# DigTOF in ASPERA/NPD

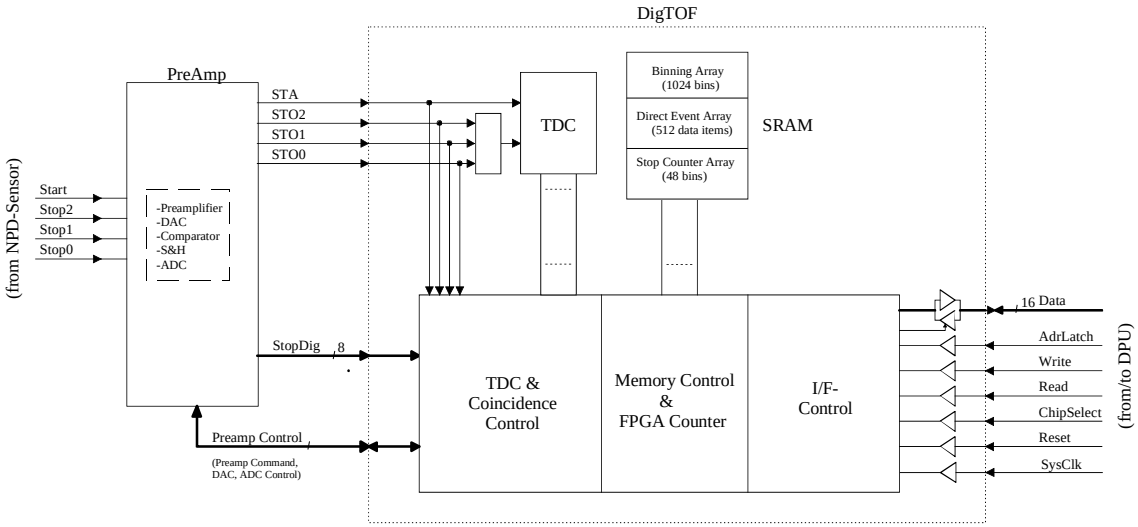
This documents contains a description of the DigTOF-electronics for ASPERA/NPD.

Version 1.0

## History

| Issue | Pg. | Date | Changes | Orig |
|---|---|---|---|---|
| 0.1 | all | 04/09/00 | first edition | cd |
| 0.2 | all | 11/06/01 | typos corrected<br>reset of single FPGA counters replaced by common reset<br>Register ADCRST added<br>flag STAT. TDCWRERR added<br>flag STAT. TDCRDERR added<br>flag STAT. TDCCMDERR added<br>flag STAT. ADCERR added<br>switch FRONTCTRL.ADCCALDIS added<br>flag MAINCTRL. LATCHUP added<br>Register LURST added<br>Register IFTEST added<br>SRAM burst readouts: exceeding of read accesses are shown now<br>coincidence changed, also default value for COINBDRS has changed<br>CALNO: precision increased<br>DACCMD: Channel numbering changed from 1,2,3,.. to A, B, C,...<br>Burst Readout: range overflow is shown by 0x5555 now<br>Appendix entries deleted<br>LATCHUP implemented | cd |
| 0.3 | All | 17/01/02 | Hamming-Code Error Correction for memory data on FM board added<br>Switch MAINCTRL.ECENA added<br>Counter SEFCCNT implemented<br>Counter DEFCCNT implemented | cd |
| 1.0 | All | 17/01/02 | numbering of document sides corrected | cd |

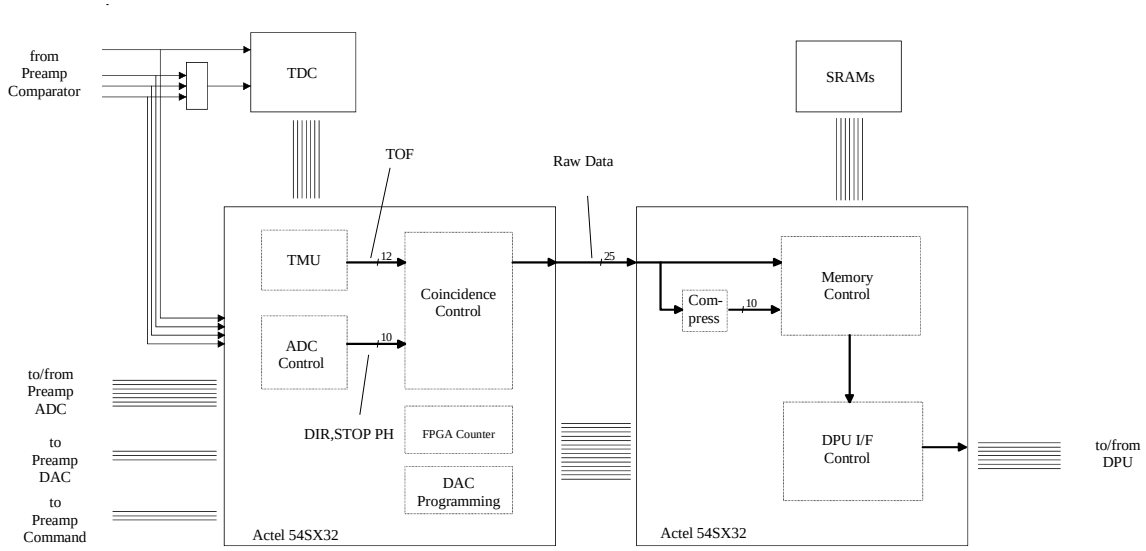
# Overview




The tasks of the DigTOF-electronics are:

- TOF measuring from one start to one out of three stop signals
- serving of the preamplifier board and initiating analog/digital conversion of stop pulseheight
- coincidence check and selection of valid TOF - Stop pulseheight pairs
- counting of events
- buffering of three different data types in SRAM
- interfacing DigTOF to the DPU

# Logic Implementation




The occurance of a start, followed by a stop signal leads to a Δt–measurement in the TDC (Time to digital converter) and a pending generation of a data item with uncorrected Δt–information. The task of the TMU (TDC management unit) is to handle the TDC, mainly read data from this and correct these values. Prior to any correction process, the necessary parameters have to be determined. This will be done with the calibration process (see calibration). The output of the TMU is a TOF value with 12 bit binary time information. In parallel to this, the occurance of a stop signal will be followed by a sampling process to the preamplifier ADC, thus a 10-bit data item with information of both direction and 8-bit stop pulseheight will be generated. The coincidence control then checks for a valid coincidence of these two data items and additionally flags the occurance of more then one start or stop signal during Δt-measurment. This now leads to a 25-bit raw data item that will be used for the pending storing process. The memory control has to handle three different memory areas in the SRAM. For the binning array, the raw data will be compressed to a 10 bit data (see compression), that represents the bin number inside the destinated array. The respective bin will be incremented by one up to the bin depth of 65536 (16 bit). For the raw data array, incoming raw data items will sequently be stored until this array is filled completely. The stop counter array will be filled in the same way as the binning array, but with the compressed stop pulseheight together with the respective direction. All data arrays will be filled in parallel (binning array will be excluded if coincidence level does not fit). Readout and the following inititalization of these arrays will be done with burst read access from the DPU. Besides the stop counter array, 16-bit event counters and two registers facing the preamplifier board are implemented in the FPGA. One of this registers will be used to program the DAC on the preamplifier board, the other for directly commanding the preamplifier electronics. All control-, counter- and memory- data will be accessed from the DPU over 16-bit registers (see registers), physically the connection to the DPU is a 16-bit bus (see description of the DPU-Interface).

# Calibration

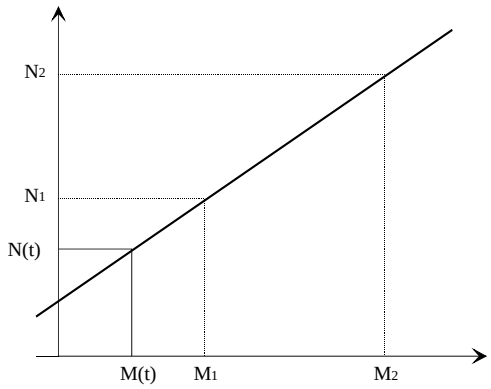
There is a linear relationship between the TDC values and the TOF, with parameters mainly depending on temperature and voltage. They have to be known to the DigTOF electronics, in order to perform correction of the TDC values. Determination of these parameters will be done by stimulation the TDC with two known time differences from a calibration clock. The resulting TDC values will than be used in the DPU to calculate gain and offset, which afterwards will be written into DigTOF. This calibration process has to be repeated within a given time to obtain a specified precision.

The goal is to obtain a binary TOF data

$$N = \frac{T}{1ns},$$

this originates from the TDC value M with following relation:

$$N(t) = O + G \cdot M(t).$$




With the measured TDC calibration values

$$M1 = CAL12 + \frac{CAL11}{16384},$$
$$M2 = CAL22 + \frac{CAL21}{16384},$$
$$N1 = \frac{TCAL}{1ns},$$
$$N2 = 2 \cdot \frac{TCAL}{1ns},$$

the gain G and offset O to transform the TOF-values can be calculated:

$$G = \frac{N2 - N1}{M2 - M1},$$

$$O = Nv - G \cdot Mv \qquad ;v = 1;2$$

With respect to the position of the comma (see registers), these two values have to be written into DigTOF.

# Data buffering

DigTOF buffers three different data items in dedicated memory areas:

1. Binning Array

This array contains 1024 bins, each with a depth of 65536 (16-bit). Data items fitting a dedicated bin will increment its counter by one up to the upper limit of 65536. Readout of this area will be done by write access to the respective register, followed by 1024 read accesses.

The bin number has following relation to its contents:

| Bin Array [1023..0] | | |
|---|---|---|
| Bit[9:8] | Bit[7:4] | Bit[3:0] |
| | | |
| DIR[1:0] | STOP PH COMPRESSED[3:0] | TOF COMPRESSED[3:0] |

(see compression)

2. Raw Data Array

This array contains 512 data places, each with a width of 32-bit. Incoming raw data items will sequently be stored until this array is filled completely. Readout of this area will be done by write access to the respective register, followed by 1024 read accesses.

Storing sequence: upper 16-bit, lower 16-bit

| Raw Data[31..0] | | | | |
|---|---|---|---|---|
| Bit[31:25] | Bit[24:22] | Bit[21:20] | Bit[19:12] | Bit[11:0] |
| | | | | |
| n.c. | COIN[2:0] | DIR[1:0] | STOP PH [7:0] | TOF [11:0] |
| | | | | |
| | 00 1 Start, 1 Stop<br>01 n Start, 1 Stop<br>02 1 Start, n Stop<br>03 n Start, n Stop<br>04 0 Start, 1 Stop<br><br>n = 2, 3, ... | 0 Stop0<br>1 Stop1<br>2 Stop2 | | in 0.5ns steps |

3. Stop Counter Array

This array contains 48 bins, each with a depth of 65536 (16-bit). Data items fitting a dedicated bin will increment its counter by one up to the upper limit of 65536. Readout of this area will be done by write access to the respective register, followed by 48 read accesses.

The bin number has following relation to its contents.

| Stop Counter Array [47..0] | |
|---|---|
| Bit[5:4] | Bit[3:0] |
| | |
| DIR[1:0] | STOP PH COMPRESSED[3:0] |

# Data Compression

To reduce the amount of data, the 12-bit TOF and the 8-bit Stop values are compressed according to the following rule:

| TOF: | TOF[11:0] | | TOF COMPRESSED[3:0] |
|---|---|---|---|
| | 100 - 124 | → | 0 |
| | 125 - 156 | → | 1 |
| | 157 - 197 | → | 2 |
| | 198 - 248 | → | 3 |
| | 249 - 311 | → | 4 |
| | 312 - 391 | → | 5 |
| | 392 - 491 | → | 6 |
| | 492 - 617 | → | 7 |
| | 618 - 774 | → | 8 |
| | 775 - 972 | → | 9 |
| | 973 - 1220 | → | 10 |
| | 1221 - 1532 | → | 11 |
| | 1533 - 1923 | → | 12 |
| | 1924 - 2413 | → | 13 |
| | 2414 - 3028 | → | 14 |
| | 3029 - 3800 | → | 15 |

| STOP: | STOP PH[7:0] | | STOP PH COMPRESSED[3:0] |
|---|---|---|---|
| | 0 - 15 | → | 0 |
| | 16 - 31 | → | 1 |
| | 32 - 47 | → | 2 |
| | 48 - 63 | → | 3 |
| | 64 - 79 | → | 4 |
| | 80 - 95 | → | 5 |
| | 96 - 111 | → | 6 |
| | 112 - 127 | → | 7 |
| | 128 - 143 | → | 8 |
| | 144 - 159 | → | 9 |
| | 160 - 175 | → | 10 |
| | 176 - 191 | → | 11 |
| | 192 - 207 | → | 12 |
| | 208 - 223 | → | 13 |
| | 224 - 239 | → | 14 |
| | 240 - 255 | → | 15 |

# DigTOF-handling / Software

The handling of the DigTOF-electronics will be done with a few macros. This chapter gives a rough overview over the steps to be made when performing special tasks. The extracts (appendix) from the EGSE S/W @ IDA give more specified information.

**Setup**

- Reset NPD electronics (H/W reset)

- Initialize Actel control register
- Initialize TDC Glob register
- Set TDC into measurement mode 1
- Write max and min borders for coincidence part
- Program DAC 0
- Program DAC 1
- Program DAC 2
- Program DAC 3
- Write Preamp Control Register
- (Perform calibration)
- Enable TDC

**Main DPU Task (Calibration+HK+Science data readout)**

- Enable TDC input for calibration
- Disable Storing into SRAM
- Read Start Counter
- Read Stop0 Counter
- Read Stop1 Counter
- Read Stop2 Counter
- Read TOF Counter
- Read RAW Counter
- Reset FPGA Counter
- Read Stop Counter array
- Set TDC into calibration mode
- Read Binning array (if this is desired)
- Read Raw data array (if this is desired)
- Read Calibration register
- Calculate Gain and Offset
- Write Gain and Offset
- Set TDC back into measurement mode 1
- Enable TDC input for measurement
- Enable Storing into SRAM

**Calibration**

- Enable TDC input for calibration
- Set TDC into calibration mode
- Wait
- Read Calibration register
- Calculate Gain and Offset
- Write Gain and Offset
- Set TDC back into measurement mode 1
- Enable TDC input for measurement

**Binning Data**

- Disable Storing into SRAM
- If desired: disable simultanously deleting of both Binning and Raw Data array (debug)
- Read Binning array
- Enable Storing into SRAM

**Raw Data**

- Disable Storing into SRAM
- If desired: disable simultanously deleting of both Binning and Raw Data array (debug)
- Read Raw data array
- Enable Storing into SRAM

**Main DPU Task**

- Disable Storing into SRAM
- Disable FPGA Counter
- Read Start Counter
- Read Stop0 Counter
- Read Stop1 Counter
- Read Stop2 Counter
- Read TOF Counter
- Read RAW Counter
- Read Stop Counter array
- Reset FPGA Counter
- Enable Storing into SRAM
- Enable FPGA Counter

Note:
- after power on, the first SRAM access delivers unvalid data and will be used to initialize the respective memory cells
- later in the flight S/W there will be one main task with performing calibration and all data readout tasks (storing into SRAM will be disabled during complete readout)

# Register

All control-, counter- and memory- data will be accessed from the DPU over 16-bit registers. Bevor accessing a destinated register, the respective address has to be written into the electronics (see description of the DPU-Interface).

| **Address** (hex) | **Read** | CS_BUS | **Write** | CS_BUS |
|---|---|---|---|---|
| | | | | |
| 0 | FRONTCTRL | F7 | FRONTCTRL | F9 |
| 1 | STAT | F8 | TDCCMD | F10 |
| 2 | TDCRD | F12 | TDCWR | F11 |
| 3 | CALIB11 | F3 | CALNG | F16 |
| 4 | CALIB12 | F4 | CALNO | F17 |
| 5 | CALIB21 | F5 | COINBDRS | F44 |
| 6 | CALIB22 | F6 | TDCRST | - |
| 7 | - | - | - | - |
| 8 | STARTCNT | F30 | CNTRST | F36 |
| 9 | STOP0CNT | F31 | ADCRST | F37 |
| A | STOP1CNT | F32 | - | - |
| B | STOP2CNT | F33 | - | - |
| C | TOFCNT | F34 | - | - |
| D | RAWCNT | F35 | - | - |
| E | - | - | DACCMD | F42 |
| F | - | - | PREAMPCTRL | F43 |
| 10 | BINARRAY | M3 | BINARRAY | M0 |
| 11 | RAWARRAY | M4 | RAWARRAY | M1 |
| 12 | STOPARRAY | M5 | STOPARRAY | M2 |
| 13 | MAINCTRL. | M7 | MAINCTRL | M6 |
| 14 | SEFCCNT | M13 | RAWDBGHIGH | M8 |
| 15 | DEFCCNT | M14 | RAWDBGLOW | M9 |
| 16 | - | - | LURST | M10 |
| 17 | IFTEST | M11 | IFTEST | M12 |

**Name:** FRONTCTRL

**Description:** This register is used to as a common control register

**Address:** see register table

**Group:** common

**Access:** Read/Write

**Note:** · see below

| Bit-Position | Name | | Description |
|---|---|---|---|
| | | | |
| 0 | TDCINENA | | TDCINENA = 1 enables attention of TDC to the input signals start, stop and calibration clock |
| 1 | TDCOUTDIS | | TDCOUTDIS = 1 disables readout of valid data from the TDC |
| 2 | CALWAITENA | | CALWAITENA=1 enables wait for calibration values |
| 3 | AUTORSTDIS | | AUTORSTDIS = 1 disables automatic TDC-reset when SYSERR occurs |
| 4 | STOARTDIS | | STOARTDIS =1 disables generation of artifical stop signal |
| 5 | CNTDIS | | CNTDIS = 1 disables FPGA-counter |
| 6 | STADIS | | STADIS = 1 disables start signal going into the TDC |
| 7 | ADCCALDIS | | ADCCALDIS=1 disables using of ADC calibration |
| 15-8 | - | - | |

**Name:** MAINCTRL

**Description:** This register is used to as a common control register

**Address:** see register table

**Group:** common

**Access:** Read/Write

**Note:** · see below

| Bit-Position | Name | | Description |
|---|---|---|---|
| | | | |
| 1-0 | LVL[1:0] | | Level for storing as binning data, storing into binning array will be omitted if COIN[2:0] > LVL[1:0] |
| 2 | STOREDIS | | STOREDIS = 1 disables storing into SRAM |
| 3 | DEBUGENA | | DEBUGENA = 1 enables simulation of rawdata (debug) |
| 4 | DELSIMDIS | | DELSIMDIS = 1 disables simultanously deleting of both Binning and Raw Data array (after each read access on data) |
| 5 | LATCHUP | | LatchUp occured, if LATCHUP = 1 |
| 6 | ECENA | | Hamming-Code Error Correction on SRAM data enabled, if ECENA=1 |
| 15-7 | n.c. | | |

**Name:** STAT

**Description:** This register is used to reflect some TDC and FPGA status signals

**Address:** see register table

**Group:** common

**Access:** Read

**Note:** · see below

| Bit-Position | Name | | Description |
|---|---|---|---|
| | | | |
| 0 | TDCSYSERR | | SYSERR-Flag from TDC |
| 1 | TDCREADY | | READY-Flag from TDC |
| 2 | TDCVALID | | VALID-Flag from TDC |
| 3 | TDCCALM | | CALM-Flag from TDC |
| 4 | TDCBUSDIR | | BUSDIR-Flag from TDC |
| 8-5 | - | | |
| 9 | TDCWRERR | | TDC write access error – Flag |
| 10 | TDCRDERR | | TDC read access error – Flag |
| 11 | TDCCMDERR | | TDC command error – Flag |
| 12 | CMDBUSY | | Commanding TDC active, if CMDBUSY=1 |
| 13 | DACBUSY | | Programming DAC active, if DACBUSY=1 |
| 14 | NEWCAL | | new calibration data available, if NEWCAL=1; any read-access to the CALIBxx registers resets this flag |
| 15 | ADCERR | | ADC error occured, if ADCERR=1 |

**Name:** TDCCMD

**Description:** After being written with an TDC-opcode, the contents of this register will be written into the TDC

**Address:** see register table

**Group:** TDC handling

**Access:** Write

**Note:**
- dependent on the TDC-opcode, a respective parameter value has to be written into TDCWR before

**Name:** TDCRD

**Description:** After the transmission of an appropriate TDC-opcode, the contents of the requested TDC register is available in this register

**Address:** see register table

**Group:** TDC handling

**Access:** Read

**Note:** ·

**Name:** TDCWR

**Description:** Together with an appropriate TDC-opcode, the contents of this register will be used as a parameter value and will be written into the TDC

**Address:** see register table

**Group:** TDC handling

**Access:** Write

**Note:** ·

**Name:** CALIB11

**Description:** This register contains part1 of the calibration data from the TDC (CAL11)
**Address:** see register table
**Group:** Calibration
**Access:** Read
**Note:**
- New calibration data will be shown with STAT.NEWCAL=1

**Name:** CALIB12

**Description:** This register contains part2 of the calibration data from the TDC (CAL12)
**Address:** see register table
**Group:** Calibration
**Access:** Read
**Note:**
- New calibration data will be shown with STAT.NEWCAL=1

**Name:** CALIB21

**Description:** This register contains part3 of the calibration data from the TDC (CAL21)
**Address:** see register table
**Group:** Calibration
**Access:** Read
**Note:**
- New calibration data will be shown with STAT.NEWCAL=1

**Name:** CALIB22

**Description:** This register contains part4 of the calibration data from the TDC (CAL22)
**Address:** see register table
**Group:** Calibration
**Access:** Read
**Note:**
- New calibration data will be shown with STAT.NEWCAL=1

**Name:** CALNG

**Description:** With respect to the position of the comma, this register has to be written with the calculated gain G from the calibration process

**Address:** see register table

**Group:** Calibration

**Access:** Write

**Note:**
- 11 digits after decimal point

**Name:** CALNO

**Description:** With respect to the position of the comma, this register has to be written with the calculated offset O from the calibration process

**Address:** see register table

**Group:** Calibration

**Access:** Write

**Note:**
- 4 digits after decimal point

**Name:** COINBDRS

**Description:** This register contains both lower and upper time borders for the coincidence part

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**
- This register has to be written with a hardware-specific value

**Name:** DACCMD

**Description:** This register will be used to program the DAC on the preamplifier board (serial link)

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**
- programming of one DAC channel needs about 4µs

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | |
| - | | | | DACCHN[3:0] | | | | DACVAL[7:0] | | | | | | | |

DACVAL[7:0]: 8-bit DAC value
DACCHN[3:0]: DAC Channel to be programmed, 0x01 ≡ Channel A, 0x02 ≡ Channel B, 0x03 ≡ Channel C etc.

**Name** PREAMPCTRL

**Description:** This lower 8 bit of this register are directly connected to the preamplifier board and will be used to control this (tbd)

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**
- PREAMPCTRLVAL[7:0] = 0xff after power up

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | |
| - | | | | | | | | PREAMPCTRLVAL[7:0] | | | | | | | |

**Name:** ADCRST

**Description:** Write access to this register resets the ADC control unit and performs new calibration of the ADC ('internal mid-scale error calibration')

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**

**Name:** STARTCNT

**Description:** This register counts the start-signals from the NPD-sensor up to the upper limit of 65536

**Address:** see register table

**Group:** FPGA Counter

**Access:** Read

**Note:**
- Write access to CNTRST sets counter to zero
- FPGA counters should be disabled (FRONTCTRL.CNTDIS = 1) before read access

**Name:** STOP0CNT

**Description:** This register counts the stop0-signals from the NPD-sensor up to the upper limit of 65536

**Address:** see register table

**Group:** FPGA Counter

**Access:** Read

**Note:**
- Write access to CNTRST sets counter to zero
- FPGA counters should be disabled (FRONTCTRL.CNTDIS = 1) before read access

**Name:** STOP1CNT

**Description:** This register counts the stop1-signals from the NPD-sensor up to the upper limit of 65536

**Address:** see register table

**Group:** FPGA Counter

**Access:** Read

**Note:**
- Write access to CNTRST sets counter to zero
- FPGA counters should be disabled (FRONTCTRL.CNTDIS = 1) before read access

**Name:** STOP2CNT

**Description:** This register counts the stop2-signals from the NPD-sensor up to the upper limit of 65536

**Address:** see register table

**Group:** FPGA Counter

**Access:** Read

**Note:**
- Write access to CNTRST sets counter to zero
- FPGA counters should be disabled (FRONTCTRL.CNTDIS = 1) before read access

**Name:** TOFCNT

**Description:** This register counts the generated TOF values up to the upper limit of 65536

**Address:** see register table

**Group:** FPGA Counter

**Access:** Read

**Note:**
- Write access to CNTRST sets counter to zero
- FPGA counters should be disabled (FRONTCTRL.CNTDIS = 1) before read access
- For debugging/evaluating purpose only

**Name:** RAWCNT

**Description:** This register counts the generated RAW values up to the upper limit of 65536

**Address:** see register table

**Group:** FPGA Counter

**Access:** Read

**Note:**
- Write access to CNTRST sets counter to zero
- FPGA counters should be disabled (FRONTCTRL.CNTDIS = 1) before read access
- For debugging/evaluating purpose only

**Name:** CNTRST

**Description:** This register is used to reset the 6 FPGA counters

**Address:** see register table

**Group:** FPGA Counter

**Access:** Write

**Note:** Write access sets all 6 FPGA counters to zero

**Name:** BINARRAY

**Description:** This register will be used to link the binning array (from SRAM) to the DPU

**Address:** see register table

**Group:** SRAM Data

**Access:** Read/Write

**Note:**

- Readout of the binning data must be initiated by a write access to this register, after that 1024 read access have to be performed.
- This register automatically reflects the next valid data item from the binning array
- The respective memory cells will be set to zero automatically after read access on BINARRAY or RAWARRAY, latter only if MAINCTRL.DELSIMDIS = 0
- Time between two read accesses should not be less than 1.2µs (tpd)
- Storing into SRAM must be disabled (MAINCTRL.STOREDIS = 1) bevor any access to this register
- exceeding the amount of read accesses is shown by Data=0x5555

**Name:** RAWARRAY

**Description:** This register will be used to link the raw data array (from SRAM) to the DPU

**Address:** see register table

**Group:** SRAM Data

**Access:** Read/Write

**Note:**

- Readout of the binning data must be initiated by a write access to this register, after that 1024 read access have to be performed.
- This register automatically reflects the next valid data item from the binning array
- The respective memory cells will be set to zero automatically after read access on RAWARRAY or BINARRAY, latter only if MAINCTRL.DELSIMDIS = 0
- Time between two read accesses should not be less than 1.2µs (tpd)
- Storing into SRAM must be disabled (MAINCTRL.STOREDIS = 1) bevor any access to this register
- exceeding the amount of read accesses is shown by Data=0x5555

**Name:** STOPARRAY

**Description:** This register will be used to link the stop array (from SRAM) to the DPU

**Address:** see register table

**Group:** SRAM Data

**Access:** Read/Write

**Note:**
- Readout of the binning data must be initiated by a write access to this register, after that 48 read access have to be performed.
- This register automatically reflects the next valid data item from the binning array
- The respective memory cells will be set to zero automatically after read access on STOPARRAY
- Time between two read accesses should not be less than 1.2µs (tpd)
- Storing into SRAM must be disabled (MAINCTRL.STOREDIS = 1) bevor any access to this register
- exceeding the amount of read accesses is shown by Data=0x5555

**Name:** SEFCCNT

**Description:** This registers counts the occurance and correction of single bit errors of the SRAM up to the upper limit of 65536.

**Address:** see register table

**Group:** SRAM Data

**Access:** Read

**Note:**
- Register is cleared by hardware-reset to DigTOF

**Name:** DEFCCNT

**Description:** This registers counts the occurance of double bit errors of the SRAM data from up to the upper limit of 65536.

**Address:** see register table

**Group:** SRAM Data

**Access:** Read

**Note:**
- Register is cleared by hardware-reset to DigTOF

**Name:** TDCRST

**Description:** This register will be used to reset the TDC and the error-flags

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**
- For debugging/evaluating purpose only

**Name:** RAWDBGHIGH

**Description:** This register will be used to simulate RAW-data (these are the upper 16 bits)

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**
- to see any effect, MAINCTRL.DEBUGEN has to be set to 1 before
- pending write access to RAWDBGLOW is needed
- For debugging/evaluating purpose only

**Name:** RAWDBGLOW

**Description:** This register will be used to simulate RAW-data (these are the lower 16 bits)

**Address:** see register table

**Group:** Miscellaneous

**Access:** Write

**Note:**
- to see any effect, MAINCTRL.DEBUGEN has to be set to 1 before
- RAWDBGHIGH have to be written before
- For debugging/evaluating purpose only

**Name:** IFTEST

**Description:** This 16-bit register is implemented to simplifiy the DPU I/F-tests (single Write and Read accesses)

**Address:** see register table

**Group:** Miscellaneous

**Access:** Read/Write

**Note:**
- For debugging/evaluating purpose only

# Appendix

The following pages previously contained extracts from the EGSE-S/W @IDA. Now these extracts are available within single files.