=30)
7	PSB_6	21	M/Q bits for GMT	3*6=18 (5*6=30)
8	GMT	18	Global Muon Trigger	16*3=48 (5*16=80)
9	*GMT_spare*	*18*	*Not used*	Not defined in DAQ record
10	*TCS_M*	*9*	*Not used*	Not defined in DAQ record
11	TIM	16	Record with test data	6 W64 words
12-15	*0 0 0 0 0*	*Not used*	*Nothing assigned*	

➔ **default value = 01FF …all boards included for a standard event record**
(without TIM)

ACTIVE_BOARDS = X'0000' → DAQ chip does not send any event.

3.7.14 SPY_FULL_LIMIT

BB10 0018 SPY_FULL_LIMIT w/r // bits 15_0 // Spy control word

!!! Is used by simulating mode only and should be set equal to the length of the event record minus 1 (because 1st word is in address = 0). SPY_FULL_LIMIT defines the end of the simulation event, when the Control Signal for the Trailer Word is applied.
In spy mode the memory is filled until the end.

3.7.15 TEST_MASK1, 2,3

BB10 001A TEST_MASK1 w/r // bits 15_0 ➔ **default value =8000**

BB10 001C TEST_MASK2 w/r // bits 15_0 ➔ **default value =8000**

BB10 001E TEST_MASK3 w/r // bits 15_0 ➔ **default value =8000**

Switch internal signal(s) to LEMO TEST outputs on the front panel.

Signals for the frontpanel LEMO connectors go through 2 FF (2x12.5ns) inside the FPGA and are then delayed by 50 Ohm drivers for about 5-6 ns. Signals for the DAQ_TEST(i) testpoints go through 1 FF (12.5ns) inside the FPGA and then immediately to the test point pins.
➔ Signals on DAQ_TEST(i) test-points appear about 18.5 ns before the signals on the frontpanel LEMO connectors if they are generated at the same time.
If more mask bits are set, then the 'OR' of the corresponding signals is displayed.
Since V001C: *status bits to EVM inserted.*
V0020, V0021(too_many_L1A)

	TEST_MASK1		**TEST_MASK2**		**TEST_MASK3**
bit	*SIGNALS*	*bit*	*SIGNALS*	*bit*	*SIGNALS*
15	**clk40**	15	**bcres_dlyed**	15	**L1A_int**
14	**L1A_int**	14	or_too_many_L1A	14	sel_din
13	vme_en	13	sel_trailer	13	sel_gtfe(1)
12	dtack	12	slink_wen	12	sel_gtfe(0)
11	warning	11	busy_priority	11	out_of_sync
10	or_warn50	10	or_warn75	10	or_full
9	res_orbitnr_int	9	busy	9	error
8	bc_error	8	end_of_event(8) GMT	8	wr_fifo(8)
7	bcres_int	7	end_of_event(7) PSB slot 21	7	wr_fifo (7)
6	res_evnr_fsm	6	end_of_event(6) PSB slot 20	6	wr_fifo (6)
5	l1res_fsm	5	end_of_event(5) PSB slot 19	5	wr_fifo (5)
4	warn_priority	4	end_of_event(4) PSB slot 15	4	wr_fifo (4)
3	ready_priority	3	end_of_event(3) PSB slot 14	3	wr_fifo (3)
2	en_crc_trailer	2	end_of_event(2) PSB slot 13	2	wr_fifo (2)
1	sl64_down	1	end_of_event(1) PSB slot 9	1	err_priority
0	sl64_full	0	end_of_event(0) FDL	0	out_of_sync_priority

Until V001B:

	TEST_MASK1		**TEST_MASK2**		**TEST_MASK3**
bit	*SIGNALS*	*bit*	*SIGNALS*	*bit*	*SIGNALS*
15	**clk40**	15	**bcres_dlyed**	15	**L1a_int**
14	sel_header	14	sel_err	14	sel_din
13	vme_en	13	sel_trailer	13	sel_gtfe(1)
12	dtack	12	slink_wen	12	sel_gtfe(0)
11	start_run	11	end_of_event(11) *not used*	11	sel_chan(11)
10	run_rop	10	end_of_event(10) *not used*	10	sel_chan(10)
9	res_orbitnr_int	9	end_of_event(9) *not used*	9	sel_chan(9)
8	bc_error	8	end_of_event(8) GMT	8	sel_chan(8)
7	bcres_int	7	end_of_event(7) PSB slot 21	7	sel_chan(7)
6	res_evnr_fsm	6	end_of_event(6) PSB slot 20	6	sel_chan(6)
5	l1res_fsm	5	end_of_event(5) PSB slot 19	5	sel_chan(5)
4	en_crc	4	end_of_event(4) PSB slot 15	4	sel_chan(4)
3	reset_crc	3	end_of_event(3) PSB slot 14	3	sel_chan(3)
2	en_crc_trailer	2	end_of_event(2) PSB slot 13	2	sel_chan(2)
1	sl64_down	1	end_of_event(1) PSB slot 9	1	sel_chan(1)
0	sl64_full	0	end_of_event(0) FDL	0	sel_chan(0)

3.8 STATUS REGISTERS

3.8.1 DAQ_CHIP_ID

BB10 0020 DAQ_CHIP_IDL -/r // bits 15_0 ➔ **default value = A032**
BB10 0022 DAQ_CHIP_IDH -/r // bits 31_16 ➔ **default value = 0001**

3.8.2 DAQ_VERSION_NR

BB10 0024 DAQ_VERSION_NR -/r // bits 15_0 Firmware Version nr.

3.8.3 DAQ_STATUS

BB10 0026 DAQ_STATUS -/r // General status with SLINK
Bit 15 -12: daq_status2tcs(3:0) // encoded status bits for TCS Trigger Control
0000 or 1111 = disconnected
0001 = warning
0010 = out of sync
0100 = busy
1000 = ready
1100 = error
1110 = bad code
Bit11: RUN_FF // = set by the START_RUN command from the TIM board
// = cleared by the STOP_RUN command from the TIM board
// = cleared by the CMD_PULSE(15) : = res_runff_vme …for tests
Bit 10: BC_ERROR // The flag is set whenever the external and internal BCReset disagree:
// Reasons: 1) wrong MAX_BC_NUMBER, 2.) bad BCRES signal
// The bc_error is cleared by command pulse (bit 4):' res_bc_error'
Bit 9: SPY_MEM_FULL // Spy memory is full
Bit 8: SLINK_FULL // SLINK cannot receive data anymore, ROP waits to continue
Bit 7: SLINK_DOWN // SLINK does not run
Bit 6: SL_RESV2 // explanation to be added
Bit 5: SL_RESV1
Bit 4: SL_RESV0
Bit 3: SL_LRL3 // explanation to be added
Bit 2: SL_LRL2
Bit 1: SL_LRL1
Bit 0: SL_LRL0

If the SLINK is not mounted then we will see: "8c7f"

- 8 => ready,
- c = 1100 => 1=don'tcare, 1=bcerror at begin(read bc_error counter to clear flag), 0= spy_mem is not full, 0=slink is not full when not connected;
- 7 = 0111 => 0=slink is not down when not connected, 111=sl_resv..not connected
- f = 1111 => sl_lrl(3:0) when slink is not connected

3.8.4 CHAN_FIFOS_FULL

BB10 0028 CHAN_FIFOS_FULL -/r // FULL flags of all channels
Bit = 1 ➔ Channel Link FIFO is full.

bit	FULL FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14

4	PSB_3	4		15
5	PSB_4	5		19
6	PSB_5	6		20
7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	GTFE	--	BCNR_FIFO	--

3.8.5 CHAN_FIFOS_WARN50

BB10 002A CHAN_FIFOS_WARN50 -/r // WARN flags of all channels

Bit =1 ➔ The FIFOs are filled up to the WARNING LEVEL (50%) ➔ A warning flag is sent via the EVM chip to the TCS board to reduce the trigger rate.

bit	WARN FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14
4	PSB_3	4		15
5	PSB_4	5		19
6	PSB_5	6		20
7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	GTFE	--	BCNR_FIFO	--

3.8.6 CHAN_FIFOS_EMPTY

BB10 002C CHAN_FIFOS_EMPTY -/r // EMPTY flags of all channels

Bit = 1 ➔ Channel Link FIFO is empty

bit	EMPTY FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14
4	PSB_3	4		15
5	PSB_4	5		19

6	PSB_5	6		20
7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	GTFE	--	BCNR_FIFO	--

3.8.7 CHAN_FIFOS_WARN75

BB10 002E CHAN_FIFOS_WARN75 -/r // ..of all channels

Bit =1 ➔ The FIFOs are filled up to the WARNING LEVEL (75%) ➔ A BUSY flag is sent via the EVM chip to the TCS board to inhibit triggers.

bit	TOO_LATE_FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14
4	PSB_3	4		15
5	PSB_4	5		19
6	PSB_5	6		20
7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	GTFE	--	BCNR_FIFO	--

3.8.8 CHAN_LINK_BAD_CODE

BB10 0030 CHAN_LINK_BAD_CODE -/r // Bad Header Code flags

Bit = 1 ➔ Channel Link sends undefined HEADER code (bits 27-24 of Channel Link data words). For example see Channel Link Formats chapter 5 in GT_Readout_format.pdf on GT-web page.

This is a spurious flag that will be seen only when the error becomes constant.

bit	BAD_CODE FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14
4	PSB_3	4		15
5	PSB_4	5		19
6	PSB_5	6		20

7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	--	--	--	--

3.8.9 CHAN_LINK_DATA_LOST

BB10 0032 CHAN_LINK_DATA_LOST -/r // ..of all channels

Bit = 1 ➔ Channel Link wants to write into a full FIFO.

This is a spurious flag that will be seen only when the TCS still is sending L1A trigger signals even when the Channel FIFOs are full. ➔ GTFE to TCS connection does not work correctly or an IGNORE ERROR bit has been set incorrectly.

bit	DATA_LOST FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14
4	PSB_3	4		15
5	PSB_4	5		19
6	PSB_5	6		20
7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	--	--	--	--

3.8.10 CHAN_LINK_OK

BB10 0034 CHAN_LINK_OK -/r // ..of all channels

Bit = 1 ➔ Channel Link header bits = A,D,C,D,E or 5(=idle)

bit	OK_FLAG of	Channel#	Remarks	SLOT
0	FDL	0		10
1	PSB_0	1		9
2	PSB_1	2		13
3	PSB_2	3		14
4	PSB_3	4		15
5	PSB_4	5		19
6	PSB_5	6		20

7	PSB_6	7		21
8	GMT	8		18
9	*GMT_spare*	*9*	*Not used*	*18*
10	*TCS_M*	*10*	*Not used*	*7*
11	*TIM*	*11*	*Test data*	*16*
12	--	--	--	--
13	--	--	--	--
14	--	--	--	--
15	--	--	--	--

3.8.11 STATE_MACHINE_STATUS

BB10 0036 STATE_MACHINE_STATUS -/r

STATE_MACHINE_STATUS word shows the present status of the **R**ead-**O**ut **C**ontroller (ROC) state machine that makes the event record.

- The ROC skips non active boards. In that case the EVENT_LENGTH register has to be programmed accordingly.
- If Channel Link data do not arrive within 511*12.5 ns = 6.387,5 ns then the ROC sends 'FFFF' instead and proceeds to the next channel.
- If the SLINK64 is not ready the ROC waits for an infinite time. Only a VME 'res_roc' command can reset the ROC to the IDLE state.
- According to the 'length_5bx' command register bit the ROC expects data form 3 or 5 bunchcrossings. If the record lengths of boards and the DAQ chip disagree bad data will be sent.

Before and during an event:

8000 = IDLE status because run_rop =0

// Since V001F the run_rop FF is always =1 and IDLE is a transient state.

0000 = ROC is transmitting an event (passing through transient states).

Begin of an event:

10FF = waiting for SLINK64 before waiting for a new event

// **res_roc_vme** (cmd-pulse) or **Resync/L1reset** (Bgo from TCS**)** return the ROC-state machine to IDLE status.

// If the SLINK is permanently either down or full the DAQ-software has to restart the S-link hardware on the PC-side and/or the GTFE-software has to send a **res_slink_vme** (cmd-pulse) signal to the S-Link board that is mounted on the GTFE-board.

20FF = waiting for a new event

The ROC-sm is waiting for a L1A signal and since V0021 is waiting also until at least one of the active board has sent an event record.

4000 = FIFO FULL error,

// A FIFO of an active board is full. L1Reset(=Resync) or res_roc_vme reset the ROC-sm to IDLE status.

8001 = RESYNC/L1RESET cycle is active. Transient status is active for about 1600 ns.

FDL =channel 0 (vme slot 10)

1000 = waiting for SLINK, no time-out, return to IDLE by res_roc_vme(cmd-pulse)

2000 = waiting for FDL data

The ROC-sm waits for 512*12.5 ns= 6400 ns. If no FDL data arrive (FIFO is empty), then it sends an error record (data= x"FFFF…").

3000 = starting with FDL data

PSB0 =channel 1 //Technical Triggers
1001 = waiting for SLINK //same behavior as in channel 0
2001 = waiting for PSB0 data (vme slot 9) //same behavior as in channel 0
3001 = starting with PSB0 data
PSB1 =channel 2 // Calo data
1002 = waiting for SLINK
2002 = waiting for PSB1 data (vme slot 13)
3002 = starting with PSB1 data
PSB2 =channel 3 // Calo data
1003 = waiting for SLINK
2003 = waiting for PSB2 data (vme slot 14)
3003 = starting with PSB2 data
PSB3 =channel 4 // Calo data
1004 = waiting for SLINK
2004 = waiting for PSB3 data (vme slot 15)
3004 = starting with PSB3 data
PSB4 =channel 5 // M/Q bits
1005 = waiting for SLINK
2005 = waiting for PSB4 data (vme slot 19)
3005 = starting with PSB4 data
PSB5 =channel 6 // M/Q bits
1006 = waiting for SLINK
2006 = waiting for PSB5 data (vme slot 20)
3006 = starting with PSB5 data
PSB6 =channel 7 // M/Q bits
1007 = waiting for SLINK
2007 = waiting for PSB6 data (vme slot 21)
3007 = starting with PSB6 data
GMT =channel 8 // Muons
2008 = waiting for GMT data (vme slot 18) // same behavior as in channel 0
1081 = waiting for SLINK before sending data of 1st BC (bunch crossing)
1082 = waiting for SLINK before sending data of 2nd BC (bunch crossing)
1092 = waiting for SLINK before sending error data of 2nd BC (bunch crossing)
1083 = waiting for SLINK before sending data of 3rd BC (bunch crossing)
1093 = waiting for SLINK before sending error data of 3rd BC (bunch crossing)
1084 = waiting for SLINK before sending data of 4th BC (bunch crossing)
1094 = waiting for SLINK before sending error data of 4th BC (bunch crossing)
1085 = waiting for SLINK before sending data of 5th BC (bunch crossing)
1095 = waiting for SLINK before sending error data of 5th BC (bunch crossing)
3008 = starting with GMT data // same behavior as in channel 0
TIM =channel 11 // TIM board test data
1011 = waiting for SLINK // same behavior as in channel 0
2011 = waiting for **TIM** data (vme slot 16) // same behavior as in channel 0
3011 = starting with TIM data // same behavior as in channel 0

3.8.12 SIM_SPY_ADDRESS

BB10 0038 SIM_SPY_ADDRESS
… shows the actual value of the spy address. Can be used to check how many events have been spied since start_simspy command pulse. This value might be useful to see if a L1A has arrived.

3.9 MONITORING COUNTERS

BB10 0100	CNTR_WAITING_FOR_FDL_DAQ_L	-/r
BB10 0102	CNTR_WAITING_FOR_FDL_DAQ_H	-/r
BB10 0104	CNTR_FIFO_FULL_FDL_DAQ_L	-/r
BB10 0106	CNTR_FIFO_FULL_FDL_DAQ_H	-/r
BB10 0108	CNTR_FIFO_WARN75_FDL_DAQ_L	-/r
BB10 010A	CNTR_FIFO_WARN75_FDL_DAQ_H	-/r
BB10 010C	CNTR_FIFO_WARN50_FDL_DAQ_L	-/r
BB10 010E	CNTR_FIFO_WARN50_FDL_DAQ_H	-/r
BB10 0110	CNTR_WAITING_FOR_PSB9_L	-/r
BB10 0112	CNTR_WAITING_FOR_PSB9_H	-/r
BB10 0114	CNTR_FIFO_FULL_PSB9_L	-/r
BB10 0116	CNTR_FIFO_FULL_PSB9_H	-/r
BB10 0118	CNTR_FIFO_WARN75_PSB9_L	-/r
BB10 011A	CNTR_FIFO_WARN75_PSB9_H	-/r
BB10 011C	CNTR_FIFO_WARN50_PSB9_L	-/r
BB10 011E	CNTR_FIFO_WARN50_PSB9_H	-/r
BB10 0120	CNTR_WAITING_FOR_PSB13_L	-/r
BB10 0122	CNTR_WAITING_FOR_PSB13_H	-/r
BB10 0124	CNTR_FIFO_FULL_PSB13_L	-/r
BB10 0126	CNTR_FIFO_FULL_PSB13_H	-/r
BB10 0128	CNTR_FIFO_WARN75_PSB13_L	-/r
BB10 012A	CNTR_FIFO_WARN75_PSB13_H	-/r
BB10 012C	CNTR_FIFO_WARN50_PSB13_L	-/r
BB10 012E	CNTR_FIFO_WARN50_PSB13_H	-/r
BB10 0130	CNTR_WAITING_FOR_PSB14_L	-/r
BB10 0132	CNTR_WAITING_FOR_PSB14_H	-/r
BB10 0134	CNTR_FIFO_FULL_PSB14_L	-/r
BB10 0136	CNTR_FIFO_FULL_PSB14_H	-/r
BB10 0138	CNTR_FIFO_WARN75_PSB14_L	-/r
BB10 013A	CNTR_FIFO_WARN75_PSB14_H	-/r
BB10 013C	CNTR_FIFO_WARN50_PSB14_L	-/r
BB10 013E	CNTR_FIFO_WARN50_PSB14_H	-/r
BB10 0140	CNTR_WAITING_FOR_PSB15_L	-/r
BB10 0142	CNTR_WAITING_FOR_PSB15_H	-/r
BB10 0144	CNTR_FIFO_FULL_PSB15_L	-/r
BB10 0146	CNTR_FIFO_FULL_PSB15_H	-/r
BB10 0148	CNTR_FIFO_WARN75_PSB15_L	-/r
BB10 014A	CNTR_FIFO_WARN75_PSB15_H	-/r
BB10 014C	CNTR_FIFO_WARN50_PSB15_L	-/r
BB10 014E	CNTR_FIFO_WARN50_PSB15_H	-/r
BB10 0150	CNTR_WAITING_FOR_PSB19_L	-/r
BB10 0152	CNTR_WAITING_FOR_PSB19_H	-/r
BB10 0154	CNTR_FIFO_FULL_PSB19_L	-/r
BB10 0156	CNTR_FIFO_FULL_PSB19_H	-/r
BB10 0158	CNTR_FIFO_WARN75_PSB19_L	-/r

BB10 015A CNTR_FIFO_WARN75_PSB19_H -/r
BB10 015C CNTR_FIFO_WARN50_PSB19_L -/r
BB10 015E CNTR_FIFO_WARN50_PSB19_H -/r

BB10 0160 CNTR_WAITING_FOR_PSB20_L -/r
BB10 0162 CNTR_WAITING_FOR_PSB20_H -/r
BB10 0164 CNTR_FIFO_FULL_PSB20_L -/r
BB10 0166 CNTR_FIFO_FULL_PSB20_H -/r
BB10 0168 CNTR_FIFO_WARN75_PSB20_L -/r
BB10 016A CNTR_FIFO_WARN75_PSB20_H -/r
BB10 016C CNTR_FIFO_WARN50_PSB20_L -/r
BB10 016E CNTR_FIFO_WARN50_PSB20_H -/r

BB10 0170 CNTR_WAITING_FOR_PSB21_L -/r
BB10 0172 CNTR_WAITING_FOR_PSB21_H -/r
BB10 0174 CNTR_FIFO_FULL_PSB21_L -/r
BB10 0176 CNTR_FIFO_FULL_PSB21_H -/r
BB10 0178 CNTR_FIFO_WARN75_PSB21_L -/r
BB10 017A CNTR_FIFO_WARN75_PSB21_H -/r
BB10 017C CNTR_FIFO_WARN50_PSB21_L -/r
BB10 017E CNTR_FIFO_WARN50_PSB21_H -/r

BB10 0180 CNTR_WAITING_FOR_GMT_L -/r
BB10 0182 CNTR_WAITING_FOR_GMT_H -/r
BB10 0184 CNTR_FIFO_FULL_GMT_L -/r
BB10 0186 CNTR_FIFO_FULL_GMT_H -/r
BB10 0188 CNTR_FIFO_WARN75_GMT_L -/r
BB10 018A CNTR_FIFO_WARN75_GMT_H -/r
BB10 018C CNTR_FIFO_WARN50_GMT_L -/r
BB10 018E CNTR_FIFO_WARN50_GMT_H -/r

BB19 0190 DAQ_MONITORING_TIME _L -/r
BB19 0192 DAQ_MONITORING_TIME _H -/r
BB19 0194 CNTR_EVENTS_TO_DAQ_SLINK_L -/r
BB19 0196 CNTR_EVENTS_TO_DAQ_SLINK_H -/r
BB19 0198 CNTR_FFF_RECORDS_DAQ_L -/r
BB19 019A CNTR_FFF_RECORDS_ DAQ_H -/r
BB19 019C CNTR_DAQ_SLINK_DOWN_L -/r
BB19 019E CNTR_DAQ_SLINK_DOWN_H -/r

BB10 01A0 CNTR_DAQ_SLINK_FULL_L -/r
BB10 01A2 CNTR_DAQ_SLINK_FULL_H -/r
BB10 01A4 CNTR_WAITING_FOR_DAQ_SLINK_L -/r
BB10 01A6 CNTR_WAITING_FOR_DAQ_SLINK_H -/r

MONITORING_TIME counts the time in BC's.
CNTR_DAQ_SLINK _DOWN is incremented while SLINK DOWN FLAG is active.
CNTR_DAQ_SLINK _FULL is incremented while SLINK FULL FLAG is active.
CNTR_WAITING_FOR_DAQ_SLINK is incremented while the Readout Controller State Machine (ROC) is in the status 'waiting _for_SLINK'.

CNTR_EVENTS_TO_SLINK counts the number of events sent to the SLINK64. The 'select_trailer' increments the counter.
CNTR_FFF_RECORDS counts the number of "fff...fff records in events sent to the SLINK64.

4 Counters per board sending event records:

CNTR_WAITING_FOR_xxx is incremented while the Readout Controller State Machine (ROC) is in the status 'waiting_for_xxx' board.
CNTR_FIFO_FULL _xx is incremented while the FIFO becomes full.
CNTR_FIFO_WARN75_xx is incremented while the FIFO becomes 75%full.
CNTR_FIFO_WARN50_xx is incremented while the FIFO becomes 50%full.

3.10 LED

Front panel LED shows errors when flashed red:

-- CHANNEL_LINK_TOO_LATE FLAGS
-- BAD_HEADER FLAGS
-- CHANNEL_LINK_DOWN FLAGS
-- BC_ERROR // normally at begin
-- SLINK DOWN FLAG

4 ROP_EVM chip

4.1 Updates

Version V101B:

BB20 0004 EVM_CMD_REG (12) := RESET_TTCRX …removed
BB20 0044 SEL_PHASE (5) := EN_TURN_CLK
BB20 0040 EVM_CMD_PULSE(12): res_ttc_err_cntrs … new!!
EVM_CMD_PULSE(13): REFRESH_MON_COUNTERS … new!!

New addresses:

BB20 005E BST_STROBE_CNTRS /r
BB20 004A EVM_TEST_MASK4 w/r // bits 15_0
Include Mask bits into programs!!!

Version V0019:

Event Length Calculator replaces 'event_length_register'.

Version V0018:

New addresses:
BB10 0058 SIM_SPY_ADDRESS -/r // actual value of spy address (V0017)
BB10 005A TTC_OFF_ERRORS -/r // TTCrx 'not ready' errors
BB10 005C TTC_DB_ERRORS -/r // TTCrx 'double bit' errors

Counters show TTCrx errors.

BST data: Asynchronous Fifo stores BST data with TTCrx clock.(CLOCKL1ACCEPT).
Text corrections:
BST_UPDATE_DELAY & BST_APPLY_DELAY: max value decreased to "0DEA"

Version V0017a:

POWER UP values and VersionNr defined in rop_pkg.vhd
sim_spy_mem with power-up values: 5555, aaaa defined in .coe file that is stored in rop_chip_lib\hdl

Version V0017: → VME logic updated.

Reset procedure for TTCrx chip is implemented in VME chip:

VME chip sends '**Reset_B**' as negative active pulse with 50 us length via 'NIRQ_FR_EVM' trace, which is forwarded to the TTCrx.
TTCrx returns **TTC_READY**, which is sent inverted via pin 'NBERR_EVM' to the VME chip.

Version V0016: TTC_READY from TTCrq is sent on trace NBERR_EVM to VME-chip (to enable a I2C-sequence for setting a register on TTCrq).
Version V0015: TCS FIFO increased ('chan_recvr_gmt' module used) to buffer more events while waiting for GMT-events. The GT runs now with 200 kHz random trigger without dead time.
Version V0014: EVM_IGNORE_ERRORS: Bit10: = 1 ignore DAQ_STATUS
BB10 0058 SIM_SPY_ADDRESS -/r
ROC: logic for get event type changed
Additional Monitoring Counters added,
chan_recvr: error in empty counter corrected
Version V0013: Monitoring Counters added, 'refresh_mon_cntrs' command pulse
Version V0012: Reset_B outff is '1' after power-up or after reset, to avoid resetting the TTCrx chip when a "RESET_EVM' has been received from the VME chip
Version V0011: New BST Format accepted;
SEL_PHASE(4) = 1 … adds 52 BST bytes to the EVM event record
SEL_PHASE(4) = 0 … adds 30 BST bytes to the EVM event record (old format)
Event record extended !!!
EVM_CMD_PULSE bit12 removed
EVM_CMD_REG bit12 RESET_TTCRX ← new meaning

Version V0010: …..
Version V000F: select phase of Channel Link data from TCS and FDL by
EVM_CMD_REG: Bit15: SELECT_FDL_PHASE1
Bit14: SELECT_TCS_PHASE1

Version V000E: compatibel to rop_daq_chip_V0023,
new testmasks, roc changed, warn50% and 75% flags, too_many_L1A flags…
Version V000B: New SPY memories for Channel Link data
Version V000A: chan_link_ok …status register, SM_STATUS…values also for transient states to find unknown errors.
Version V0006: accepts TCS_V7 data (EVM_CMD_REG(12))
Version V0003: EVM testmask2,3 for OSC_E changed

4.2 General information

Configuration file: Size for XC2V2000 chip

- Bit file 1,3 Mbytes
- 2 MCS files: 1,5 + 1,0 Mbytes
- SVF file (JTAG): 10 Mbytes

Required RAM blocks: XC2V2000 (56 RAMB),

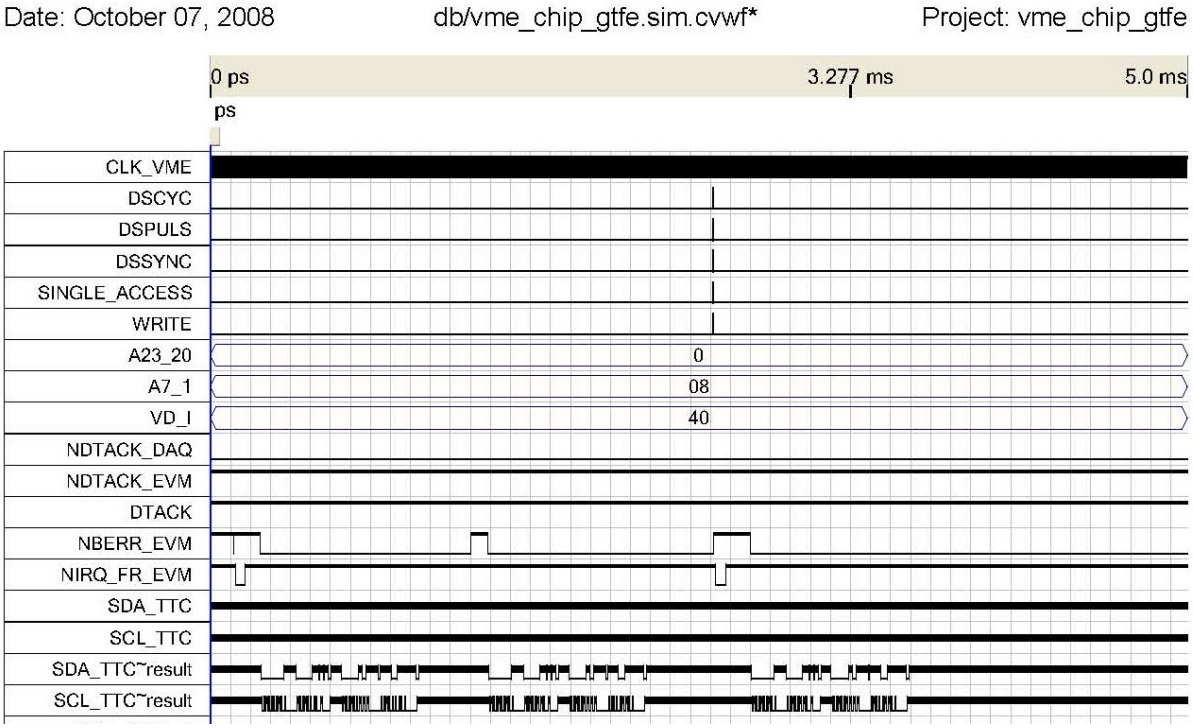
2 Board_receiver FIFOs each 2 RAMB	➔	4 RAMB to contain >20 3bx_events
1 board receiver FIFOs for GMT 4 RAMB	➔	4 RAMB to contain >20 3bx_events
1 bcnr FIFO	➔	1 RAMB
SIMSPY for SLINK: (32k x 16)	➔	32 RAMB to contain ~40 3bx_events
		total ➔➔ 41 RAMB

Size of 3bx event = 199 W64 ~800 W16

4.3 Power-up sequence

A power-up-sequence is implemented, which sends a RESET_B low active pulse (~50us, from VME-chip, over trace NIRQ_FR_EVM via EVM-chip [V0017 or higher]) to TTCrx-chip when TTCREADY is active.
RESET_B causes a reset-cycle on TTCrx-chip, which means that TTCREADY becomes inactive. At the end of the RESET_B pulse, the TTCrx-chip sets TTCREADY (if there is no error). Receiving this pos. edge of TTCREADY on VME-chip (via NBERR_EVM trace from EVM-chip, neg. active) an I2C-bus sequence (traces SDA and SCL) is started (on VME-chip), to set the control-register on the TTCrx-chip to 0xF3, which means that outputs for SUBADDR, DOUT_STROBE and DOUT are enabled to get "BST-data" on EVM-chip.
There is an option to send a RESET_B via VME, therefore one has to set 0x40 in the command-pulse-register on the VME-chip.
The following picture shows three sequences:

1. After power-up, detecting TTCREADY active (NBERR_EVM, neg. active) causes RESET_B (NIRQ_FR_EVM, neg. active, ~50us) followed by an I2C-bus sequence (SDA_TTC~result and SCL_TTC~result).
2. Every following pos.edge on TTCREADY (NBERR_EVM, neg. edge) causes an I2C-bus sequence (SDA_TTC~result and SCL_TTC~result).
3. RESET_B via VME.



4.4 EVM chip overview VME Addresses

A31-A24: = 'BB' = base address
A23-20: = 0001 …DAQ chip
A23-20: = 0010 …EVM chip

23-20	19-16	15-12	11	10	9	8	7	6	5	4	3	2	1	0
0010	0000	0000	0	Register space (1kx16 readback)										
0010	0000	0000	1	STATUS REGISTERS										
0010	0001	SIM SPY MEMORY 32kx16bits												
0010	0010	0000	0	TCS-spy memory 1kx16										

0010	0010	0000	1	FDL-spy memory 1kx16

4.5 EVM_SIM_SPY MEMORY

BB21 0000 EVM_SIM_SPY_MEMORY (32kx16)

The SIM_SPY memory is implemented as Dual Port Memory(DPM). On one side VME accesses the SIM_SPY memory as 32k x 16 bit. On the other side the memory is configured as 8k x 64 bit memory to spy data going to the SLINK64 or to send simulation data to the SLINK64.

VME side 16 bits	← →	SLINK side 64 bits
Addr 0: bits 15-0	← →	Adr0: bits 15-0
Addr 1: bits 15-0	← →	Adr0: bits 31 -16
Addr 2: bits 15-0	← →	Adr0: bits 47 - 32
Addr 3: bits 15-0	← →	Adr0: bits 63 - 48
Addr 4 bits 15-0	← →	Adr1 bits 15-0
Addr 5 bits 15-0	← →	Adr1 bits 31 -16
Addr 6 bits 15-0	← →	Adr1 bits 47 - 32
Addr 7 bits 15-0	← →	Adr1 bits 63 - 48

4.6 TCS and FDL_SPY memory

BB22_0000+addr TCS_SPY_MEMORY
BB22_0800+addr FDL_SPY_MEMORY

Spy memories for TCS and FDL channel-link data.
Readout via VME.

VME side:	Size:	1kx16bits	write/read access
Channel Link side:	Size:	512x32 bits	write only

➔ One Channel Link word is stored in 2 consecutive VME words.
The same words will be written into the spy-memories and into the Channel-Link FIFOs.

SPY-procedure:

- Setup the boards for either for a readout test or for normal data taking, but suppress any L1A.
- Start the RUN
- Send the command pulse 'res_chan_spy_addr' to reset the spy address counters of both Channel_Link Spy-memories so that spying of the next event will start with the first address. (See EVM_CMD_PULSE).
- Send a L1A signal.
- Read the TCS_SPY_MEMORY and FDL_SPY_MEMORY and save the data in a file.
- …repeat last 3 points

4.7 Register overview

Setup registers

BB20 0000	EVM_BCRES_DELAY	w/r
BB20 0002	EVM_SHOW_SLINK_BITS	w/r // mask register to show S-Link64 bits
BB20 0004	EVM_CMD_REG	w/r // sim_mode, enable bits,

			// set SLINK mode etc.
BB20 0006	EVM_MAX_BC_NUMBER	w/r	//=orbit length -1 = **0DEB**
BB20 0008	EVM_INITIAL_CRC	w/r	// bits 15_0 // initial CRC value =**FFFF**
BB20 000A	EVM_CMS_HEADER19_4	w/r	//CMS HeaderWord1(19:4)
	bits15:4:= GT-identifier in EVM software: 813 =32D(hex)		
	bits 3:0:= FORMAT_VERSION currently =0000 ➔ ***default value = 32D0***		
BB20 000C	EVM_EVENT_LENGTH_15_0	w/r	// bits 15_0 : 24 bit length
BB20 000E	EVM_EVENT_LENGTH_31_16	w/r	// bits 31_16 : 24 bit length
	// optional: Hardware Adder using ACTIVE_BOARDS		
BB20 0010	EVM_BOARD_ID	/r	
	// bits 15:8: Board_ID of GTFE = VME slot number= 17dec=11hex		
	// bits 7:0: length in bx =3 or 5 ➔ ***default value = 1103***		
BB20 0012	EVM_SETUP_VERSION_15_0	w/r	// bits15_0 //GFTE HeaderWord1
BB20 0014	EVM_SETUP_VERSION_31_16	w/r	// bits31_16//GFTE HeaderWord1
BB20 0016	EVM_ACTIVE_BOARDS	w/r	//GFTE HeaderWord2
BB20 0018	EVM_SPY_FULL_LIMIT	w/r	// bits 15_0 // Spy control word
BB20 001A	EVM_TEST_MASK1	w/r	// bits 15_0
BB20 001C	EVM_TEST_MASK2	w/r	// bits 15_0
BB20 001E	EVM_TEST_MASK3	w/r	// bits 15_0
Status words read only			
BB20 0020	EVM_CHIP_IDL	-/r	// bits 15_0
BB20 0022	EVM_CHIP_IDH	-/r	// bits 31_16
BB20 0024	EVM_VERSION_NR	-/r	// bits 15_0 Firmware Version nr.
BB20 0026	EVM_STATUS	-/r	// General status with SLINK
BB20 0028	EVM_CHAN_FIFOS_FULL	-/r	// FULL flags of all channels
BB20 002A	EVM_CHAN_FIFOS_WARN50	-/r	// WARN flags of all channels
BB20 002C	EVM_CHAN_FIFOS_EMPTY	-/r	// EMPTY flags of all channels
BB20 002E	EVM_CHAN_FIFOS_WARN75	-/r	// ..of all channels
BB20 0030	EVM_CHAN_LINK_BAD_CODE	-/r	// Bad Header Code flags
BB20 0032	EVM_CHAN_LINK_DATA_LOST	-/r	// ..of all channels
BB20 0034	EVM_CHAN_LINK_OK	-/r	// ..of all channels
BB20 0036	EVM_STATE_MACHINE_STATUS	-/r	//actual status of ROP state machine
BB20 0038	PHASE_CNTR_P3_0	-/r	//
BB20 003A	PHASE_CNTR_01	-/r	//
BB20 003C	PHASE_CNTR_12	-/r	//
BB20 003E	PHASE_CNTR_23	-/r	//
Write registers			
BB20 0040	EVM_CMD_PULSE	w/	//
BB20 0042	EVM_IGNORE_ERR_FOR_TCS	w/r	// bits 15_0
BB20 0044	SEL_PHASE	w/r	// bits 3_0 used
BB20 0046	BST_UPDATE_DELAY	w/r	// bits 15_0
BB20 0048	BST_APPLY_DELAY	w/r	// bits 15_0
BB20 004A	**EVM_TEST_MASK4**	**w/r**	**// bits 15_0**
BB20 004C	free		
BB20 004E	free		

BCNR at different times

BB20 0050	BCNR_BST1	-/r	// bits 15_0
BB20 0052	BCNR_BST_LAST	-/r	// bits 15_0
BB20 0054	BCNR_UPDATE	-/r	// bits 15_0
BB20 0056	BCNR_APPLY	-/r	// bits 15_0

BB20 0058	SIM_SPY_ADDRESS	-/r	// actual value of spy address
BB20 005A	**TTC_OFF_ERRORS**	**-/r**	**// TTCrx 'not ready' errors**
BB20 005C	**TTC_DB_ERRORS**	**-/r**	**// TTCrx 'double bit' errors**
BB20 005E	**BST_STROBE_CNTRS**	**-/r**	**// TTCrx strobe counters**

BST words read only

BB20 0060	GPS_TIME0	-/r	// GPS 15 – 0 // bst-byte 1, 0
BB20 0062	GPS_TIME1	-/r	// GPS 31 – 16// bst-byte 3, 2
BB20 0064	GPS_TIME2	-/r	// GPS 47 - 32 // bst-byte 5, 4
BB20 0066	GPS_TIME3	-/r	// GPS 63 – 48// bst-byte 7, 6
BB20 0068	ACQU_TRIGGERS9_8	-/r	// // bst-byte 9, 8
BB20 006A	ACQU_TRIGGERS11_10	-/r	// // bst-byte 11,10
BB20 006C	ACQU_TRIGGERS13_12	-/r	// // bst-byte 13, 12
BB20 006E	ACQU_TRIGGERS15_14	-/r	// // bst-byte 15, 14
BB20 0070	BST_STATUS_ABBI_DIAGN	-/r	// bst-byte 17, 16
BB20 0072	TURN_NRL	-/r	// bits 15_0 // bst-byte 19, 18
BB20 0074	TURN_NRH	-/r	// bits 31_16 // bst-byte 21, 20
BB20 0076	LHC_FILL_NRL	-/r	// bits 15_0 // bst-byte 23, 22
BB20 0078	LHC_FILL_NRH	-/r	// bits 31_16 // bst-byte 25, 24
BB20 007A	MACHINE_MODE	-/r	// // bst-byte 27, 26
BB20 007C	PARTICLE_TYPE	-/r	// // bst-byte 29, 28
BB20 007E	BEAM_MOMENTUM	-/r	// // bst-byte 31, 30
BB20 0080	INTENSITY_RING1_L	-/r	// // bst-byte 33, 32
BB20 0082	INTENSITY_RING1_H	-/r	// // bst-byte 35, 34
BB20 0084	INTENSITY_RING2_L	-/r	// // bst-byte 37, 36
BB20 0086	INTENSITY_RING2_H	-/r	// // bst-byte 39, 38
BB20 0088	INT_BCTF_RING1_L	-/r	// // bst-byte 41, 40
BB20 008A	INT_BCTF_RING1_H	-/r	// // bst-byte 43, 42
BB20 008C	INT_BCTF_RING2_L	-/r	// // bst-byte 45, 44
BB20 008E	INT_BCTF_RING2_H	-/r	// // bst-byte 47, 46
BB20 0090	BC_BCTF_RING1	-/r	// // bst-byte 49, 48
BB20 0092	BC_BCTF_RING2	-/r	// // bst-byte 51, 50
BB20 0094	CODE_FOR_BST_SOURCE	-/r	
BB20 0096	not used		
BB20 0098	not used		
BB20 009A	not used		
BB20 009C	not used		
BB20 009E	not used		

BST SIMULATION registers

BB20 00A0	SIM_GPS_TIME0	w/r	// GPS 15 – 0	// bst-byte 1,0
BB20 00A2	SIM_GPS_TIME1	w/r	// GPS 31 – 16//	bst-byte 3,2
BB20 00A4	SIM_GPS_TIME2	w/r	// GPS 47 - 32 //	bst-byte 5,4
BB20 00A6	SIM_GPS_TIME3	w/r	// GPS 63 – 48//	bst-byte 7,6
BB20 00A8	SIM_ACQU_TRIGGERS9_8	w/r	//	// bst-byte 9,8
BB20 00AA	SIM_ACQU_TRIGGERS11_10	w/r	//	// bst-byte 11,10
BB20 00AC	SIM_ACQU_TRIGGERS13_12	w/r	//	// bst-byte 13, 12
BB20 00AE	SIM_ACQU_TRIGGERS15_14	w/r	//	// bst-byte 15, 14
BB20 00B0	SIM_BST_STATUS_ABBI_DIAGN	w/r		// bst-byte 17, 16
BB20 00B2	SIM_TURN_NRL	w/r	// bits 15_0	// bst-byte 19,18
BB20 00B4	SIM_TURN_NRH	w/r	// bits 31_16	// bst-byte 21, 20

BB20 00B6	SIM_LHC_FILL_NRL	w/r	// bits 15_0	// bst-byte 23, 22
BB20 00B8	SIM_LHC_FILL_NRH	w/r	// bits 31_16	// bst-byte 25, 24
BB20 00BA	SIM_MACHINE_MODE	w/r	//	// bst-byte 27, 26
BB20 00BC	SIM_PARTICLE_TYPE	w/r	//	// bst-byte 29, 28
BB20 00BE	SIM_BEAM_MOMENTUM	w/r	//	// bst-byte 31, 30
BB20 00C0	SIM_INTENSITY_RING1_L	w/r	//	// bst-byte 33, 32
BB20 00C2	SIM_INTENSITY_RING1_H	w/r	//	// bst-byte 35, 34
BB20 00C4	SIM_INTENSITY_RING2_L	w/r	//	// bst-byte 37, 36
BB20 00C6	SIM_INTENSITY_RING2_H	w/r	//	// bst-byte 39, 38
BB20 00C8	SIM_INT_BCTF_RING1_L	w/r	//	// bst-byte 41, 40
BB20 00CA	SIM_INT_BCTF_RING1_H	w/r	//	// bst-byte 43, 42
BB20 00CC	SIM_INT_BCTF_RING2_L	w/r	//	// bst-byte 45, 44
BB20 00CE	SIM_INT_BCTF_RING2_H	w/r	//	// bst-byte 47, 46
BB20 00D0	SIM_BC_BCTF_RING1	w/r	//	// bst-byte 49, 48
BB20 00D2	SIM_BC_BCTF_RING2	w/r	//	// bst-byte 51, 50
BB20 00D4	SIM_CODE_FOR_BST_SOURCE	w/r	// =”DDDD” for simulated data	

Monitoring counters

BB20 0100	MONITORING_TIME_L	-/r	
BB20 0102	MONITORING_TIME_H	-/r	
BB20 0104	CNTR_SLINK_DOWN_L	-/r	
BB20 0106	CNTR_SLINK_DOWN_H	-/r	
BB20 0108	CNTR_SLINK_FULL_L	-/r	
BB20 010A	CNTR_SLINK_FULL_H	-/r	
BB20 010C	CNTR_WAITING_FOR_SLINK_L	-/r	
BB20 010E	CNTR_WAITING_FOR_SLINK_H	-/r	
BB20 0110	CNTR_WAITING_FOR_FDL_L	-/r	
BB20 0112	CNTR_WAITING_FOR_FDL_H	-/r	
BB20 0114	CNTR_FIFO_FULL_FDL_L		-/r
BB20 0116	CNTR_FIFO_FULL_FDL_H		-/r
BB20 0118	CNTR_FIFO_WARN75_FDL_L	-/r	
BB20 011A	CNTR_FIFO_WARN75_FDL_H	-/r	
BB20 011C	CNTR_FIFO_WARN50_FDL_L	-/r	
BB20 011E	CNTR_FIFO_WARN50_FDL_H	-/r	
BB20 0120	CNTR_TOO_MANY_L1A_FDL_L	-/r	
BB20 0122	CNTR_TOO_MANY_L1A_FDL_H	-/r	
BB20 0124	CNTR_WAITING_FOR_TCS_L	-/r	
BB20 0126	CNTR_WAITING_FOR_TCS_H	-/r	
BB20 0128	CNTR_FIFO_FULL_ TCS_L	-/r	
BB20 012A	CNTR_FIFO_FULL_ TCS_H	-/r	
BB20 012C	CNTR_FIFO_WARN75_ TCS_L	-/r	
BB20 012E	CNTR_FIFO_WARN75_ TCS_H	-/r	
BB20 0130	CNTR_FIFO_WARN50_ TCS_L	-/r	
BB20 0132	CNTR_FIFO_WARN50_ TCS_H	-/r	
BB20 0134	CNTR_TOO_MANY_L1A_ TCS_L		-/r
BB20 0136	CNTR_TOO_MANY_L1A_ TCS_H		-/r
BB20 0138	CNTR_EVENTS_TO_SLINK_L	-/r	
BB20 013A	CNTR_EVENTS_TO_SLINK_H	-/r	
BB20 013C	CNTR_FFF_RECORDS _L	-/r	
BB20 013E	CNTR_FFF_RECORDS _H	-/r	

4.8 Register descriptions

4.8.1 EVM_CMD_PULSE

BB20 0040 EVM_CMD_PULSE w/ //
Bit 15: res_runff_vme // resets RUN_FF and stops ROC-state machine
Bit 14: res_chan_spy_addr // resets address counters of Channel_Link Spy-memories
Bit 13: REFRESH_MON_COUNTERS // moves counter contents into registers and clears the monitoring counters.
Bit 12: res_ttc_err_cntrs // reset TTC error counters and DQ counters --V101B
Bit 11: L1A_vme // simulate a L1A pulse for a standalone test
Bit 10: res_roc_vme // reset ROC state machine by VME
Bit 9: res_slink_vme // reset SLINK by VME by 800 ns pulse //See also evm_cmd_reg(13) to reset SLINK alternatively by software defined pulse length.
Bit 8: stop_spying_simulating
Bit 7: start_spying_simulating
Bit 6: stop_rop_vme
Bit 5: start_rop_vme
Bit 4: res_bc_error
Bit 3: ResEvnr_vme
Bit 2: ResOrbnr_vme
Bit 1: BCRes_vme
Bit 0: L1Res_vme
//ROC = **R**ead**o**ut **C**ontroller is implemented as finite state machine

4.8.2 EVM_IGNORE_ERR_FOR_TCS

BB20 0042 EVM_IGNORE_ERR_FOR_TCS w/r // bits 15_0
If one of the ignore bits is set then the respective flags of active Channel Links are not transmitted to the TCS Trigger Control board (via the FDL board).
Bit15-11 not used
Bit10: = 1 ignore DAQ_STATUS
Bit9: = 1 ignore ROC_BUSY FLAG → busy
Bit8: = 1 ignore ROC_ERROR FLAG → error
Bit7: = 1 ignore SLINK DOWN FLAG → error
Bit6: = 1 ignore BC_ERROR → error
Bit5: = 1 ignore FIFO WARN50 FLAGS → warning //since V000E
Bit4: = 1 ignore FIFO WARN75 FLAGS → busy //since V000E
Bit3: = 1 ignore DATA_LOST FLAGS → not included in out_of_sync to TCS
Bit2: = 1 ignore BAD_HEADER FLAGS → not included in error to TCS
Bit1: = 1 ignore TOO_MANY_L1A_IN_CHAIN → busy //since V000E
Bit0: = 1 ignore FIFO FULL FLAGS → out_of_sync

4.8.3 SEL_PHASE

BB20 0044 SEL_PHASE w/r // bits 3_0 used

	15 - 12	11 - 8	7 - 4	3 - 0
SEL_PHASE	0000	0000	0, 0, EN_TURN_CLK, BST52	SELECT_PHASE

Bit 15 downto 6 are not used.

Bit 5: The BST electronics sends the 'bst_turn_clock' via the TTC-A-channel 1.2 us before sending a new set of BST-data via the TTC-B-channel.

EN_TURN_CLK **=1 The 'bst_turn_clock' signal loads 52 BST bytes from input registers into intermediate registers to append it at begin of the next orbit to event records.**

EN_TURN_CLK **=0 The new bst data set is loaded into intermediate registers at a programmed bc number, defined by the 'BST_UPDATE_DELAY'**

Bit 4:

BST52 =1 … adds 52 BST bytes to the event record,

BST52 =0 … adds 30 BST bytes to the event record (= old format)

Bits 3 – 0: ***NOT USED with BST FIFO***

SELECT_PHASE

B"0000" selects PHASE 0 (default value of firmware)

B"0001" selects PHASE 1

B"0010" selects PHASE 2

B"0011" selects PHASE 3

B"0100" - B"1111" inhibits BST data

4.8.4 BST DELAYS

The BCRES signal from the TIM board is delayed by 'BST_UPDATE_DELAY' to receive the BST data correctly.

The same BCRES signal is delayed in parallel by 'BST_ APPLY _DELAY' to apply the new data at the end of the orbit. Both delay values have to be different by at least one count.

4.8.4.1 BST_UPDATE DELAY

BB20 0046 BST_UPDATE_DELAY w/r // bits 15_0

Maximum value =3562 = 0DEA; Min.value= 0; UNIT= 1 BC

BST_UPDATE_DELAY = delay (unit = 1BC) since arrival time of bcres at EVM chip.

The LHC Beam Synchronous Timing Master (=BST) sends BST messages every orbit.

The GPS time(UTC format = us since 1.1.1970 as 64 bit nr) and 'turn count number' and the 'BST master status' are updated every orbit. The other parameters are updated at a maximum rate of 10 Hz and can be as old as 100ms.

The BST_UPDATE_DELAY should be set so that the delayed BCRES signal updates the BST registers before or after receiving new BST-data.

The test signals 'first_bst' and 'last_bst' or 'dout_strobe'show when new BST bytes arrive. The' update_bst' signal should never appear during that time.

4.8.4.2 BST_APPLY_DELAY

BB20 0048 BST_APPLY_DELAY w/r // bits 15_0

Maximum value =3562 = 0DEA; Min.value= 0; UNIT= 1 BC

BST_APPLY_DELAY = delay (unit = 1BC) since arrival time of bcres at EVM chip.

The BST_ APPLY _DELAY is used to apply the updated BST data at the end of an LHC orbit so that events with same orbit number have got also equal 'turn count number'.

The signal 'apply_bst' should appear shortly before 'bcres_delayed' which starts the new orbit. See register 'EVM_BCRES_DELAY'.

4.8.5 EVM_BCRES_DELAY

BB20 0000 EVM_BCRES_DELAY w/r

The BCRES signal from the TIM board is first delayed by 'EVM_BCRES_DELAY' to get correct bunch crossing numbers for the event records.

--UNIT= ½ BC (12.5 ns)

15 - 12	11 - 8	7 - 4	3 - 0
Delay C	Delay B	Delay A	Delay= 0…3

Total Delay = Delay C + Delay B + Delay A + (0…3)

-- DELAY =0 ➔ 0000 0000 0000 0000
-- DELAY =1 ➔ 0000 0000 0000 0001
-- DELAY =2 ➔ 0000 0000 0000 0010
-- DELAY =3 ➔ 0000 0000 0000 0011
-- DELAY =C+B+A+3 ➔ CCCC BBBB AAAA 0011

-- **For DELAY <4 the bits 15-4 have to be =0 !!**
-- Bits 3,2 are always =0; are not decoded

4.8.6 EVM_SHOW_SLINK_BITS

BB20 0002 EVM_SHOW_SLINK_BITS w/r
Mask register to show S-Link64 bits.
If a mask bit =1 then 15 signals are switched to 16 EVM_TEST points.
If all mask bits = 0 then all EVM_TEST points are switched off (0V).
Bit 0 : =1 Show S64_D(15:0) on testpoints EVM_TEST(15:0)
Bit 1 : =1 Show S64_D(31:16) on testpoints EVM_TEST(15:0)
Bit 2 : =1 Show S64_D(47:32) on testpoints EVM_TEST(15:0)
Bit 3 : =1 Show S64_D(63:48) on testpoints EVM_TEST(15:0)

Bit 4 : =1 Show 'gnd' on EVM_TEST(0) // to S-Link64
Show S64_NCTRL on EVM_TEST(1) // to S-Link64
Show S64_NRESET on EVM_TEST(2) // to S-Link64
Show S64_NTEST on EVM_TEST(3) // to S-Link64
Show S64_DW(0) on EVM_TEST(4) // to S-Link64
Show S64_DW(1) on EVM_TEST(5) // to S-Link64
Show S64_DOWN on EVM_TEST(6) // from S-Link64
Show S64_FULL on EVM_TEST(7) // from S-Link64
Show S64_LRL(3:0) on EVM_TEST(11:8) // from S-Link64
Show S64_RESV(2:0) on EVM_TEST(14:12) // from S-Link64
Show SEL_HEADER on EVM_TEST(15) // begin of record
Bit 5 : =1 Show REC_CLK(1:0) on testpoints EVM_TEST(1:0),
EVM_TEST(15:2) = all '0'
Bit 6: =1 Show rd_fifo(0) on EVM_TEST(0) // from ROC READ
Show rd_fifo(1) on EVM_TEST(1) // from ROC
Show 'gnd' on EVM_TEST(14:2) //
Show rd_bc_fifo on EVM_TEST(15) // from ROC

Bit 7: =1 Show wr_fifo_tp(0) on EVM_TEST(0) // chan_recvr
Show wr_fifo_tp(1) on EVM_TEST(1) // chan_recvr
Show 'gnd' on EVM_TEST(7:2) //
Show 'fdl_head27_24' on EVM_TEST(11:8) // Channel Link
Show 'tcs_head27_24' on EVM_TEST(15:12) // Channel Link

4.8.7 EVM_CMD_REG

BB20 0004 EVM_CMD_REG w/r // sim_mode, enable bits

Difference to DAQ chip: The Event_TYPE will be extracted from the TCS data record.

The Channel Link data from the FDL and TCS boards are sampled 2 times to select and transfer the better sample to the Channel FIFOs.

Bit15: SELECT_FDL_PHASE1

=1 selects phase1 and 0= selects phase0 of oversampled FDL data

Bit14: SELECT_TCS_PHASE1

=1 selects phase1 and 0= selects phase0 of oversampled TCS data

Bit13 := 1 RESET_SLINK, =0 release RESET_SLINK signal

This option can be used if the Slink requires a reset signal for more than 800 ns.

Bit 12: RESET_TTCRX …removed in V101B

// Sets and resets RESET_B signal to the TTCrx chip, this must be done by software in a sequence, because the "pulse" has to be < 50ms for proper operation.

=1 → RESET_B signal is active, resetting the TTCrx chip

=0 → RESET_B signal is inactive

Bit 11: not used since V0004

Bit 10: **LENGTH_5BX // Record Length of 5bx per Event from FDL expected**

The bit is copied to "**3/5BX_DEFINITION FOR FDL" word of event record.**

Bit 9: IGNORE_SLINK_DOWN // 1= ignore that the S-Link is down to send data
// whether the SLINK is connected or not

Bit 8: IGNORE _SLINK_FULL // ignore that the S-Link is full to send data
// whether the SLINK is full or not

Bit 7 : RES_SLINK_WITHOUT_RESYNC (V000E)

// 1= reset by VME only; 0= Resync and VME will reset the S-LINK chip

Bit 6 : SPY_EVERY_TICK (also when not sending data)

Bit 5: SIM_MODE ….simulation mode

=1: Data are transmitted from the SIM/SPY memory to the S-Link64 and data from Channel Links are ignored.

Bit 4: SLINK_TEST_MODE default: =0

Bit 3 : DEBUG_CHANLINKS … is valid for all channels

=0: standard data taking mode

=1: All Channel Link data are recorded, except when

header = "5"➔ IDLE code or

header = "F"➔ End of record

Bit 2 : **EN_ROBUS =1 enable ROBUS signals from TIM board**

'Start run' and 'stop run' are taken from ROBUS. The other signals (bcres, res_evnr, l1res…) are always taken from the encoded backplane signals.

Bit 1 & 0:

00 ➔ EVM_ROP DISCONNECTED for TCS

01 ➔ EVM_ROP BUSY_FOR_TCS

10 ➔ EVM_ROP READY_FOR_TCS

11 ➔ EVM_ROP DISCONNECTED for TCS

4.8.8 EVM_MAX_BC_NUMBER

BB20 0006 EVM_MAX_BC_NUMBER w/r //=orbit length -1

Default value := 0deb = 3564-1

4.8.9 INITIAL_CRC

BB20 0008 INITIAL_CRC w/r // bits 15_0 // initial CRC value
Default: "FFFF"

4.8.10 EVM_CMS_HEADER19_4 - CMS HeaderWord1

BB20 000A EVM_CMS_HEADER19_4 w/r //CMS HeaderWord1(19:4)
bits15:4:= GT-identifier in EVM software: 814 =32E(hex)
bits 3:0:= FORMAT_VERSION currently =0000
➔ **default value = X"32E0"**

4.8.11 EVM_EVENT_LENGTH

BB20 000C EVM_**EVENT_LENGTH_15_0 w/r // bits 15_0 : 24 bit length**
BB20 000E EVM_**EVENT_LENGTH_31_16 w/r // bits 31_16 : 24 bit length**
Calculate Event Length according to table about active boards:
3 Bunch-crossings per event: **32 W64** =
1 HEADER + 6 GTFE + 3 TCS + 21 FDL + 1 TRAILER
5 Bunch-crossings per event: **46 W64** =
1 HEADER + 6 GTFE + 3 TCS + 35 FDL + 1 TRAILER

3 bx / 80MHz : S-LINK64 transfer time per event: 32 x 12.5 ns ~ 400 ns = 0.4 us
5 bx / 80MHz: S-LINK64 transfer time per event: 46 x 12.5 ns ~ 575 ns = 0.58 us

3 bx / 40 MHz Channel Link transfer times: 16 data bits/transfer
FDL: 21 W64 x 4 = 84 ➔ *25ns ➔ 2.1 us
TCS: 3 W64 x 4 = 12 ➔ *25ns ➔ 0.3 us
5 bx / 40 MHz Channel Link transfer times:
FDL: 35 W64 x 4 = 140 ➔ *25ns ➔ 3.5 us
TCS: 3 W64 x 4 = 12 ➔ *25ns ➔ 0.3 us

4.8.12 EVM_GFTE HEADER WORD_1

4.8.12.1 EVM_BOARD_ID

BB20 0010 BOARD_ID w/r
bits 15:8: Board_ID of GTFE = VME slot number= 17dec=11hex
bits 7:0: length in bx =3 or 5 ➔ **default value = 1103**

4.8.12.2 EVM_SETUP_VERSION 31-0

BB20 0012 SETUP_VERSION_15_0 w/r // bits 15_0 //GFTE HeaderWord1
BB20 0014 SETUP_VERSION_31_16 w/r // bits 31_16 //GFTE HeaderWord1
➔ **default values = 0000 0000 …to be defined by EVM group**

4.8.13 EVM_ACTIVE_BOARDS

If bit xx =1 then the board is included in the event record.
GTFE board is always included. GTFE record length in EVM chip = 2 W64 (3 in EVM chip)

BIT#	BOARD	SLOT# in crate	Remark	Record_length (W64)
0	TCS	7	TriggerControl Board	2 (Version V0007) will be extended later
1	FDL	19	Final Decision Board	3bx*7=21; 5bx*7=35
2-15		---		

➔ **default value = 0003 …all boards included for a standard event record**

EVM_ACTIVE_BOARDS = X'0000' ➔ EVM chip does not send any event.

4.8.14 EVM_SPY_FULL_LIMIT

BB20 0018 EVM_SPY_FULL_LIMIT w/r // bits 15_0 // Spy control word

!!! Is used by spying as well as by simulating mode.

Value for 1 event: 0000 001F = 32 W64 -1 = 1event – 1
Value for n events = n*32 - 1
Value for 100 events = 3200 -1
Maximum: 4096-1 = 128 full events

4.8.15 EVM_TEST_MASK1, 2,3,4

BB20 001A EVM_TEST_MASK1 w/r // bits 15_0 ➔ **default value =8000**
BB20 001C EVM_TEST_MASK2 w/r // bits 15_0 ➔ **default value =8000**
BB20 001E EVM_TEST_MASK3 w/r // bits 15_0 ➔ **default value =8000**
BB20 004A EVM_TEST_MASK4 w/r // bits 15_0 ➔ **default value =0000**

Switch internal signal(s) to LEMO TEST outputs on the front panel if the corresponding mask bit(s) has(have) been set to '1'.
If more mask bits are set, then the 'OR' of the corresponding signals is displayed.
Signals of **TESTMASK3 and TESTMASK4 are 'OR'ed and go to the front panel Lemo 'EVM3' !!!!!!!!**

V101B Jan 2010

	EVM_TESTMASK1	**EVM_TESTMASK2**		**EVM_TESTMASK3**	**EVM_TESTMASK4**
bit	➔ 'EVM1'	➔ 'EVM2'	bit	➔ 'EVM3'	➔ **'EVM3'**
15	**clk40**	**bcres_dlyed**	15	**L1A_int**	bst_turnclk ←TTCrq
14	**L1A_int**	or_too_many_L1A	14	sel_din	dout_strobe ←TTCrq
13	vme_en	sel_trailer	13	rd_fifo(1) ←roc	TTC_ready ←TTCrq
12	dtack	slink_wen➔ Slink64	12	rd_fifo(0) ←roc	-
11	warning_evm	end_of_event(1) FDL	11	sel_chan(1) FDL	-
10	or_warn50	end_of_event(0) TCS	10	sel_chan(0) TCS	-
9	res_orbitnr_int	dout_strobe← TTCrq	9	sl_nctrl ➔ Slink64	L1Res_int
8	bc_error	busy_ priority	8	update_bst ← (bcres)	start_run
7	bcres_int	or_warn75	7	BCRes_long	run_rop
6	res_evnr_fsm	busy_evm	6	rd_bc_fifo ←roc	en_crc
5	l1res_fsm	wr_fifo(1) ←chan_recv	5	out_of_sync_evm	reset_crc
4	warn_ priority	wr_fifo(0) ←chan_recv	4	or_full	sel_header
3	ready_priority	apply_bst ←(bcres)	3	error_evm	sel_gtfe(3)
2	en_crc_trailer	ttc_ready ← TTCrq	2	error_priority	sel_gtfe(2)
1	sl64_down	last_bst ← TTCrq	1	out_of_sync_priority	sel_gtfe(1)
0	sl64_full	first_bst ← TTCrq	0	sel_gtfe(0) ← gtfe word	sel_err

V000D

	EVM_TESTMASK1		**EVM_TESTMASK2**		**EVM_TESTMASK3**
bit	SIGNALS	bit	SIGNALS	bit	SIGNALS
15	**clk40**	15	**bcres_dlyed**	15	**L1A_int**
14	sel_header	14	sel_err	14	sel_din
13	vme_en	13	sel_trailer	13	rd_fifo(1) ←roc

12	dtack	12	slink_wen➔ Slink64	12	rd_fifo(0) ←roc
11	start_run	11	end_of_event(1) FDL	11	sel_chan(1) FDL
10	run_rop	10	end_of_event(0) TCS	10	sel_chan(0) TCS
9	res_orbit_int	9	dout_strobe ← TTCrq	9	sl_nctrl ➔ Slink64
8	Bc_error	8	0	8	update_bst ← (bcres)
7	bcres_int	7	0	7	BCRes_long
6	res_evnr_fsm	6	0	6	rd_bc_fifo ←roc
5	l1res_fsm	5	wr_fifo(1) ←chan_recv	5	sel_gtfe(5) ← gtfe word
4	en_crc	4	wr_fifo(0) ←chan_recv	4	sel_gtfe(4)
3	reset_crc	3	apply_bst ←(bcres)	3	sel_gtfe(3)
2	en_crc_trailer	2	ttc_ready ← TTCrq	2	sel_gtfe(2)
1	sl64_down	1	last_bst ← TTCrq	1	sel_gtfe(1)
0	sl64_full	0	first_bst ← TTCrq	0	sel_gtfe(0)

4.9 EVM_STATUS REGISTERS

4.9.1 EVM_CHIP_ID

BB20 0020 EVM_CHIP_IDL -/r // bits 15_0 ➔ **default value = A031**
BB20 0022 EVM_CHIP_IDH -/r // bits 31_16 ➔ **default value = 0001**

4.9.2 EVM_VERSION_NR

BB20 0024 EVM_VERSION_NR -/r // bits 15_0 Firmware Version nr.

4.9.3 EVM_STATUS

BB20 0026 EVM_STATUS -/r // General status with SLINK
Bit 15 -12: EVM_status2tcs(3:0) // encoded status bits for TCS Trigger Control
0000 or 1111 = disconnected
0001 = warning
0010 = out of sync
0100 = busy
1000 = ready
1100 = error
1110 = bad code
Bit11: RUN_FF // = set by the START_RUN command from the TIM board
// = cleared by the STOP_RUN command from the TIM board
// = cleared by the CMD_PULSE(15) : = res_runff_vme …for tests
Bit 10: BC_ERROR // The flag is set whenever the external and internal BCReset disagree:
// Reasons: 1) wrong MAX_BC_NUMBER, 2.) bad BCRES signal
// The bc_error is cleared when reading the BC_ERROR counter.
Bit 9: SPY_MEM_FULL // Spy memory is full
Bit 8: SLINK_FULL // SLINK cannot receive data anymore, ROP waits to continue
// Original SLINK signal before gated by optional IGNORE bit
Bit 7: SLINK_DOWN // SLINK does not run
// Original SLINK signal before gated by optional IGNORE bit
Bit 6: SL_RESV2 // explanation to be added
Bit 5: SL_RESV1
Bit 4: SL_RESV0
Bit 3: SL_LRL3 // explanation to be added
Bit 2: SL_LRL2
Bit 1: SL_LRL1

Bit 0: SL_LRL0

4.9.4 EVM_CHAN_FIFOS_FULL

BB20 0028 EVM_CHAN_FIFOS_FULL -/r // FULL flags of all channels
Bit = 1 ➔ Channel Link FIFO is full.

bit	FULL FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
2	Event_typeFIFO		Data from TCS	(7)
3 -14			*Not defined*	
15	BCNR_FIFO		GTFE	

4.9.5 EVM_CHAN_FIFOS_WARN50

BB20 002A EVM_CHAN_FIFOS_WARN50 -/r // WARN50 flags of all channels
Bit =1 ➔ The FIFOs are filled up to the WARNING LEVEL (50%)

bit	WARN50 FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
2	Event_typeFIFO		Data from TCS	(7)
3 -14			*Not defined*	
15	BCNR_FIFO		GTFE	

4.9.6 EVM_CHAN_FIFOS_EMPTY

BB20 002C EVM_CHAN_FIFOS_EMPTY -/r // EMPTY flags of all channels
Bit = 1 ➔ Channel Link FIFO is empty
EMPTY means = no complete events

bit	EMPTY FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
2	Event_typeFIFO		Data from TCS	(7)
3 -14			*Not defined*	
15	BCNR_FIFO		GTFE	

4.9.7 EVM_CHAN_FIFOS_WARN75

BB20 002E EVM_CHAN_FIFOS_WARN75 -/r // WARN75 flags of all channels
Bit =1 ➔ The FIFOs are filled up to the WARNING LEVEL (75%)
(Previous CHAN_IS_TOO_LATE flags are not generated anymore.)

bit	WARN75 FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
2	Event_typeFIFO		Data from TCS	(7)
3 -14			*Not defined*	
15	BCNR_FIFO		GTFE	

4.9.8 EVM_CHAN_LINK_BAD_CODE

BB20 0030 EVM_CHAN_LINK_BAD_CODE -/r // Bad Header Code flags
Bit = 1 ➔ Channel Link sends undefined HEADER code.
This is a spurious signal that will be seen only when the error becomes permanent.

bit	BAD CODE FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
3 -15			*Not defined*	

4.9.9 EVM_CHAN_LINK_DATA_LOST

BB20 0032 EVM_CHAN_LINK_DATA_LOST -/r // ..of all channels
Bit = 1 ➔ Channel Link wants to write into a full FIFO.
This is a spurious signal that will be seen only when the TCS is sending L1A trigger signals even when the Channel FIFOs are full. In that case the GTFE-to-TCS connection does not work correctly or an IGNORE ERROR bit has been set incorrectly.

bit	DATA_LOST FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
2	Event_typeFIFO		Data from TCS	(7)
3 -15			*Not defined*	

4.9.10 EVM_CHAN_LINK_OK

BB20 0034 EVM_CHAN_LINK_OK -/r // ..of all channels
Bit = 1 ➔ Channel Link header bits = A,D,C,D,E, F or 5(=idle)
This was the previous CHAN_LINK_DOWN register.

bit	OK_ FLAG of	Channel#	Remarks	SLOTS
0	TCS	0		7
1	FDL	1		10
3 -15			*Not defined*	

4.9.11 EVM_STATE_MACHINE_STATUS

BB20 0036 EVM_STATE_MACHINE_STATUS -/r
EVM_STATE_MACHINE_STATUS word shows the present status of the **R**ead-**O**ut **C**ontroller (ROC) state machine that makes the event record.

- The ROC skips non active boards. In that case the EVENT_LENGTH register has to be programmed accordingly.
- If Channel Link data do not arrive within 511*12.5 ns = 6.387,5 ns then the ROC sends ‘FFFF’ instead and proceeds to the next channel.
- If the SLINK64 is not ready the ROC waits for an infinite time. Only a VME ‘res_roc’ command can reset the ROC to the IDLE state.
- According to the ‘length_5bx’ command register bit the ROC expects data form 3 or 5 bunchcrossings. If the record lengths of boards and the EVM chip disagree bad data will be sent.

Before and during an event:
8000 = IDLE status because run_rop =0…
(rop = 1 permanently in actual firmware versions V0022,23)
0000 = ROC is transmitting an event
Begin of an event:
10FF = waiting for SLINK64 before waiting for a new event //could be permanent
// res_roc_vme or L1RESET resets ROC to IDLE status
// The SLINK is either down or is full and has to be ready before we are waiting for a new L1A (event)
20FF = waiting for a new event= waiting for a L1A signal
4000 = FIFO FULL error, // permanent
// L1Reset(=Resync) or res_roc_vme reset ROC to IDLE status
8001 = L1reset/Resync procedure //transient
EE0F = end of event //transient
EEEF = end of run //time_out
TCS =channel 0 //TCS board …to get EVENT TYPE
0005 = get event type begin //transient
10EE = waiting for TCS data to get Event_Type //transient + timeout
0006 = eventype_state1 //transient
0007 = eventype_state2 //transient
0008 = eventype_state3 //transient
GTFE-Header data …transient states(12.5 ns)
0009 = sending header word //transient
000A = sending gtfe_word0 //transient
000B = sending gtfe_word1 //transient
000C = sending gtfe_word2 //transient
000D = sending gtfe_word3 //transient
000E = sending gtfe_word4 //transient
000F = sending gtfe_word5 //transient
TCS =channel 0 //TCS board …data
3000 = starting transfer of TCS data //transient
1000 = waiting for SLINK //could be permanent
// res_roc_vme resets ROC to IDLE status
2000 = waiting for TCS data //time_out
// fifo_empty(0) = '1
0011 = sending TCS data //transient
0012 = sending “FFFF”..error data instead of TCS words //transient
0013 = end of tcs data //transient
FDL =channel 1
3001 = starting transfer of FDL data //transient
1001 = waiting for SLINK //could be permanent
2001 = waiting for FDL data //time_out
0015 = sending FDL words of 3 bx //transient
0016 = sending FDL words of 5 bx //transient
0017 = sending error data (“FFFF..”) instead of 3bx-FDL words //transient
0018 = sending error data (“FFFF..”) instead of 5bx-FDL words //transient
TRAILER
001A = sending trailer word //transient

4.9.12 PHASE COUNTERS

BB20 0038 PHASE_CNTR_P3_0 -/r //

BB20 003A PHASE_CNTR_01 -/r //
BB20 003C PHASE_CNTR_12 -/r //
BB20 003E PHASE_CNTR_23 -/r //

The EVM chip samples the BST data from the TTCrq mezzanine board with a 160 MHz to find the switching time. Differences between consecutive sample increment the phase counters. PHASE_CNTR_P3_0 sees differences between the preceding sample3 and the sample0. A VME read access also clears the phase counters.

4.10 BC_NUMBERS of BST timing

The status registers show the BC-number at the time when the first and last BST-bytes arrive, when the new values are updated and when the new values are applied to be included into the event records. The registers are used when adjusting the BCRES and BST delay registers to integrate the BST time correctly into the event records.

BCNR at different times

BB20 0050 **BCNR_BST1** -/r // bits 15_0
-- shows the BC number when the first BST byte arrives via the TTCrq mezzanine board
BB20 0052 **BCNR_BST_LAST** -/r // bits 15_0
-- shows the BC number when the last BST byte arrives via the TTCrq mezzanine board
BB20 0054 **BCNR_UPDATE** -/r // bits 15_0
-- shows the BC number when the new BST values are loaded into intermediate registers
BB20 0056 **BCNR_APPLY** -/r // bits 15_0
-- shows the BC number when the new updated BST values are ready to be included into event records.

4.11 SIM_SPY_ADDRESS

BB20 0058 **EVM_SIM_SPY_ADDRESS**

The evm_sim_spy_address shows the actual value of the spy address. Can be used to check how many events have been spied since start_simspy command pulse. This value might be useful to see if a L1A has arrived.

4.12 TTC_OFF_ERRORS V0018

BB20 005A TTC_OFF_ERRORS shows how often the TTCrx became not ready since the last reset command pulse EVM_CMD_PULSE(12).
In case of overflow the counter is locked to X“FFFF“.

4.13 TTC_DOUBLE_ERRORS V0018

BB20 005C TTC_DOUBLE_ERRORS shows the number of double errors of the TTCrx chip since the last reset command pulse EVM_CMD_PULSE(12).
This counter should always be zero. In case of overflow the counter is locked to X“FFFF“.

4.14 BST_STROBE_CNTRS

BB20 005E BST_STROBE_CNTRS -/r
--bit15-8: number of strobe signals after input FIFO
--bit 7-0: number of strobe signals before FIFO at the chip input
The difference between both counters should should not exceed 1.

4.15 BST registers

The GTFE circuit assumes that BST byte 1 is sent first.
A new version with different structure:

The bytes are assembled as described in Wiki page: BST-config.html
LHC-BOB-ES-0001-10-00.pdf from 2006-02-08 ... is obsolete

4.15.1 GPS TIME or UTC TIME

BB20 0060	GPS_TIME0	-/r	// GPS 15 - 0
BB20 0062	GPS_TIME1	-/r	// GPS 31 - 16
BB20 0064	GPS_TIME2	-/r	// GPS 47 - 32
BB20 0066	GPS_TIME3	-/r	// GPS 63 – 48

Updated every LHC orbit.
The UTC time is sent as 64 bit number since 1.1.1970.
Bits 63 – 32: (seconds) + Bits 31 – 00: (micro seconds)

4.15.2 ACQUISITION_TRIGGERS

BB20 0068	ACQU_TRIGGERS9_8	-/r	//	// bst-byte 9,8
BB20 006A	ACQU_TRIGGERS11_10	-/r	//	// bst-byte 11,10
BB20 006C	ACQU_TRIGGERS13_12	-/r	//	// bst-byte 13, 12
BB20 006E	ACQU_TRIGGERS15_14	-/r	//	// bst-byte 15, 14

Table: See WEB page of BST … *only few bits are used*

4.15.3 BST_STATUS_ABBI_DIAGN

BB20 0070 BST_STATUS_ABBI_DIAGN -/r // bst-byte 17, 16

Bit 15-8: BST_MASTER_STATUS
updated every LHC orbit.
Bit 9=1: BST Master Beam #2 sent this message
Bit 8=1: BST Master Beam #1 sent this message

Bit 7 – 0: AB-BI DIAGNOSTICS
Bit 7: reset the interrupt history in evers BOBR
Bit 0 trigger an interrupt

4.15.4 TURN NUMBER

BB20 0072	TURN_NRL	-/r	// bits 15_0	// bst-byte 19,18
BB20 0074	TURN_NRH	-/r	// bits 31_16	// bst-byte 21, 20

Updated every LHC orbit. Reset on first injection

4.15.5 LHC FILL UMBER

BB20 0076	LHC_FILL_NRL	-/r	// bits 15_0	// bst-byte 23, 22
BB20 0078	LHC_FILL_NRH	-/r	// bits 31_16	// bst-byte 25, 24

Updated on change. Alternatively could show the UTC time in seconds.

4.15.6 MACHINE_MODE

BB20 007A MACHINE_MODE -/r // // bst-byte 27, 26

Machine mode is updated when changed:
no_beam, filling, ramping, adjusting, physics

4.15.7 PARTICLE_TYPE

BB20 007C PARTICLE_TYPE -/r // // bst-byte 29, 28

15_8: of RING 2: proton, Pb,..
7_0: of RING 1: proton, Pb,..

4.15.8 BEAM_MOMENTUM

BB20 007E BEAM_MOMENTUM -/r // // bst-byte 31, 30

// in GeV/c;

Updated with 10 Hz.

4.15.9 BEAM INTENSITIES

BB20 0080	INTENSITY_RING1_L	-/r	//	// bst-byte 33, 32
BB20 0082	INTENSITY_RING1_H	-/r	//	// bst-byte 35, 34
BB20 0084	INTENSITY_RING2_L	-/r	//	// bst-byte 37, 36
BB20 0086	INTENSITY_RING2_H	-/r	//	// bst-byte 39, 38

Updated with 10 Hz. Integer x 10e8 charges

4.15.10 INTENSITY FROM BCTF

BB20 0088	INT_BCTF_RING1_L	-/r	//	// bst-byte 41, 40
BB20 008A	INT_BCTF_RING1_H	-/r	//	// bst-byte 43, 42
BB20 008C	INT_BCTF_RING2_L	-/r	//	// bst-byte 45, 44
BB20 008E	INT_BCTF_RING2_H	-/r	//	// bst-byte 47, 46

4.15.11 BUNCH COUNT FROM BCTF

BB20 0090	BC_BCTF_RING1	-/r	//	// bst-byte 49, 48
BB20 0092	BC_BCTF_RING2	-/r	//	// bst-byte 51, 50

4.15.12CODE_FOR_BST_SOURCE

BB20 0094 CODE_FOR_BST_SOURCE -/r

= “0000” TTCrx is DOWN, link to BST is missing

= “BEA0” TTCrx is working, data from BST LHC Beam Monitoring

4.15.13BST SIMULATION REGISTERS

The firmware applies Simulation data to the Event record automatically when link to BST is missing, that means that TTCrx is down (TTC_READY=0).

BB20 00A0	SIM_GPS_TIME0	w/r	// GPS 15 – 0	// bst-byte 1,0
BB20 00A2	SIM_GPS_TIME1	w/r	// GPS 31 – 16	// bst-byte 3,2
BB20 00A4	SIM_GPS_TIME2	w/r	// GPS 47 - 32	// bst-byte 5,4
BB20 00A6	SIM_GPS_TIME3	w/r	// GPS 63 – 48	// bst-byte 7,6
BB20 00A8	SIM_ACQU_TRIGGERS9_8	w/r	//	// bst-byte 9,8
BB20 00AA	SIM_ACQU_TRIGGERS11_10	w/r	//	// bst-byte 11,10
BB20 00AC	SIM_ACQU_TRIGGERS13_12	w/r	//	// bst-byte 13, 12
BB20 00AE	SIM_ACQU_TRIGGERS15_14	w/r	//	// bst-byte 15, 14
BB20 00B0	SIM_BST_STATUS_ABBI_DIAGN		w/r	// bst-byte 17, 16
BB20 00B2	SIM_TURN_NRL	w/r	// bits 15_0	// bst-byte 19,18
BB20 00B4	SIM_TURN_NRH	w/r	// bits 31_16	// bst-byte 21, 20
BB20 00B6	SIM_LHC_FILL_NRL	w/r	// bits 15_0	// bst-byte 23, 22
BB20 00B8	SIM_LHC_FILL_NRH	w/r	// bits 31_16	// bst-byte 25, 24
BB20 00BA	SIM_MACHINE_MODE	w/r	//	// bst-byte 27, 26
BB20 00BC	SIM_PARTICLE_TYPE	w/r	//	// bst-byte 29, 28
BB20 00BE	SIM_BEAM_MOMENTUM	w/r	//	// bst-byte 31, 30
BB20 00C0	SIM_INTENSITY_RING1_L	w/r	//	// bst-byte 33, 32
BB20 00C2	SIM_INTENSITY_RING1_H	w/r	//	// bst-byte 35, 34
BB20 00C4	SIM_INTENSITY_RING2_L	w/r	//	// bst-byte 37, 36
BB20 00C6	SIM_INTENSITY_RING2_H	w/r	//	// bst-byte 39, 38
BB20 00C8	SIM_INT_BCTF_RING1_L	w/r	//	// bst-byte 41, 40
BB20 00CA	SIM_INT_BCTF_RING1_H	w/r	//	// bst-byte 43, 42
BB20 00CC	SIM_INT_BCTF_RING2_L	w/r	//	// bst-byte 45, 44
BB20 00CE	SIM_INT_BCTF_RING2_H	w/r	//	// bst-byte 47, 46

BB20 00D0	SIM_BC_BCTF_RING1	w/r	//	// bst-byte 49, 48
BB20 00D2	SIM_BC_BCTF_RING2	w/r	//	// bst-byte 51, 50

BB20 00D4	SIM_CODE_FOR_BST_SOURCE	w/r	// =”DDDD” for simulated data

4.16 MONITORING COUNTERS

Version V0013: 13. Aug 08

The registers contain the contents of the ‘Monitoring Counters’. A common REFRESH_MON_COUNTERS signal moves the actual counter contents into registers and clears all counters concurrently at the next BCRES signal.

Without a command pulse every 100 s the registers are updated and the counters are cleared automatically.

Counting Unit: BC ~25 ns; f= 40,0786 MHz

BB20 0100	MONITORING_TIME_L	-/r	unit: BC
BB20 0102	MONITORING_TIME_H	-/r	unit: BC
BB20 0104	CNTR_SLINK_DOWN_L	-/r	unit: BC
BB20 0106	CNTR_SLINK_DOWN_H	-/r	unit: BC
BB20 0108	CNTR_SLINK_FULL_L	-/r	
BB20 010A	CNTR_SLINK_FULL_H	-/r	
BB20 010C	CNTR_WAITING_FOR_SLINK_L	-/r	
BB20 010E	CNTR_WAITING_FOR_SLINK_H	-/r	
BB20 0110	CNTR_WAITING_FOR_FDL_L	-/r	
BB20 0112	CNTR_WAITING_FOR_FDL_H	-/r	
BB20 0114	CNTR_FIFO_FULL_FDL_L		-/r
BB20 0116	CNTR_FIFO_FULL_FDL_H		-/r
BB20 0118	CNTR_FIFO_WARN75_FDL_L	-/r	
BB20 011A	CNTR_FIFO_WARN75_FDL_H	-/r	
BB20 011C	CNTR_FIFO_WARN50_FDL_L	-/r	
BB20 011E	CNTR_FIFO_WARN50_FDL_H	-/r	
BB20 0120	CNTR_TOO_MANY_L1A_FDL_L	-/r	
BB20 0122	CNTR_TOO_MANY_L1A_FDL_H	-/r	
BB20 0124	CNTR_WAITING_FOR_TCS_L	-/r	
BB20 0126	CNTR_WAITING_FOR_TCS_H	-/r	
BB20 0128	CNTR_FIFO_FULL_ TCS_L	-/r	
BB20 012A	CNTR_FIFO_FULL_ TCS_H	-/r	
BB20 012C	CNTR_FIFO_WARN75_ TCS_L	-/r	
BB20 012E	CNTR_FIFO_WARN75_ TCS_H	-/r	
BB20 0130	CNTR_FIFO_WARN50_ TCS_L	-/r	
BB20 0132	CNTR_FIFO_WARN50_ TCS_H	-/r	
BB20 0134	CNTR_TOO_MANY_L1A_ TCS_L		-/r
BB20 0136	CNTR_TOO_MANY_L1A_ TCS_H		-/r

BB20 0138	CNTR_EVENTS_TO_SLINK_L	-/r	
BB20 013A	CNTR_EVENTS_TO_SLINK_H	-/r	
BB20 013C	CNTR_FFF_RECORDS _L	-/r	
BB20 013E	CNTR_FFF_RECORDS _H	-/r	

MONITORING_TIME counts the time in BC’s.

CNTR_ SLINK _DOWN is incremented while SLINK DOWN FLAG is active.
CNTR_ SLINK _FULL is incremented while SLINK FULL FLAG is active.
CNTR_WAITING_FOR _SLINK is incremented while the Readout Controller State Machine (ROC) is in the status 'waiting _for_SLINK'.
CNTR_EVENTS_TO_SLINK counts the number of events sent to the SLINK64. The 'select_trailer' increments the counter.
CNTR_FFF_RECORDS counts the number of "fff...fff records in events sent to the SLINK64.

Counters for each board sending event records:

CNTR_WAITING_FOR_xxx is incremented while the Readout Controller State Machine (ROC) is in the status 'waiting_for_xxx' board.
CNTR_FIFO_FULL _xx is incremented while the FIFO becomes full.
CNTR_FIFO_WARN75_xx is incremented while the FIFO becomes 75%full.
CNTR_FIFO_WARN50_xx is incremented while the FIFO becomes 50%full.
CNTR_TOO_MANY_L1A_xx is incremented while too many events are in the data chain and the FIFO could become full. That means when the number of expected events plus the number of events in the FIFO exceeds the FIFO capacity. The number of expected events is the difference between the number of L1A signals received from the TIM board and the number of events received until now.

Firmware: FDL → Channel 1; TCS → Channel0

4.17 LED

Front panel LED shows errors:

-- CHANNEL_LINK_TOO_LATE FLAGS
-- BAD_HEADER FLAGS
-- CHANNEL_LINK_DOWN FLAGS
-- BC_ERROR
-- SLINK DOWN FLAG

5 TEST Programs

5.1 General remarks

- After Power-up clear BC-ERROR FLAG with command pulse 'res_bc_error'. Then also the front panel LED might become off, when no other error is active.

5.2 Send Simulation data to SLINK64

- Load data into SIMSPY memory:

Four 16-bit words are sent as one 64-bit word to the SLINK64.

VME_address + 0: SLINK word bits 15 - 0
VME _address + 2: SLINK word bits 31 - 16
VME _address + 4: SLINK word bits 47 - 32
VME _address + 6: SLINK word bits 63 – 48

- Set SPY_FULL_LIMIT = nnn….# of 64-bit words
- Set DAQ_CMD_REG = hex 0020 …sim_mode=1
- Send CMD_PULSE = hex 0080 → Bit 7: start_spying_ simulating

→ The logic sends now the content of the SIMSPY memory (0 up to the 'full_limit') to the SLINK64.

5.3 Data transfer test PSB ➔ GTFE

Hardware: TIM6UV2, PSB somwhere mounted, GTFE with DAQ chip

5.3.1 Prepare TIM6U_V2

- TIM6UV2 should broadcast 40 MHz clock and a BCRES continually.

5.3.2 Prepare PSB9U_(V2)

- PSB9U(_V2) : Load all 8 SIM memories with test data from file(s).
- PSB9U(_V2) : Send simulation data continually. (sim_mode, 'startsimspy'…)

5.3.3 Setup DAQ GTFE registers

- **Bold registers have to be set for the test.**
- Check the other registers if the default values are really set.

BB10 0042 **IGNORE_ERRORS** **X"00FF"** // ignore all errors
BB10 0000 BCRES_DELAY X"0000"
BB10 0002 SHOW_SLINK_BITS X"0010" to show S-Link64 CONTROL bits
BB10 0004 **DAQ_CMD_REG** **X"1002"**
// event_type=1, READY =2, bit5=0=spy_mode
BB10 0006 MAX_BC_NUMBER X" 0DEB" //=orbit length -1
BB10 0008 INITIAL_CRC X" FFFF" // initial CRC value
BB10 000A CMS_HEADER19_4 X" 32D0"
BB10 000C **EVENT_LENGTH_15_0** **X"0016"** // bits 15_0 : 24 bit length
// default: 00C7 =for all boards
BB10 000E EVENT_LENGTH_31_16 X"0000" // bits 31_16 : 24 bit length

Calculate Event Length in W64 words according to table about active boards:

3 Bunch-crossings per event: **199(=00C8) W64** =

1 HEADER + 2 GTFE + 21 FDL + 18 PSB_slot9 + 18 PSB_ slot13 + 18 PSB_ slot14 + 18 PSB_ slot15 + 48 GMT + 18 PSB_ slot 19 + 18 PSB_ slot 20 + 18 PSB_ slot 21 + 1 TRAILER

Example: 1 PSB sends data: 1 HEADER + 2 GTFE + 18 PSB + 1 TRAILER = 22dec = **X"0016"**…W64 (=64bit word)

BB10 0010 BOARD_ID X"1103" // ID=11, 3= 3bx event
BB10 0012 SETUP_VERSION_15_0 X"0000" //GFTE HeaderWord1
BB10 0014 SETUP_VERSION_31_16 X"0000" //GFTE HeaderWord1
BB10 0016 **ACTIVE_BOARDS** **X"0004"** //GFTE HeaderWord2
Example: PSB is mounted in slot 13

For other slots see table:

BIT#	BOARD	SLOT# in crate	Remark	Record_length (W64)
0	FDL	10	Final Decision board	3*7=21 (5*7=35)
1	PSB_0	9	Techn.Triggers for FDL	3*6=18 (5*6=30)
2	PSB_1	13	Calo data for GTL	3*6=18 (5*6=30)
3	PSB_2	14	Calo data for GTL	3*6=18 (5*6=30)
4	PSB_3	15	Calo data for GTL	3*6=18 (5*6=30)
5	PSB_4	19	M/Q bits for GMT	3*6=18 (5*6=30)
6	PSB_5	20	M/Q bits for GMT	3*6=18 (5*6=30)
7	PSB_6	21	M/Q bits for GMT	3*6=18 (5*6=30)
8	GMT	18	Global Muon Trigger	16*3=48 (5*16=80)

BB10 0018 **SPY_FULL_LIMIT X"00DC"**
Depends from the event size (W64) and the number of events we want to store in the SIM/SPY memory . Maximum is X"0FFF" = 4095 W64.
Example: 3bx-event with 1 PSB : 22 W64/event; 10 events ➔ 220 W64 ➔ X"00DC"

BB10 001A **TEST_MASK1 X"8000" ➔ clk40 ➔ front-panel LEMO 'D1'**
BB10 001C **TEST_MASK2 X"8000" ➔ bcres_dlyed➔ front-panel LEMO 'D2'**
BB10 001E **TEST_MASK3 X"8000" ➔ L1a_int ➔ front-panel LEMO 'D3'**
➔ Check the signals with an oscilloscope.

5.3.4 Check GTFE status

Read DAQ_STATUS expected:
If the SLINK is not mounted then we will see: "8c7f"

- 8 => ready,
- c = 1100 => 1=don'tcare, 1=bcerror at begin(read bc_error counter to clear flag), 0= spy_mem is not full, 0=slink is not full when not connected;
- 7 = 0111 => 0=slink is not down when not connected, 111=sl_resv..not connected
- f = 1111 => sl_lrl(3:0) when slink is not connected

5.3.5 Data transfer with L1A

- TIM6U:

start tim6UV2_gui: go to TestMode: select CLK source, and which clock goes to backplane

- PSB9U: 'startsimspy'…..all SIM memories should send data to the backplane
- GTFE: 'start_spying_ simulating' (CMD_PULSE: bit7=1 =>"0080")
- GTFE: check STATE_MACHINE_STATUS = X"**8000**" = IDLE
- TIM6U: 'start_run' GUI: single bgo-commands
  or use daq_cmd_pulse: 'start_rop' GTFE:CMD_PULSE(5) <= "0020"
  ==> STATE_MACHINE_STATUS = X"**20FF**" =waiting for an event
- Send L1A signals (one or more)
  TIM6U GUI: set BCTABLE: PER_L1A….assign an address
  L1A_PERIOD ….small/ big number
  ➔ check L1Afrequency with oscilloscope :GTFE D3 Lemo
  Remark: If we send on the gtfe-daq a 'l1a_vme' pulse we get an event of correct format but with 'FFF' data.
- Check if SIM/SPY is full: DAQ_STATUS(9) = SIMSPY_FULL
- Read DAQ: SIM/SPY into a file and compare data with PSB-sim memories
  Remark: Missing data are replaced by 'FFFF' in the event record going to the SLINK.
- *See also GTFE_readout_format.doc*

5.3.6 Loop and check data transfer automatically

- **Set up boards and registers in GTFE:**
  - Setup the boards TIM, PSB, FDL, GTFE as before and in
  - **GFTE** set DAQ_CMD_REG: bit 11: =1 ONE_EVENT_MODE to start writing of an event always at begin of the memory
- **Prepare for spying of next L1A**
  - Send in GTFE_DAQ: CMD_PULSE: bit 7: 'start_spying_ simulating'(0080) ➔ also resets SIM_SPY_ADDRESS
- **Send next L1A**
- **Automatic data checking:**
  - Read SIM_SPY_ADDRESS to see if data after an L1A have arrived.

=0 when no L1A has been sent, = event_size(W64)+1
    - Send GTFE_DAQ: CMD_PULSE: bit 8: 'stop_spying_ simulating' (0100)
    - Compare SIM/SPY memory/file with PSB memories/files
    - *(Write SIM_SPY memory '0000' to clear the old values.)…= not necessary because next L1A will contain different data.*
- Return to 'prepare for spying of next L1A'

Generate L1A periodically or by pushing the L1A button in TIM-GUI. The time interval between two L1A should be long enough to do the data comparison in software.

6 Transfer time in simulation

1 tick = 12.5 ns
Waiting time for data after a L1A:
The GTFE sends Header word, then waits for about 195 ticks ~2.4us until FDL data have arrived, that is the Channel Link transfer time.
Break of 1 tick: between 2 consecutive boards.
GMT: Break of 8 ticks after 8 words in 2nd bx-data.
Break after last data: 19 ticks, then the trailer word is sent.

FIFO Sizes:
-- FIFO 100%level: gmt=80, fdl=48.5, psb =56.7 tcs= 204
-- FIFO 75% level: gmt=60, fdl=37, psb= 43, tcs= 153
-- FIFO 50% level: gmt=40, fdl=24, psb= 28, tcs= 102

ChanLink transfer time: (FFFF word included)
-- FDL: 2425 ns <-- 2125ns(85 words) + 300 ns between
-- PSB: 2225 ns <-- 1975ns(79 words) + 250 ns between
-- GMT: 5375 ns <-- 5150ns(206 words)+ 225 ns between records
-- TCS: 575 ns <-- 525ns(21 words)+ 50 ns between records ... on EVM chip

Calculation: At maximum trigger rate
-- PSB50%warning appears after 19.7 events (50%level =28 events)
-- PSB75%warning appears after 30.3 events (75%level =43 events)
s = events to slink,
f = events sent to fifo,
w = 50%warning level (events)
t_xx = transfer time per event on channel link
The GMT transfer needs more time (**t_gmt)** than any other transfer. →
Therefore the maximum rate to S-Link ~ maximum rate into GMT FIFO →
While sending **s** events to S-Link with t_gmt a PSB-, FDL-, TCS FIFO receives **f** events:
→ **t_gmt * s = t_xxx * f** … same time
- When a PSB-, FDL-, TCS FIFO becomes 50% full a warning flag will be sent:
→ **w = f – s** **… events in fifo**
→ **s = w/((t_gmt/t_xxx) -1) ... events at maximum rate until warning flag**
Chan_link transfer times per event:
t_gmt = 5375 ns, t_psb = 2225 ns, t_fdl = 2425, t_tcs = 575 ns

-- PSB: 28/((5375/2225)- 1) = 19.7 events !!!!! ...seen in ModelSim
-- FDL: 24/((5375/2425)- 1) = 19.7 events !!!!! ...seen in ModelSim
-- TCS: 102/((5375/ 575)- 1) = 12.2 events !!!!!

-- TCS: 204/((5375/ 575)- 1) = 24.4 events when EVM FIFO has double size!!
---

7 SLINK64 modules

7.1 CMC boards

IEEE P1386 Standard Mechanics for a Common Mezzanine Card Family
The card size is 74mm x 149mm x 8.2mm (single width).