FDL
Description

A. Taurok

FPGA XC2V4000
Version 1017
24 June 2015

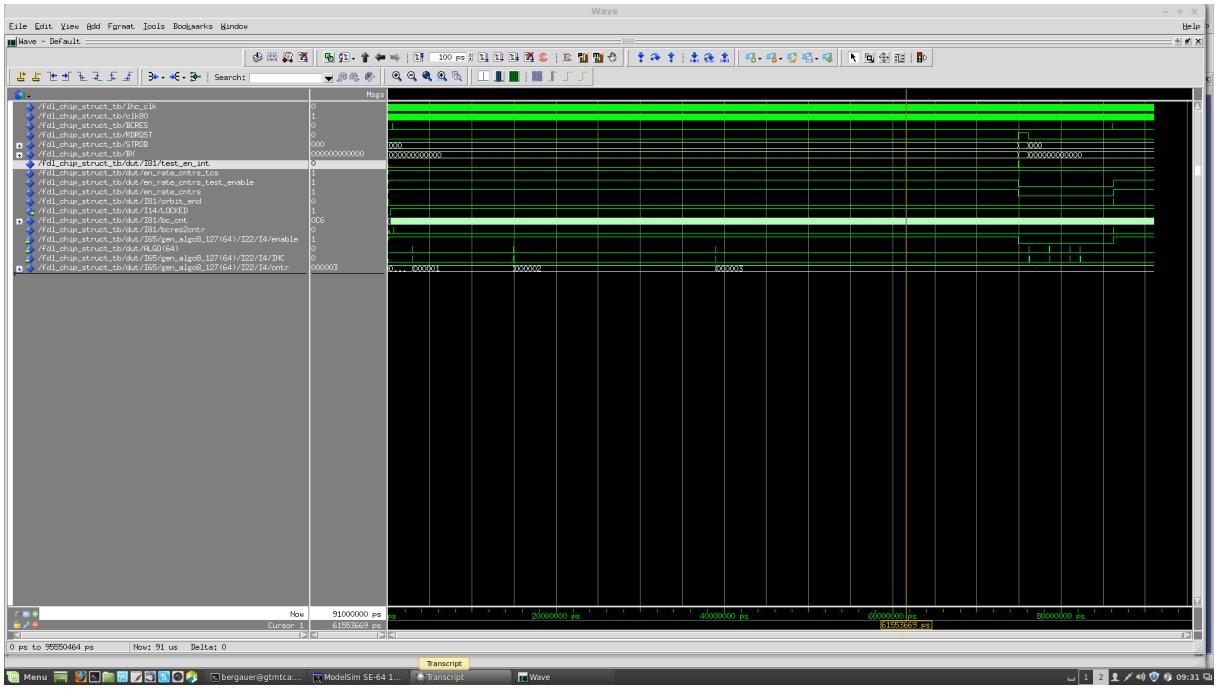
FDL Module

1 FDL chip

1.1 Update Notes:

1.1.1 V1017

-- HB 2015-06-24: changed logic in 'time_run_control.vhd' to **inhibit** rate-counters
in the period starting with BGo 'test enable' and ending with end of orbit.
This is needed, because TCS is not working anymore (now TCDS is used), so Calibration Trigger can cause data, which are generating algos with no relevance for physics, they have to be not counted.
See picture below for simulation of V1017 for rate-counters.



1.1.2 V1016 & V1015

-- Errors of V1015 (eventnr, bc nr) corrected
-- FPGAdv8.2 & ISE10.1 used; Firmware implemented on X-disk on Schreiner_pc
-- ROP: New 5bx-event logic:
A FIFO stores the ringbuffer readout address -2 of the L1A signal until the 'l1a_sm'-state machine loads it into a second address counter to extract data from the ring buffer.
-- en_rate_cntrs <-- FDLUTCS (3) ... error of V1014 corrected
-- vme_decoder: FF added on 'vme_rd_rate_reg', 'vme_rd_status_reg' signals
-- additional register for vme-data(‚vdin') to be loaded into registers.

1.1.3 V1014

- General_Register(14):=' tcs_stops_cntrs' ; ←**Software change required**

TCS sends signal to inhibit rate counters during calibration cycles.
Interface signals fdl<--> tcs signals with new definition.

- BCRES delays corrected to get coherent orbit+lumsegm numbers precisely at the end of an orbit.
- General_Register(11) := ‚en_freeze' ... is not used
  (Signal 'freeze_counters' has been deleted in firmware)

1.1.4 V1013

Error of V1012 corrected: Now the same latency_delay can be used for 3 and 5bx events.

1.1.5 V1012

- ROP/delay_fdl_rop:

1.) 2 FF were added after the bcres_delay to write trigger data concurrently into the Ring_buffer and the SPY memory.

2.) The delayed bcres signal is then applied to the latency dely units so that even for a latency = 0 the read_address counter is always reset after the write_address counter thus avoiding concurrent read and write access at the same memory location.

New Delay scheme:

```
-- BCRES→dly(11:8)→FD → FD → res_wr_addr
--                    |→dly(7:4)→dly(3:0)→ → FD → res_rd_addr(5bx event)
--                                        |→ res_rd_addr(3bx event)
```

1.1.6 V1011

- Ringbuffer & FIFO for Event-, Orbit- and Luminosity Segment nr added
- **'rate_register_v6' replaced by 'read_rates' so that FDL and TCS deliver counter contents from the same luminosity_segment.**
- prescale_version_nr and prescale factors are updated concurrently
- 'fdl_slice' simplified, same behavior as before
- No_Algo logic updated
- **Route Signal2Pan Register : Testpoint: res_orbnr added**
- VME logic: vme_enn with vme_en1,2,3
- Lumi Segment Length =2**20 ~93sec …0010_0000 hex default value

1.1.7 V1010

New address: PRESCALE_VERSION .. applied with new set of prescale factors
New format of FDL event record.

1.1.8 V100F

Luminosity Segment counter (lumi_cntr) is also reset when a new lum_segment_length has been loaded.

Software change required: Testpoint: BCRES_i added

1.1.9 V100E

Ringbuffer: permanent read instead of applying 'L1Apulse'

1.1.10 V100D

TTRIG input: additional pipeline register added to compensate inverted 40MHz clock in FDL chip.

1.1.11 V100C

New test-point signal: L1A ← RoutSignal2Pan Router
New Power-up values:

- Noalgo includes 4 no_algo_s signals and is pre-scaled by 400.000.
- all slices go to Finor0
- no prescaling for algo- and ttrig- signals

Out_of_sync & error logic modified.

1.1.12 V100B

2 4c68	lum_segm_period	32 bits

New update period selection.
‚Route_sig2pan' : 2 new testpoint signals!!

1.1.13 V100A Dec2007

Status register: new status bits, bit positions changed
Rate_H registers 0x31000: 4 are implemented
General Register: Bit 0,1,12,13 : sel_sim_... etc
Bit 28: disable status signals of slot 10 (FDL)
Common Command Pulses
Bit 3: load_noalgo_prescaler
Bit 4: start simspy_finor with next bc reset.
STATUS decoding corrected according to CMS Note 02-033

1.2 Hardware remarks

2 Panel_output pins: TTRIG_PAN….switch an ALGO bit to panel
8 bits = algo number
FINOR_PAN…switch a FINOR bit to panel.
3 bits = finor number
FINOR …**differential** output pins
A_BEAM… **differential** output pins

Signals from TCS:
FDLU_TCS(3) : Freeze signal sent by TCS in case of errors.
FDLU_TCS(2) : Refresh signal of TCS to reset rate counters, when selected in UPDATE PERIOD registers.
FDLTCS(3), FDLTCS(2): Signals from TCS are not used by FDL
Signals to TCS:
FDLU_TCS(1) = FDLU_TCS(0) =0 …FDL does not send signals
FDLTCS(1) = FDLTCS(0) =0 …FDL does not send signals
Signals from TIM:
TIMFDL(3:0) : = TIM status

1.3 Logic Description

The VHDL modules of the FDL_CHIP are described in the following chapters.

1.3.1 DCM & clear Clock logic

The DCM module (see unisim library) converts the 40 MHz CLK_TO_FDL signal to a set of internal 40 and 80 MHz clock signals. The „clk“ signal is **inverted** with referece to the input clock to reduce the latency for trigger data when transferring data from GTL to FDL and from FDL to TCS board.
The „clr“ is applied to all flip-flops in the chip to load correct power-up values.
Other IO signals: CLK_LOCKED_FDL, RESET_FDL_CHIP

1.3.2 VME interface logic

vme_en_sigs ... makes signals vme_en, vme_wr, vme_addr which are used by vme_decoder;
vme_dis_rd controls the direction of the bidirectional vme data flow.

vme_decoder_fdl : generates enable and read signals to load and to read registers of the chip and also the DTACK data acknowledge signal for the VME control chip.

3 modul types for VME Registers:

- vme_wr_reg ...is used to make 16 or 32 bit registers, which can be loaded and read back by software.
- vme_pulse_reg ...is used to generates pulses by software. When a bit is set =1 then a pulse will be sent during a write access to start or stop a activities in the chip
- vme_status_reg ...is used to make 16 or 32 bit status registers, where the bits are set by the internal fdl logic and read by software. Some are cleared after a read access. Examples: Version number, chip_id, ...

Instances of VME- registers can be found in several modules.

1.3.3 TIME_RUN_CONTROL

Input:

The signals L1A, BCRES, L1RESET from the TIM board are encoded to carry also other Bgo commands.

--**L1RESET**	0 0 1	
--**BCRES**	x 1 0	
--res_evnr_tim	0 1 1	...Reset Event number
--**L1A**	1 x 0	
--start_stop_tim	1 0 1	-- not used in FDL
--res_orbnr_tim	1 1 1	or TIM board is off -- not used in FDL

Remark: **BCRES** and **L1A** may arrive concurrently!!

The RO-bus signals (BX(11:0), STROB(2:0), RDRQST) carry also BGO commands. In the FDL chip only the hardware reset signal („hard_res“) is taken from the robus.

Function:

The control signals from the TIM board are merged with software generated commands when doing stand-alone tests.

BCRES can be delayed to adjust trigger data inside the FDL chip correctly to the LHC orbit.

A Bunch counter and comparator allow to run the chip also without or with a non periodic the external BCRES signal. But a error counter and erroflag show any discrepancies between the external and internally generated BCRES signal.

An ORBIT CNTR is incremented by every internal BCRES signal and cleared either by software or by a Bgo command, broadcast to all boards by the TCS chip via the TIM board.

Output:

1.3.4 EVM_STATUS_ENCODER

STATUS ENCODER creates a 4 bit code according to status of the rop_evm and rop_daq chip. The result called "EVM_STATUS" is sent via the backplane to the FDL board, where it is merged with the states of other boards of the GT-crate. From the FDL board the "GT_STATUS" is sent to the TCS board. The Partition Controller (PTC) on the TCS board accepts the GT_STATUS either to broadcast or to inhibit trigger signals (L1A).

Monitoring counter for each error and warning type.

Input signals:

- Several error and warning signals from bcnr_fifo and FIFOs in chan_recvr and evm_get_event_type, evm_evnr_check, evm_time_run_control and SLINK
- 16 cmd_reg bits to set the EVM chip to disconnected or busy or ready status by software and to ignore some error types.

- ‚active_chan' register to include or exclude FDL or TCS channels.
- DAQ2EVM ... status bits from the rop_daq chip

Output signals:
- EVM_STATUS (see above)
- ERR_LED to show error on the front panel
- Monitoring counters to be read by software
- Several status signals going to test point logic to be sent to the front panel Lemo connector and monitored by an oscilloscope.

1.3.5 FDL Core

Each Algo and Technical Triggerbit goes first to a

- **PRESCALER nn bits**:

Each downscaled trigger bit goes to

- **a RATE COUNTER and to**
- **the RINGBUFFER for readout**.

- **RATE COUNTER: nn bits**
  - The counter stops counting at ‚FF…FF' indicating an overflow.
  - Selection between 8 different refresh periods is possible. Normally refresh is done with every new Luminosity Segment.
  - The refresh signal saves the actual counter content into a register and clears the counter afterwards (in the same clock cycle). Then the software is responsible to read the counter register before the actual luminosity segment is over.

The FDL chip contains 128 Algo- and 64 Technical Trigger (TTrig) slices. A slice consists of a prescaler, a rate counter and the Mask/Veto logic and sends the pre-scaled and masked trigger bit to 8 final ORs circuits.
A SLICE is configured by a:

- **PRESCALE_FACTOR register:** A 'nn' bit number used for setting the factor of the prescaler. When loaded for instance with the value 3, each fourth Algo bit appearing at the input of the slice is passed through. Loaded with a value of 0 each Algo bit is passed through. Whenever a new set of prescale factors have been loaded then the command pulse 'apply_new_prescale_values' should be sent to switch simultaneously to the new prescale values.
- **SLICE register containing:**
  - **FINOR_MASK:** An 8 bit number acting as a mask for enabling an Algo or TTrig bit in one or more of the 8 FINOR circuits.
  - **EN_VETO_MASK:** An 8 bit number for defining the corresponding TTrig bit as a veto bit in one or more of the 8 FINOR circuits. When defined as a veto bit a TTrig bit can suppress a trigger to be set.

The VETO MASK is implemented only for TTRIG bits and not for ALGO bits.

- **UPDATE Period register:** By default the **'new_luminosity_segment'** signal of the FDL chip is taken to update the counter registers and to reset the counters. For other options see description of this register and of the **UPDATE_STEP_SIZE** register.

1.3.6 Final OR circuits

There are 8 Final OR circuits which have basically two tasks:

- All enabled (using FINOR_MASK) Algo and TTrig bits are reduced to one bit by a large OR-function.
- All TTrig bits which are enabled for being used as veto bit (using EN_VETO_MASK)

suppress the final OR output signal.

1.3.7 PRESCALER and Rate Counter Table

See also Prescale Factors below.

Version 1012	Rate counter range (bits)	Max Prescaled Rate (120 s Lum-segment)	Prescale Factor Range (bits)	➔ Max. Reduction
Tech. Triggers 0 … 63	20	8.7 kHz	16	40 MHz ➔ 610 Hz
Algo 0 … 7	24	139 kHz	20	40 MHz ➔ 40 Hz
Algo 8 … 127	24	139 kHz	18	40 MHz ➔ 153 Hz
NoAlgo	24	139 kHz	24	40 MHz ➔ 2,4 Hz

For 90s Luminosity Segments the maximum prescaled rate can be increased accordingly.

1.3.8 PRESCALER and Rate Counter firmware behavior

See also Prescale Factors below.

-- 1.) New PRESCALE FACTORS
-- a) Load a new prescale factor into 'vme_reg' by vme-software.
-- b) The vme_common_command_pulse 'apply_new_presc_value' sets the request_ff =1.
-- c) The next 'rate_cntr_update' signal (default:= 'new_luminosity_segment' pulse from FDL)
-- ==> clears the request_ff
-- ==> clears the pre_counter (to restart with the new prescale factor)
-- ==> moves the new prescale factor from the 'vme_reg' into the 'prescale_reg'
-- 2.) Prescaling and Rate Counter
-- a) The next trig signals increment the pre_counter until the
-- new prescale value and the counter are equal (limit =1).
-- b) The next trigger passes through (en_trigger=1),
-- ==> increments the rate counter and
-- ==> clears the pre_counter again.
-- //Loop: 2a==>2b==>2a==>2b==> ...

Remark: *Prescale_factor =0 → every trigger passes through.*
Prescale_factor =1 → every 2nd trigger passes through.
Prescale_factor =2 → every 3rd trigger passes through.

-- 3.) Rate counter readout:
-- Every 'cntr_update" signal (default:= 'new_luminosity_segment' pulse from FDL)
-- ==> moves the actual rate counter value into a rate_register and
-- ==> clears the rate counter.
--
-- *** The update is done for all Algo and Technical Trigger bits concurrently. ***

1.3.9 PRE_SCALER Software Guide

Firmware version V1011 (Aug 2008) :

Software procedure before starting data taking:

- Check if the LUMINOSITY SEGMENT PERIOD has been loaded correctly = **X"000A4678"** ← ~ 60 sec or other value
- Load the UPDATE PERIOD register for all ALGO and Technical Triggers with =**X"00000002"** ← **Select 'new_luminosity_segment' signal of FDL chip**
- Load the new PRESCALE FACTORS for all ALGO and Technical Triggers
- Load the new PRESCALE VERSION number.
- Send the command pulse 'apply_new_prescale_values'.
- Start the RUN : If the signal RESET_ORBIT_COUNTER (OCR) is sent then the luminosity segment counter will be reset in the FDL chip.

Software procedure during data taking::

- Load the new PRESCALE FACTORS for all ALGO and Technical Triggers
- Load the new PRESCALE VERSION number.
- Send the command pulse 'apply_new_prescale_values'.

➔ At begin of the next luminosity segment the pre_scaler are cleared and the new set of PRESCALE FACTORS will become active and also the new PRESCALE VERSION number will be included simultaneously into the FDL event records.

REMARK: As long as the pre_scale values are not changed by the command pulse* 'apply_new_prescale_values'** ***the pre_scalers ignore the 'new_luminosity_segment' signals.

1.3.10 SIM/SPY memories

FDL runs with BCRES only.
Modes to run the SIM/SPY memories for Algo/TTRIG bits and the FinOR signals.

- SIM one orbit: reads until END of MEMORY ignoring following BCRES signals
- **SIM continuously: restart reading from begin of mem. with every BCRES signal**
- **SPY one orbit: write until END of MEMORY ignoring following BCRES signals**
- SPY continuously: restart writing into begin of mem. with every BCRES signal

When spying one orbit, writing into the memory starts with the next BCRES but it will be filled until address 4095 not returning to 0 at the end of anLHC orbit.

'SIM_ one orbit' might be used with 'SPY_ one orbit' to check if exactly data from this orbit were written into the spy memory.

1.3.10.1 How to run the SIM/SPY memories during tests

FDL internal test:

- Algo/TTRIG mem: 'sim_continuously', FINOR mem: 'spy_one_orbit'
- Algo/TTRIG mem: 'sim_one_orbit', FINOR mem: 'spy_one_orbit'

GTL to FDL test:

- Algo/TTRIG mem: 'spy_one_orbit', FINOR mem: 'spy_one_orbit'

FDL to TCS test:

- Algo/TTRIG mem: 'sim_continuously', FINOR mem: 'spy_one_orbit'
- Algo/TTRIG mem: inactive, FINOR mem: 'sim_continuously''

1.3.10.2 BCRES delay for SIM/SPY

BCRES delay: =0 ➔ SIM_address=0 is applied after 3rd clk edge (after bcres signal)
BCRES delay: =0 ➔ SPY_address=0 is applied after 4th clk edge
Maximum BCRES delay = 15 bx.

1.3.11 Readout processor FDL_ROP

The readout processor FLD_ROP sends a data record via 2 channel link interfaces to the EVM as well as to the DAQ interface on the Global Trigger Front End board (GTFE). Each time the FDL receives a L1A signal the ROP starts extracting data from the Ringbuffer memories, combines all data and sends a formatted record to the GTFE board. The Table below shows the format.

C-word: = BCnr when reading from the Ringbuffer

E-word: = localBCnr when writing into the Ringbuffer ➔ to check LATENCY value

PRESCALE_VERSION(15:0) = Version# for ALGO prescale factors

PRESCALE_VERSION(31:16) = Version# for TECHNICAL_TRIGGER prescale factors

27-24 (ID)	23 –16 (mon)	15 -0	Comment
A		EVNR(15:0)	*1st BC data*
B		X“00“, EVNR(23:16)	*1st BC data*
C		bx_in_event(3:0), BCNR(11:0)	*1st BC data*
D		BOARD_ID	*1st BC data* (X“FD0A“)
1		TECH_TRIG (15:0)	*1st BC data*
1		TECH_TRIG (31:16)	*1st BC data*
1		TECH_TRIG (47:32)	*1st BC data*
1		TECH_TRIG (63:48)	*1st BC data*
1		ALGO (15:0)	*1st BC data*
1		ALGO (31:16)	*1st BC data*
1		ALGO (47:32)	*1st BC data*
1		ALGO (63:48)	*1st BC data*
1		ALGO (79:64)	*1st BC data*
1		ALGO (95:80)	*1st BC data*
1		ALGO (111:96)	*1st BC data*
1		ALGO (127:112)	*1st BC data*
1		ALGO (143:128)	*not implemented*
1		ALGO (159:144)	*not implemented*
1		ALGO (175:160)	*not implemented*
1		ALGO (191:176)	*not implemented*
1		„0000_000“ & NoAlgo & FINOR(7:0)	*1st BC data*
1		X“0000“	*1st BC data*
1		PRESCALE_VERSION(15:0)	*1st BC data*
1		PRESCALE_VERSION(31:16)	*1st BC data*
E		„0000“ & localBCNr(11:0)	*1st BC data*
E		LUM_SEGM_NR(15:0)	*1st BC data*
E		ORBIT_NR(15:0)	*1st BC data*
E		ORBIT_NR(31:16)	*1st BC data*
A		EVNR(15:0)	*2nd BC data*
B		X“00“, EVNR(23:16)	*2nd BC data*
C		bx_in_event(3:0), BCNR(11:0)	*2nd BC data*
D		BOARD_ID	*2nd BC data*
1		TECH_TRIG (15:0)	*2nd BC data*
		...	*2nd BC data*

1		PRESCALE_VERSION(15:0)	*2nd BC data*
1		PRESCALE_VERSION(31:16)	*2nd BC data*
E		„0000“ & localBCNr(11:0)	*2nd BC data*
E		LUM_SEGM_NR(15:0)	*2nd BC data*
E		ORBIT_NR(15:0)	*2nd BC data*
E		ORBIT_NR(31:16)	*2nd BC data*
A		EVNR(15:0)	*3rd BC data*
B		X“00“, EVNR(23:16)	*3rd BC data*
C		bx_in_event(3:0), BCNR(11:0)	*3rd BC data*
D		BOARD_ID	*3rd BC data*
1		TECH_TRIG (15:0)	*3rd BC data*
		...	*3rd BC data*
1		PRESCALE_VERSION(15:0)	*3rd BC data*
1		PRESCALE_VERSION(31:16)	*3rd BC data*
E		„0000“ & localBCNr(11:0)	*3rd BC data*
E		LUM_SEGM_NR(15:0)	*3rd BC data*
E		ORBIT_NR(15:0)	*3rd BC data*
E		ORBIT_NR(31:16)	*3rd BC data*
F			*End of board record*
5	IDLE code	IDLE-code	*Space between records*

Channel Link bits 27 – 24: Header code used by the GTFE board to assemble GT events.
A-, B-, C-, D- and E words: Control data
1- words: Trigger data
F- word: End of record
5 – word: IDLE code between data records.
Channel Link bits 23 – 16: actually not used; could be used for monitoring data
Channel Link bits 15 - 0: data bits

The readout controller can be configured sending information either from 3 or 5 bunch crossings (sel_long bit in GENERAL_REGISTER). A delayed BCRes Signal is used to reset the ring buffer address counters. The Board Identifier as well as the delay for reading from and writing into the ring buffer can be set in the Board ID and Latency Delay register (see register table). The idle code that is sent between records can be set in the IDLE_CODE register.
The format of the channel link record is foreseen for 256 slices but only 192 slices can be implemented in the XC2V4000 chip.

1.3.12 SIM/SPY memories

The ‘simspy’-memories on the chip can be set either to simulation or to spy mode using bits of the GENERAL REGISTER. The spy mode for the ALGO and TTRIG memories is used to monitor the transfer from the GTL and the PSB board. One can choose also between spying continuously and spying just one orbit beginning with the next BCRes.
In SIM mode data stored in the ALGO/TTRIG memories will be transferred through the slice modules to the final OR circuits and can be spied by the FINOR_simspy_memory (=internal test). These simulated FINOR signals can be forwarded by the TCS board to generate L1A signals.
Alternatively the FINOR_simspy_memory can be set also to simulation mode sending simulated FINOR signals directly to the TCS board.

1.3.13 Timing signals

BCRes, L1Res, and L1A are decoded as shown in the TIM Chip documentation.
BCRES: If the board runs without a TIM6U module, the BCRES signal is generated automatically as defined by the 'ORBIT LENGTH REGISTER-1'.

1.3.14 NOALGO Slice

If no Algo bit for a given bunch crossing is valid then the GTL board sends so called 'NO-ALGO' bits to be used for tests and checking trigger data. All NoAlgo input bits are combined by an OR function that acts as input bit for the NoAlgo-slice.
The NoAlgo Slice contains a slice for a no Algo condition which is slightly different to the other slices. The NoAlgo input bit can also be spied using bit 8 of the finor spy memory. The prescaler as well as the 'no_algo' rate counter length have been extended to 24 bits to reduce the expected 40 MHz rate down to a few Hz.

1.3.15 Status Merger

The FDL board receives encoded STATUS bits from other GT boards and combines them to a common GT_STATUS that is sent to the TCS board. The 'General register' bits 28…16 allow disabling the status of boards when merging. A disabled board is set to 'ready'. All enabled boards have to be ready to forward 'ready' to TCS!

4 bit STATUS CODE:

0=F = disconnected
1 = warning
2 = out_of_sync
4 = busy
8 = ready
C = error
Other codes = bad_code

Priority chain when merging:

Bad_code > disconnected > error > out_of_sync > busy > warning > ready

1.4 FDL Chip Registers and Memory Description and Addresses

1.4.1 Address space overview

The address structure is defined for optional upgrade to 192 Algo bits. Actually logic for 128 Algo bits has been implemented with a XC2V3000 chip.

RED= inside Readback memory BLUE= address as in old Version
GREEN = new register PINK = changed address

19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	SIM / SPY MEM (32 x 4k) *TTRIG31-0*												0	0
0	0	0	0	0	1	SIM / SPY MEM (32 x 4k) *TTRIG63-32*												0	0
0	0	0	0	1	0	SIM / SPY MEM (32 x 4k) *ALGO31-0*												0	0
0	0	0	0	1	1	SIM / SPY MEM (32 x 4k) *ALGO63-32*												0	0
0	0	0	1	0	0	SIM / SPY MEM (32 x 4k) *ALGO95-64*												0	0
0	0	0	1	0	1	SIM / SPY MEM (32 x 4k) *ALGO127-96*												0	0
0	0	0	1	1	0	SIM / SPY MEM (32 x 4k) *ALGO159-128*												0	0
0	0	0	1	1	1	SIM / SPY MEM (32 x 4k) *ALGO191-160*												0	0
0	0	1	0	0	0	FINOR SPY MEM **(9 x 4k)**												0	0
19	18	17	16	15 - 12		11 - 10	9	8	7	6	5	4	3	2				1	0
0	*0*	*1*	*0*	*0100*		*00*	*SLICE_REG (256x32bit) (rw)*											*0*	*0*
0	*0*	*1*	*0*	*0100*		*01*	*UPDATE_PERIOD (256x32bit) (rw)*											*0*	*0*

0	*0*	*1*	*0*	*0100*	*10*	*PRESCALE_REG (256x32bit) (rw)* *4 regs implemented with 32 bits*	*0*	*0*
0	*0*	*1*	*0*	*0100*	*Bits 11-5: 1100_000*	*UPDATE CMD PULSes (8x32bit) (r*w)*	*0*	*0*
0	*0*	*1*	*0*	*0100*	*Bits 11-6: 1100_01*	*COMMON REGS (10 used of 16*32bit) (wr)*	*0*	*0*
0	0	1	0	1000	0000_000 STATUS REGISTERS *(4x32) (r)*		0	0
0	0	1	1	0000	00	RATE REGISTERS (256x32bit)(r)	0	0

24000 .. 24FFC … 32bit readback memory for w/r registers
r* … last cmd pulse can be read back.

1.4.2 Register Overview

32 and 16 bit words

	UPDATE COMMAND PULSES	bits used
2 4c00	Update cmd bits (31…0)	32 bits
2 4c04	Update cmd bits (63…32)	32 bits
2 4c08	Update cmd bits (95…64)	32 bits
2 4c0c	Update cmd bits (127…96)	32 bits
2 4c10	Update cmd bits (159…128)	32 bits
2 4c14	Update cmd bits (191…160)	32 bits
2 4c18	Update cmd bits (223…192)	32 bits
2 4c1c	Update cmd bits (255…224)	32 bits
	COMMON REGISTERS	# valid bits
2 4c40	update_step	16 bits
2 4c44	general register	32 bits
2 4c48	cmd_pulses	8 bits
2 4c4c	idle code	32 bits
2 4c50	board_id+latency_delay	32 bits
2 4c54	orbit length	16 bits
2 4c58	route_sig2pan	32 bits
2 4c5c	noalgo_prescale_factor	32 bits
2 4c60	noalgo_setup	32 bits
2 4c64	noalgo_rate_reg	32 bits
2 4c68	lum_segm_period	32 bits
2 4c6c	prescale_version	32 bits
	2 4c6c-24c7c not used	
	STATUS REGISTERS	
2 8000	chip_id_reg	32 bits
2 8004	version_reg	32 bits
2 8008	fdl_status_reg	16 bits
2 800c	Slice_count_reg	32 bits
	2 8010 … 2 801c not used	

1.4.3 SIM/SPY MEMORIES 8

Addresses 0x00000 – 0x20000

Implementation

- 64 TTrig: base addr: 0x00000, 0x04000
- 128 Algos: base addr: 0x08000, 0x0C000, 0x10000, 0x14000
  (Algo128-191: 0x18000, 0x1C000)
- FINOR mem: base addr: 0x20000

1.4.4 256 SLICE Registers (rw) 0x24000 – 0x243FC (256x32 bits)

256 registers
One Slice Register exists for each Algo or TTrig:

Slices 63 - 0 for TTRIG 63 - 0
Slices 191 – 64 for Algo 127 - 0
Slices 255 – 192 for Algo 191 -128 …not implemented

31 - 24	23 - 16	15 - 8	7 - 0
----------	----------	EN_VETO_MASK	FINOR_MASK

Power-up values: X"0001" → all slices go to Finor0, without any 'veto-ing'

EN_VETO_MASK:
An EN_VETO_MASK bit enables the TTRIG bit to inhibit ('veto') the corresponding FINOR signal.
Bit 0: enable veto for finor 0
Bit 1: enable veto for finor 1
Bit 2: enable veto for finor 2
…
Bit 7: enable veto for finor 7

FINOR_MASK:
A FINOR_MASK bit connects the TTRIG/ALGO trigger bit to the corresponding FINOR signal.
Bit 0: enable bit for FINOR 0
Bit 1: enable bit for FINOR 1
Bit 2: enable bit for FINOR 2 …

0x24000 Slice 0 //ttrig0
0x24004 Slice 1 //ttrig1
……..
0x240FC Slice 63 //ttrig63
0x24100 Slice 64 //algo0
…….
0x242FC Slice 191 //algo127
…….
0x243FC Slice 255 //algo191

…

1.4.5 256 Update Period Registers (rw) 0x24400 – 0x247FC (256x32 bits)

256 registers for 256 slices.
Power-up value: = X"00000002" → *Select* ***'new_luminosity_segment'*** *signal of FDL*
One Update Period Register exists for each Algo or TTrig:

Slices 63 - 0 for TTRIG 63 - 0
Slices 191 – 64 for Algo 127 - 0
Slices 255 – 192 for Algo 191 -128

31 - 24	23 - 16	15 - 8	7 - 0
----------	----------	----------	UPDATE PERIOD

The update period based on the time base set in the update step register can be chosen as follows:

UPDATE PERIOD	Corresponding timescale factor
1	Select **'new_luminosity_segment'** signal of TCS board
2	Select **'new_luminosity_segment'** signal of FDL
4	16 * (update_step_size+1) [BC]
8	2.048* (update_ step_size+1) [BC]
16 =hex 10	8.192* (update_ step_size+1) [BC]
32 = hex 20	16.384* (update_ step_size+1) [BC]
64 = hex 40	131.072* (update_ step_size+1) [BC]
128= hex80	2.097.152* (update_ step_size+1) [BC]

Only one out of 8 bits should be selected.
With bit 0 or 1 the programmed length of the luminosity segment is selecected.
For bits 2 to 7 execute the formula in the table above to find the length of the update period.

1.4.6 256 PRESCALE FACTOR (rw) 0x24800 – 0x24BFC (256x32 bits)

256 registers: 32 bits wr/ rdback from memory
For each TechnicalTrigger and Algo bit exists a Prescale Factor Register.
Power-up value: = X"00000000" → no prescaling
Max. prescale factors:
40 MHz → 2.4 Hz : 24 bit prescale factor
1 MHz → 1 Hz : 20 bit prescale factor
250 kHz → 1 Hz : 18 bit prescale factor
1 MHz → 4 Hz : 18 bit prescale factor
~60 kHz → 1 Hz : 16 bit prescale factor

Register Name	Trigger Name	# bits implemented	raw rate to	final rate
PRESCALE_FACTOR_SLICE_0	TechTrig_0	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_1	TechTrig_1	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_2	TechTrig_2	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_3	TechTrig_3	16	60 kHz	→ 1 Hz
.....	TechTrig_xx	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_61	TechTrig_61	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_62	TechTrig_62	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_63	TechTrig_63	16	60 kHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_64	Algo_0	20	1 MHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_65	Algo_1	20	1 MHz	→ 1 Hz
.....	Algo_xx	20	1 MHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_71	Algo_7	20	1 MHz	→ 1 Hz
PRESCALE_FACTOR_SLICE_72	Algo_8	18	1 MHz	→ 4 Hz
PRESCALE_FACTOR_SLICE_73	Algo_9	18	1 MHz	→ 4 Hz
.....	Algo_xx	18	1 MHz	→ 4 Hz
PRESCALE_FACTOR_SLICE_190	Algo_126	18	1 MHz	→ 4 Hz

PRESCALE_FACTOR_SLICE_191	Algo_127	18	1 MHz	→ 4 Hz
PRESCALE_FACTOR_SLICE_192	Algo_128	---	---	
.....	.....	---	---	
PRESCALE_FACTOR_SLICE_ 255	Algo_191	---	---	

PRESCALE FACTOR = value -1 ➔

0 … no prescaling (every trigger is forwarded) (= nr of suppressed trig's)
1 … every second trigger is forwarded (= nr of suppressed trig's)
2 … every third trigger is forwarded (= nr of suppressed trig's)

Send Common Command Pulse bit 8: 'apply_new_prescale_values' after having loaded a new prescale factor. By the same software procedure also the 'prescale_version' register should be updated.

1.4.7 Update Command Pulses (w) 0x24C00 – 0x24C1C (8x32 bits)

NOT USED SINCE V100B

8 registers

For each slice one 'update command bit' is defined.

The software should send Update cmd pulses every time a prescale factor has been loaded or changed.

Cmd bit number = slice number

Address	bit 31 ..... bit 0						
24c00	31	30	…	Update cmd bits (31…0)	…	1	0
2 4c04	63	62	…	Update cmd bits (63…32)	…	33	32
2 4c08	95	94	…	Update cmd bits (95…64)	…	65	64
2 4c0c	127	126	…	Update cmd bits (127…96)	…	97	96
2 4c10	159	158	…	Update cmd bits (159…128)	…	129	128
2 4c14	191	190	…	Update cmd bits (191…160)	…	161	160
2 4c18	223	222	…	Update cmd bits (223…192)	…	193	192
2 4c1c	255	254	…	Update cmd bits (255…224)	…	225	224

1.4.8 Common registers

1.4.8.1 Update Step Size Register (rw) 0x24C40 (16 bits)

15	0
Update Step Size	

Calculation: N=(desired time base in NS)/25NS -1

The Update Step Size Register contains a timebase common to all slices. Its value is used to specify the length of the timebase for update commands for the rate counters. This value is multiplied by a factor which can be set individually for each slice using the UPDATE PERIOD register to get individual update periods for each slice when testing. For data taking normally the 'new_luminosity_segment' signal will be used.

1.4.8.2 General Register (rw) 0x24C44 (32 bits)

General Register

Power-up value: X"0000 0710" // 28-16: status of all boards enabled,
// 15-0: en_chan_d,_e; en_backrecv; fdl_status=disconn for TCS,
// use_BCRes_dly; send 3bx events

Bit 0: sim_mode_algo: 0: spy mode (for Algo and TTrig bits) (was: sel_sim)
1: simulation mode (for Algo and TTrig bits)
Bit 1: contin_mode_algo 0: run 1 orbit start with next bcres (was: spy_mode)
1: run synchronous with each bcres
Bit 2: sel_long 0: ROP sends 3 BX
1: ROP sends 5 BX
Bit 3: unused (was: sel_warn50)
Bit 4: unused
Until V1011: use_BCRes_dly 0: Bypass bcres delay circuit
1: Use bcres delay
Bit 5: not used
Bit 6: fdl_status0 encoded FDL status bit 0
Bit 7: fdl_status1 encoded FDL status bit 1
00 = disconnected, 11 = disconnected
01= busy, 10 = ready
Bit 8: en_backrecv 1: enables receivers for backplane signals
Bit 9: en_chan_e 1: enables channel link chip for GTFE:EVM chip
Bit 10: en_chan_d 1: enables channel link chip for GTFE:DAQ chip
Bit 11: en_freeze 1: enables freeze mon commands coming from TCS (not used)
Bit 12: sim_mode_finor 0: spy mode (for FinOR bits)
1: simulation mode (for FinOR bits)
Bit 13: contin_mode_finor 1: send/spy data continuously 0: send/spy 1 orbit
Bit 14: tcs_stops_cntrs 1: Calibration Cycle from TCS can stop RATE Counters
Default = '1'
Bit 15: en_robus 1: enables ROBUS signals

1.1.1.1.1. Disable Slot Status for TCS:

Bit 16: disable status signals of slot 9 (PSBT)
Bit 17: disable status signals of slot 11 (GTL1)
Bit 18: disable status signals of slot 12 (GTL2)
Bit 19: disable status signals of slot 13 (PSB1)
Bit 20: disable status signals of slot 14 (PSB2)
Bit 21: disable status signals of slot 15 (PSB3)
Bit 22: disable status signals of slot 16 (TIM) // TIM status arrives via TIMFDL(3:0)
Bit 23: disable status signals of slot 17 (GTFE)
Bit 24: disable status signals of slot 18 (GMT)
Bit 25: disable status signals of slot 19 (PSB4)
Bit 26: disable status signals of slot 20 (PSB5)
Bit 27: disable status signals of slot 21 (PSB6)
Bit 28: disable status signals of slot 10 (FDL)
Bit 29 – 31: not used

*1.4.8.3 Common Cmd Pulse (r*w) 0x24C48*

(command pulses)

Common Command Pulses

Bit31-9 not used
Bit 8: apply_new_prescale_values
// Apply the prescale values which have been loaded into the prescale factor registers to the prescalers and use the new 'prescale_version' number at begin of the next luminosity segment.

Bit 7: fdl_reset_orbnr // reset the orbit number by VME
Bit 6: fdl_reset_evnr // reset the event number by VME
Bit 5: fdl_reset_bc_error // reset the BC-ERROR flag
Bit 4: start simspy_finor // with next bc reset.
Bit 3: ~~load_noalgo_prescaler~~ …not used in V100B
Bit 2: fdl_resynchronize // Resynchronize the logic by VME
//(TCS: =L1RESET or RESYNC)
//(old: reset out_of_sync);
Bit 1: fdl_reset_rop_error // reset the error in the fdl_rop module
Bit 0: start simspy_ttrig_algo //with next bc reset.

1.4.8.4 Idle Code Register (wr) 0x24C4C (32 bits)

Defines IDLE code for Channel Link, that is sent between events.
Default value: X"05555555" …. is expected by GTFE board.

1.4.8.5 Board ID, Latency Delay Register (rw) 0x24C50 (32 bits)

31 16	15 0
Board ID	Latency Delay

Bit 31-16: Board ID for channel link record: **Default value:= "FD0A"**
Bit15-12: not used
Bit 11-8: write_delay // adjusts writing into the Ringbuffer and Spy memory
Bit 7-4: read_delay2 // adjusts reading from the Ringbuffer for L1A signals
Bit 3-0: read_delay1 // adjusts reading from the Ringbuffer for L1A signals
- **unit** = BC(bunchcrossings) ~ 25 ns
- **write_delay +3** defines the time when the BCRES signal resets the write-address counter for the Ring-buffer. The adjustment is correct if Algo bits of BC0-trigger data are being written into the first address of the Ringbuffer.

write_delay + 3 = BCRES delay for write address counter

maximum value: 15 + 3 (BC)

(read_delay1 + read_delay2 +1) defines the time when the BCRES signal resets the read-address counter for the Ring-buffer. The adjustment is correct if the trigger data extracted from the ring-buffer have generated the returning L1A signal. The time between a Fin_OR signal and the returning L1A signal is called "local_L1A_latency". If the L1A signal returns from the TCS board via the GT-backplane then the local_L1A_latency = 0 + 2 +1 BC (bits 7-0: = 02 hex).

read_delay1 + read_delay2 + 1 = local_L1A_latency

maximum value: 15 + 15 + 1 (BC)

VHDL: ROP/delay_fdl_rop: V1012

1.) 2 FF were added after the bcres_delay to write trigger data concurrently into the Ring_buffer and the SPY memory.
2.) The delayed bcres signal is then applied to the latency delay units so that even for a latency = 0 the read_address counter is always reset after the write_address counter thus avoiding concurrent read and write access at the same memory location.

Delay scheme:

```
-- BCRES→dly(11:8)→FD → FD → res_wr_addr
--                  |→dly(7:4)→dly(3:0)→ → FD → res_rd_addr(5bx event)
--                                    |→ res_rd_addr(3bx event)
```

1.4.8.6 Orbit Length Register (rw) 0x24C54 (16 bits)

15	11	0
xxx	Orbit Length -1	

The register defines the LHC-Orbit Length in the FDL board.
If this number does not correspond to the BCRES signal from the TIM board, an out_of_sync error will be set.
Default value: X"0DEB" (hex)

1.4.8.7 Route Signal2Pan Register (rw) 0x24C58 (32 bits)

31- 16	15 - 8	7 - 0
Show TEST SIGNALS at → "FINOR" on frontpanel	Show FINOR7,6,….0 at → "FINOR" on frontpanel	Show TTRIG and ALGO bits at → "TTRIG" **behind** the frontpanel

Power-up value: X"0000_0100" ...show FinOR0
The selected signal(s) can be monitored on the front panel LEMO connector by an oscilloscope.

Show Test Signals at FrontPanel_LEMO "FINOR":
Bit 31 – 22: free for other signals
Bit 21: =1: show 'res_orbnr' … internal signal resets the orbit couter
Bit 20: =1: show 'new_prescale_version' … internal signal applies new prescale version
Bit 19: =1: show 'BCRES_i' … decoded BCRES signal from TIM board
Bit 18: =1: show 'L1A' … decoded L1A signal from TIM board
Bit 17: =1: show 'new_luminosity_segment_tcs' … signal from TCS board
Bit 16: =1: show 'new_luminosity_segment_fdl' … signal increments lum_segm counter

If several bits are set then the 'OR' of the signals will be seen.
Bit 15: =1: show 'FINOR(7)' … Final_OR signal forwarded to TCS_PTC7
Bit 14: =1: show 'FINOR(6)' … Final_OR signal forwarded to TCS_PTC6
Bit 13: =1: show 'FINOR(5)' … Final_OR signal forwarded to TCS_PTC5
Bit 12: =1: show 'FINOR(4)' … Final_OR signal forwarded to TCS_PTC4
Bit 11: =1: show 'FINOR(3)' … Final_OR signal forwarded to TCS_PTC3
Bit 10: =1: show 'FINOR(2)' … Final_OR signal forwarded to TCS_PTC2
Bit 9 : =1: show 'FINOR(1)' … Final_OR signal forwarded to TCS_PTC1
Bit 8: =1: show 'FINOR(0)' … Final_OR signal forwarded to TCS_PTC0

Show ALGO resp. Technical Trigger bits at LEMO "TTRIG"behind the Frontpanel
Bits 7 – 0 : Select one of 256 slices:
Values 0 - 63 show Technical Triggers 0 … 63;
Values 64 – 191 show Algo_bits 0 … 127
Values192 – 255 not implemented

1.4.8.8 NoAlgo Prescale Factor Register (rw) 0x24C5C (32 bits)

31	0
Prescale Factor NoAlgo	

Power-up value: **X"00061A7F"** = 400.000 // **40 MHz → 100 Hz**
The NO_ALGO trigger =1 when all ALGO bits are =0. Therefore a very high prescale factor is required to decrease the rate from about 40 MHz to a reasonable value.
40 MHz → 2.4 Hz : 24 bit prescale factor

PRESCALE FACTOR:

0 … every trigger is forwarded
1 … every second trigger is forwarded
2 … every third trigger is forwarded

1.4.8.9 NoAlgo Setup Register (rw) 0x24C60 (32 bits)

31	24	23	16	15	8	7	0
en_no_algo mask		Finor mask		xxx		Update period	
Default: 0 0		**0 0**		**0 0**		**X“02“**	

EN_NO_ALGO MASK is used to combine up to 6 algo_s signals to one NO_ALGO signal.

-- 1) The GTL board sends an **algo_s(i)** = 1 if all 32 algo bits of connector(i) are =0.
-- 2) A disabled algo_s does not inhibit other enabled algo_s signals.
-- 3) All enabled algo_s have to be =1 to get a ‘no_algo’ trigger signal.
-- 4) At least one algo_s must be enabled.

The No_Algo signal goes also to bit9 of Finor_SimSpy_memory.

The Algo_s(5) arrives from Connector 5 that carries Algo 191-160 . NOT USED
The Algo_s(4) arrives from Connector 4 that carries Algo 159-128 . NOT USED
The Algo_s(3) arrives from Connector 3 that carries Algo 127 - 96 .
The Algo_s(2) arrives from Connector 2 that carries Algo 95 - 64 .
The Algo_s(1) arrives from Connector 1 that carries Algo 63 - 32 .
The Algo_s(0) arrives from Connector 0 that carries Algo 31 - 0 .
Bit 29 : 1 = includes Algo_s(5) to No_Algo bit. (default: = 0)
Bit 28 : 1 = includes Algo_s(4) to No_Algo bit. (default: = 0)
Bit 27 : 1 = includes Algo_s(3) to No_Algo bit. (default: = 0)
Bit 26 : 1 = includes Algo_s(2) to No_Algo bit. (default: = 0)
Bit 25 : 1 = includes Algo_s(1) to No_Algo bit. (default: = 0)
Bit 24 : 1 = includes Algo_s(0) to No_Algo bit. (default: = 0)

FINOR MASK(7:0)
Bits 23 – 16: A FINOR_MASK bit connects the NO_ALGO trigger bit to the corresponding FINOR signal.
Example: Bit 22 =1 connects the NO_ALGO trigger bit to FINOR(6).
UPDATE PERIOD: default = X”02”
Depending from the prescale factor select the update period to avoid any overflow of the rate counter.
Bit 1 =1 selects the ‘new_lum_segm’ signal from FDL
Bit 0 =1 selects the ‘new_lum_segm’ signal from TCS
… to save and reset the rate counters with every new luminosity segment.

1.4.8.10 NoAlgo RateCount Register (r) 0x24C64 (32 bits)

2 4c64	31-24: = 00	23-0: = NoAlgo Rate Counter	32 bits

24 bit Counter;
100 kHz prescaled no_algo trigger rate for
2.5 min → 24 bits … Luminosity segment period
Rate Counter: Reset depends from selected update period. See ‘NoAlgo Setup Register’.

1.4.8.11 Luminosity_Segment_Period (rw) 0x24C68 (32 bits)

2 4c68	31-24: = 00	23-0: =lum_segm_period	32 bits

Defines length of Luminosity Segment in the FDL board.
24 bits used
Power-up value: **X"0010_0000"** ~ 93 s
Unit: 1 LHC_orbit
Load the same value as into the TCS chip otherwise the Luminosity Segment numbers will disagree.
Remark: Maybe the LumSegmNr will be included into the FDL event record.

1.4.8.12 Prescale_Version

2 4c6c	prescale_version	32 bits

w/r
The register is loaded by software. The content is appended to the event record.
A new values becomes active at start of a new luminosity segment when the common command pulse 'apply_new_prescale_values' has been sent before.
The prescale_version should be renewed together with a new set of prescale values.

1.4.9 Status Registers

1.4.9.1 Chip ID Register (r) 0x28000 (32 bits)

FDL chip ID = 0001_6131 hex
Firmware defined value will be used by software for checking.
VHDL: *Chip ID value is defined as generics in the symbol definition of the FDL chip.*
Open Symbol ➔ Interface ➔ Generics

1.4.9.2 Version Register (r) 0x28004 (32 bits)

Each design version is identified by a version number.
Firmware defined value will be used by software for checking.
Jan 2008: Version 0000 100C hex
VHDL: *Version value is defined as generics in the symbol definition of the FDL chip.*
Open Symbol ➔ Interface ➔ Generics

1.4.9.3 FDL Status Register (r) 0x28008 (16 bits)

15	0
FDL Status Bits	

Bit 31 – 16: luminosity_segm_nr
Luminosity-Segment-Number of FDL ..depends from size of Luminosity-Segment-Period.
Bit 8-15: *not used*
Bit 11 - 8: gt_status 4 bit status code: 0=F=disconn, 1=warn, 2= out_of_sync, 4=busy, 8=ready, C=error
Bit 7: fdl_bc_error (new in V100A)
Bit 6: fdl_rop_error
Bit 5: fdl_out_of_sync
Bit 4: fdl_warn75 (new in V100A)
Bit 3: fdl_warn50 (was: fdl_warning in different bit position)
Bit 2: fdl_disconnected (set by VME)

Bit 1: fdl_busy (set by VME)
Bit 0: fdl_ready (set by VME)

1.4.9.4 Slice Count Register (r) 0x2800C (32 bits)

15 … 8	7 … 0
Number of Algo slices	Number of TTrig slices

Firmware defined values will be used by software for correct addressing of memories and registers.

VHDL comment: *Both values are defined as generics in the symbol definition of the FDL chip and are converted to logic vectors in 'embedded blocks' on page 14 of fdl_chip/behavioral as inputs to instance 'slice_count_reg' of a 'vme_status_reg' module.*

Open Symbol ➔ Interface ➔ Generics

Version 100A for XC2V4000 chip: no_of_algo = 128, no_of_ttrig = 64

1.4.10 Rate Registers (r) 0x30000 – 0x303FC (256x32bits)

256 registers:
100 kHz rate for 20 min → 27 bits
50 kHz rate for 10 min → 25 bits
100 kHz rate for 2.5 min → 24 bits … Luminosity segment period
50 kHz rate for 5 min → 24 bits
50 kHz rate for 83 sec → 22 bits
10 kHz rate for 104 sec → 20 bits … Technical Triggers

31 - 0	Trigger Name	# bits implemented
TRIGGER RATE SLICE_0	TechTrig_0	19_00
TRIGGER RATE SLICE_1	TechTrig_1	19_00
TRIGGER RATE SLICE_2	TechTrig_2	19_00
TRIGGER RATE SLICE_3	TechTrig_3	19_00
.....	TechTrig_xx	19_00
TRIGGER RATE SLICE_61	TechTrig_61	19_00
TRIGGER RATE SLICE_62	TechTrig_62	19_00
TRIGGER RATE SLICE_63	TechTrig_63	19_00
TRIGGER RATE SLICE_64	Algo_0	23_00
TRIGGER RATE SLICE_65	Algo_1	23_00
TRIGGER RATE SLICE_66	Algo_2	23_00
.....	Algo_xx	23_00
TRIGGER RATE SLICE_188	Algo_124	23_00
TRIGGER RATE SLICE_189	Algo_125	23_00
TRIGGER RATE SLICE_190	Algo_126	23_00
TRIGGER RATE SLICE_191	Algo_127	23_00
TRIGGER RATE SLICE_192	Algo_128	---
.....	.....	---
TRIGGER RATE SLICE_ 255	Algo_191	---

The Rate Register contains the **prescaled** Trigger Rate of each slice. A Trigger Rate of 0xffff indicates an overflow. To get reasonable rate numbers for high and low rates the reset interval can be defined by the Update Step Size Register. Normally the RATE COUNTERS are reset by the **'new_luminosity_segment'** signal generated either on the FDL or on the TCS board.

1.5 READOUT FORMAT

1.5.1 Format sent to by GTFE board to S-Link (DAQ, EVM)

BOARD_ID =FDL	bx_in_event(3:0), BCNR(11:0)	00hex, EVENTNR(23:16)	EVENTNR(15:0)	1st BC
TECH_TRIG(63:48)	TECH_TRIG(47:32)	TECH_TRIG(31:16)	TECH_TRIG(15 : 0)	1st BC
ALGO(63 : 48)	ALGO(47 : 32)	ALGO(31 : 16)	ALGO(15 : 0)	1st BC
ALGO(127 : 112)	ALGO(111 : 96)	ALGO(95 : 80)	ALGO(79 : 64)	1st BC
ALGO(191 : 176)	ALGO(175 : 160)	ALGO(159 : 144)	ALGO(143 : 128)	1st BC
PRESCALE_ VERSION(15:0)	PRESCALE_ VERSION(15:0)	0 0 0 0	FINOR(15 : 0)	1st BC
0 0 0 0	0 0 0 0	LUM_SEGMENT_NR	0000b localBCNR(11:0)	1st BC
BOARD_ID =FDL	bx_in_event(3:0), BCNR(11:0)	00hex EVNR(23:16)	EVNR(15:0)	2nd BC
TECH_TRIG(63:48)	TECH_TRIG(47:32)	TECH_TRIG(31:16)	TECH_TRIG(15 : 0)	2nd BC
ALGO(63 : 48)	ALGO(47 : 32)	ALGO(31 : 16)	ALGO(15 : 0)	2nd BC
ALGO(127 : 112)	ALGO(111 : 96)	ALGO(95 : 80)	ALGO(79 : 64)	2nd BC
ALGO(191 : 176)	ALGO(175 : 160)	ALGO(159 : 144)	ALGO(143 : 128)	2nd BC
0 0 0 0	0 0 0 0	0 0 0 0	FINOR(15 : 0)	2nd BC
0 0 0 0	0 0 0 0	0 0 0 0	0000b localBCNR (11:0)	2nd BC
BOARD_ID =FDL	bx_in_event(3:0), BCNR(11:0)	00hex EVENTNR(23:16)	EVENTNR(15:0)	3rd BC
TECH_TRIG(63:48)	TECH_TRIG(47:32)	TECH_TRIG(31:16)	TECH_TRIG(15 : 0)	3rd BC
ALGO(63 : 48)	ALGO(47 : 32)	ALGO(31 : 16)	ALGO(15 : 0)	3rd BC
ALGO(127 : 112)	ALGO(111 : 96)	ALGO(95 : 80)	ALGO(79 : 64)	3rd BC
ALGO(191 : 176)	ALGO(175 : 160)	ALGO(159 : 144)	ALGO(143 : 128)	3rd BC
0 0 0 0	0 0 0 0	0 0 0 0	FINOR(15 : 0)	3rd BC
0 0 0 0	0 0 0 0	0 0 0 0	0000b localBCNR(11:0)	3rd BC

1.5.2 Channel Link format sent by FDL to GTFE board

27-24 (ID)	23 –16 (mon)	15 -0	Comment
A	0 0	EVNR(15:0)	*1st BC data word 0*
B	0 0	~~BCERR(7:0),~~ EVNR(23:16)	*1st BC data BCERR missing*
C	0 0	Bx_in_event(3:0), BCNR(11:0)	*1st BC dat: word 2*
D	0 0	BOARD_ID	*1st BC dat: word 3*
1	0 0	TECH_TRIG(15 : 0)	*1st BC dat: word 4*
1	0 0	TECH_TRIG(31 : 16)	*1st BC dat: word 5*
1	0 0	TECH_TRIG(47 : 32)	*1st BC dat: word 6*
1	0 0	TECH_TRIG(63 : 48)	*1st BC dat: word 7*
1	0 0	ALGO(15 : 0)	*1st BC dat: word 8*
1	0 0	ALGO(31 : 16)	*1st BC dat: word 9*
1	0 0	ALGO(47 : 32)	*1st BC dat: word 10*
1	0 0	ALGO(63 : 48)	*1st BC dat: word 11*
1	0 0	ALGO(79 : 64)	*1st BC dat: word 12*
1	0 0	ALGO(95 : 80)	*1st BC dat: word 13*
1	0 0	ALGO(111 : 96)	*1st BC dat: word 14*
1	0 0	ALGO(127 : 112)	*1st BC dat: word 15*
1	0 0	ALGO(143 : 128)	*1st BC dat: word 16*
1	0 0	ALGO(159 : 144)	*1st BC dat: word 17*
1	0 0	ALGO(175 : 160)	*1st BC dat: word 18*
1	0 0	ALGO(191 : 176)	*1st BC dat: word 19*
1	0 0	FINOR(15 : 0)	*1st BC dat: word 20*
1	0 0	0 0 0 0	*1st BC dat: word 21*
1	0 0	PRESCALE_VERSION(15:0)	*1st BC dat: word 22*
1	0 0	PRESCALE_VERSION(31:16)	*1st BC dat: word 23*
E	0 0	0 BCNR(11:0)	*1st BC dat: word 24*
E	0 0	LUM_SEGMENT_NR	*1st BC dat: word 25*
E	0 0	ORBIT_NUMBER(15:0)	*1st BC dat: word 26*
E	0 0	ORBIT_NUMBER(31:16)	*1st BC dat: word 27*
A	0 0	EVNR(15:0)	*2nd BC data word 0*
B	0 0	BCERR(7:0), EVNR(23:16)	*2nd BC data BCERR missing*
C	0 0	Bx_in_event(3:0), BCNR(11:0)	*2nd BC dat: word 2*
D	0 0	BOARD_ID	*2nd BC dat: word 3*
1	0 0	TECH_TRIG(15 : 0)	*2nd BC dat: word 4*
1	0 0	TECH_TRIG(31 : 16)	*2nd BC dat: word 5*
1	0 0	TECH_TRIG(47 : 32)	*2nd BC dat: word 6*
1	0 0	TECH_TRIG(63 : 48)	*2nd BC dat: word 7*
1	0 0	ALGO(15 : 0)	*2nd BC dat: word 8*
1	0 0	ALGO(31 : 16)	*2nd BC dat: word 9*
1	0 0	ALGO(47 : 32)	*2nd BC dat: word 10*
1	0 0	ALGO(63 : 48)	*2nd BC dat: word 11*
1	0 0	ALGO(79 : 64)	*2nd BC dat: word 12*
1	0 0	ALGO(95 : 80)	*2nd BC dat: word 13*
1	0 0	ALGO(111 : 96)	*2nd BC dat: word 14*

1	0 0	ALGO(127 : 112)	*2nd BC dat: word 15*
1	0 0	ALGO(143 : 128)	*2nd BC dat: word 16*
1	0 0	ALGO(159 : 144)	*2nd BC dat: word 17*
1	0 0	ALGO(175 : 160)	*2nd BC dat: word 18*
1	0 0	ALGO(191 : 176)	*2nd BC dat: word 19*
1	0 0	FINOR(15 : 0)	*2nd BC dat: word 20*
1	0 0	0 0 0 0	*2nd BC dat: word 21*
1	0 0	PRESCALE_VERSION(15:0)	*2nd BC dat: word 22*
1	0 0	PRESCALE_VERSION(31:16)	*2nd BC dat: word 23*
E	0 0	0 BCNR(11:0)	*2nd BC dat: word 24*
E	0 0	LUM_SEGMENT_NR	*2nd BC dat: word 25*
E	0 0	ORBIT_NUMBER(15:0)	*2nd BC dat: word 26*
E	0 0	ORBIT_NUMBER(31:16)	*2nd BC dat: word 27*
A	0 0	EVNR(15:0)	*3rd BC data word 0*
B	0 0	BCERR(7:0), EVNR(23:16)	*3rd BC data BCERR missing*
C	0 0	Bx_in_event(3:0), BCNR(11:0)	*3rd BC dat: word 2*
D	0 0	BOARD_ID	*3rd BC dat: word 3*
1	0 0	TECH_TRIG(15 : 0)	*3rd BC dat: word 4*
1	0 0	TECH_TRIG(31 : 16)	*3rd BC dat: word 5*
1	0 0	TECH_TRIG(47 : 32)	*3rd BC dat: word 6*
1	0 0	TECH_TRIG(63 : 48)	*3rd BC dat: word 7*
1	0 0	ALGO(15 : 0)	*3rd BC dat: word 8*
1	0 0	ALGO(31 : 16)	*3rd BC dat: word 9*
1	0 0	ALGO(47 : 32)	*3rd BC dat: word 10*
1	0 0	ALGO(63 : 48)	*3rd BC dat: word 11*
1	0 0	ALGO(79 : 64)	*3rd BC dat: word 12*
1	0 0	ALGO(95 : 80)	*3rd BC dat: word 13*
1	0 0	ALGO(111 : 96)	*3rd BC dat: word 14*
1	0 0	ALGO(127 : 112)	*3rd BC dat: word 15*
1	0 0	ALGO(143 : 128)	*3rd BC dat: word 16*
1	0 0	ALGO(159 : 144)	*3rd BC dat: word 17*
1	0 0	ALGO(175 : 160)	*3rd BC dat: word 18*
1	0 0	ALGO(191 : 176)	*3rd BC dat: word 19*
1	0 0	FINOR(15 : 0)	*3rd BC dat: word 20*
1	0 0	0 0 0 0	*3rd BC dat: word 21*
1	0 0	PRESCALE_VERSION(15:0)	*3rd BC dat: word 22*
1	0 0	PRESCALE_VERSION(31:16)	*3rd BC dat: word 23*
E	0 0	0 BCNR(11:0)	*3rd BC dat: word 24*
E	0 0	LUM_SEGMENT_NR	*3rd BC dat: word 25*
E	0 0	ORBIT_NUMBER(15:0)	*3rd BC dat: word 26*
E	0 0	ORBIT_NUMBER(31:16)	*3rd BC dat: word 27*
F	F F	F F F F	*End of record: word 28*
0	IDLE code	IDLE-code	*Space between records*