28.20 Digital PLL Module (DPLL)

28.20.1 Overview

The digital PLL (DPLL) sub-module is used for frequency multiplication. The purpose of this module is to get a higher precision of position or value information also in the case of applications with rapidly changed input frequencies. There are two input signals *TRIGGER* and *STATE* for which periodic events are processed. The time period between two active events is called an increment. Each increment is divided into a given number of sub increments by pulses called SUB_INC. The resolution of the generated pulses is restricted by the period of the CMU_CLK0 clock or the TS_CLK respectively (see description of the modules TBU, CMU). The input signals *TRIGGER* and *STATE* can have the meaning of position information of linear or angle motions, mass flow values, temperature, pressure or level of liquids.By means of the DPLL the load of the CPU can be reduced essentially by relieving it from repeated or periodic standard tasks.

The DPLL has to perform the following tasks:

- prediction of the duration of the current increment in **Section 28.20.6**
- generation of SUB_INC1,2 pulses for up to 2 position counters in normal or emergency mode (see **Section 28.20.8.3**)
- synchronization of the actual position (under CPU control, see **Section 28.20.8.6.1**)
- possibility of seamless switch to emergency mode and back under CPU control, see configuration register DPLL_CTRL_0 at **Section 28.20.12**
- prediction of position and time related events in **Section 28.20.7**

28.20.2 Requirements and demarcation

The two input signals *TRIGGER* and *STATE* can be sensor signals from the same device or from two independent devices. When they come from the same device the *TRIGGER* input is typically a more frequent signal and *STATE* is a less frequent signal. In such a case the *STATE* signal can support an emergency mode, when no *TRIGGER* signal is available. There are also applications supported when *STATE* and *TRIGGER* are independent signals from different devices. Both input signals are combined with a validation signal *T_VALID* or *S_VALID* respectively, which shows the appearance of new data and must result in a data fetch and a start of the correspondent state machine to perform the calculations (see explanation below).

When *STATE* is a redundant signal of the same device only the *TRIGGER* input is used to generate the SUB_INC1 pulses in normal mode. There is a configuration possible, called emergency mode, for which the SUB_INC1 pulses are generated using the *STATE* input signal.

The decision to switch in the emergency mode and back is made outside the DPLL. The CPU must switch the configuration bit RMO (reference mode) in the DPLL_CTRL_0 register (see **Section 28.20.12**). Because a switch in emergency mode can appear suddenly, the information of the last increment duration of the *STATE* input up to FULL_SCALE should be stored always as a precaution.

The filtering as well as the combination or choice of the input signals is made in the TIM sub-module (see chapter “Timer Input Module (TIM)”) by use of a configurable filter algorithm for each slope and signal as well as in the MAP module (see chapter “TIM0 Input Mapping Module (MAP)”) the right *TRIGGER* or *STATE* signal is selected by a multiplexer or in the SPE module (see chapter “Sensor Pattern Evaluation (SPE)”) different signals are combined to a *TRIGGER* or *STATE* signal by using an antivalence operation.

The filter delay value of the signal is transmitted from the TIM module in the *FT* part of the corresponding signal, because the delay conditions of the signals can change during application.

The filter delays depend also on the filter algorithms used. Only the effective filter delay can be considered in the DPLL.

In order to provide the timing conditions to the DPLL the input trigger signals should have a time stamp (and optional in addition a filter value and a signal level value, as stated above) with an appropriate resolution. The resolution of the time stamps can be either the same resolution as the input time base TBU_TS0 (see **Figure 123**) or 8 times higher, selected by configuration bits in the DPLL_CTRL_1 register (see **Register DPLL_CTRL_1**). The time base TBU_TS0 is used to predict events in the future, called actions.

At the SUB_INCx outputs a predefined number of pulses between each active slope of the *TRIGGER/STATE* signal is generated, when the correspondent pulse generator is enabled by the enable bits SGEx=1 in the DPLL_CTRL_1 register (see **Register DPLL_CTRL_1**).

Dependent on configuration different strategies can be used to correct a wrong pulse number.

The FULL_SCALE range is divided into a fix number of nominal increments. Nominal increments do have the same size. The number of nominal increments in HALF_SCALE is specified in the DPLL_CTRL_0 register (see **Register DPLL_CTRL_0**).For synchronization purposes some *TRIGGER/STATE* input signals can be suppressed in dependency on the current position. Therefore an increment as duration between two active input events can be either a nominal increment or it can consist of more than one nominal increment. While a true nominal increment starts with an active event a virtual increment (of always nominal size) is an increment which starts with a missing event. Each increment which represents a gap (e.g. for synchronization purposes) consists of exactly one true nominal increment and at least one virtual increment, each of them having the same nominal duration (see figure below).

28.20.3 Input signal courses

Typical input signal courses are shown in the figure below.

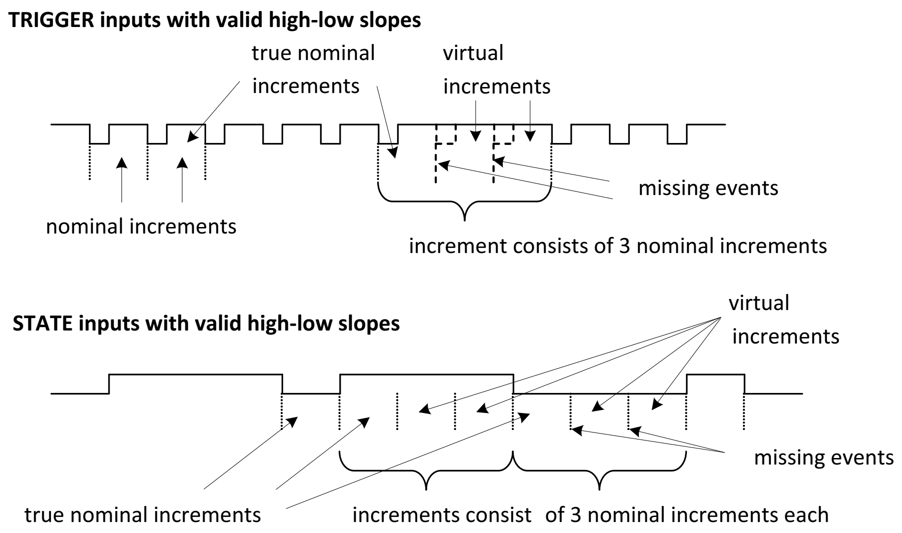

TRIGGER inputs with valid high-low slopes
true nominal increments
virtual increments
missing events
nominal increments
increment consists of 3 nominal increments
STATE inputs with valid high-low slopes
virtual increments
missing events
true nominal increments
increments consist of 3 nominal increments each

Figure 122 Trigger and State Input Signal

28.20.4 Block and interface description

The block description of the DPLL is shown in the following figure.

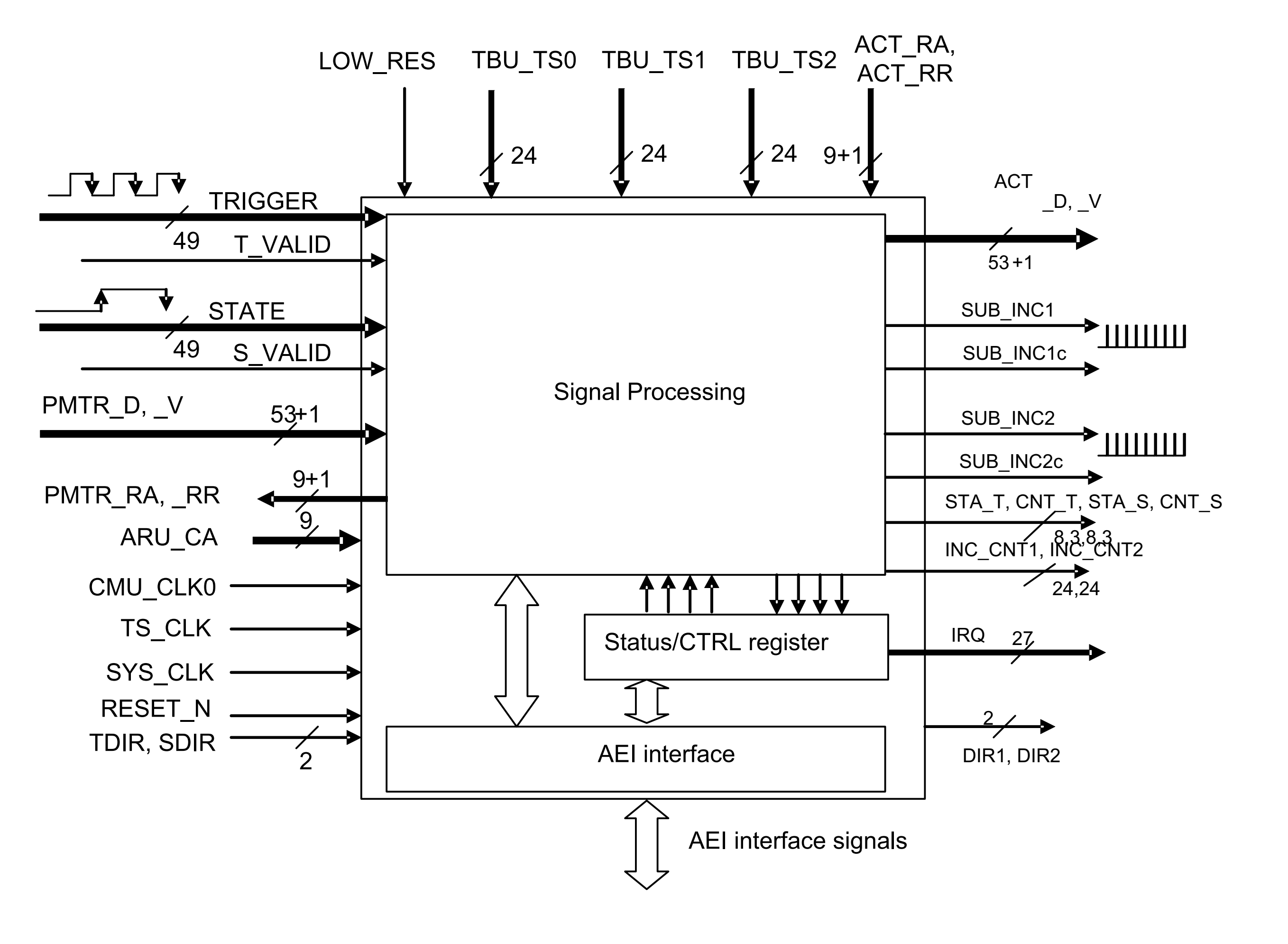


Figure 123 DPLL Block Diagram

Section 28.20.4.1 summarizes the interface signals of the DPLL shown by the block diagram above.

28.20.4.1 Interface description of DPLL

Table 70 Interface description of DPLL

Name	Width	I/O	Description	Comment
TRIGGER	49	I	Normal Signal for triggering DPLL by positions/values Bit(48)= TRIGGER_S Bits(47:24)= TRIGGER_FT Bits(23:0)= TRIGGER_TS	One bit signal value (SV), 24 bits filter delay value info and 24 bits time stamp, filtered in different modes.
STATE	49	I	Assistance signal for synchronization STATE(48)= STATE_S STATE(47:24)= STATE_FT STATE(23:0)= STATE_TS	Replacement of signal *TRIGGER* for emergency situations, or signal from an independent device; bits like above, corresponding
T_VALID	1	I	The values of *TRIGGER* are valid	Announces the arrival of a new *TRIGGER* value
S_VALID	1	I	The values of STATE are valid	Announces the arrival of a new STATE value
PMTR_D	53	I	Position minus time request data, delivered by ARU on request for up to 24 requests PMTR_RR; SV_i=PMTR_D(52:48): ACB bits, directly written to the correspondent DPLL_ACB_j registers PSA[i]=PMTR_D(47:24): position value for action DLA[i]=PMTR_D(23:0) time delay value for action	Data values for calculation of actual Actions; the values are requested by AENi=1[1] and CAIP=0[2]; a served request is shown by PMTR_V which signals that valid PMTR data arrived and they are written immediately after that to the corresponding RAM regions and registers; The DLA[i] values must have the same resolution as the TBU_TS0 input.
PMTR_V	1	I	signals a valid PMTR_D value, that means data is delivered on request	when valid: PMTR_D overwrites data in the PSA[i] and DLA[i] registers, also when the corresponding ACT_N[i][3] bit =1;
ARU_CA	9	I	Channel address; for valid PMTR addresses: demand data by setting PMTR_RR=1 when enabled by AENi=1[1] and CAIP=0[2];	counter value of ARU selects PMTR_RA and PMTR_RR when a valid address
PMTR_RA	9	O	read address of PMTR access	reflects ID_PMTR_i according to the selected channel address
PMTR_RR	1	O	read request of PMTR access; suppressed for CAIPi=1 (see DPLL_STATUS register)	reflects the value of the corresponding AENi[1] bit while the correspondent bit CAIPi=0[2]

Table 70 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
ACT_D	53	O	Output of a time stamp, a position and a control signal for a calculated action; SV_i=ACT_D(52:48): ACB bits, directly written from the correspondent PMTR_D signals; ACT_D(47:24) is the calculated position value PSAC[i] for the action in relation to TBU_TS1 or 2[4] and ACT_D(23:0) is the time stamp value TSAC[i] for the action in relation to TBU_TS0 [4]	Future time stamp with the resolution as TBU_TS0 input, additional position information and additional control bits;
ACT_V	1	O	ACT_D value is available and valid; blocking read access	for a valid action address: ACT_V reflects the shadow value of ACT_N[i][3] (ACT_N[i] is 1 when new PMTR values are available and the shadow register is updated, when a calculation of the actual PMTR values was done); reset after reading of the ACT_D values
ACT_RA	9	I	ACTION read address;	address bits for selection of all 24 action channels
ACT_RR	1	I	read request of selected action	the action data is demanded from another module
IRQ	27	O	Interrupt request output	Interrupts of DPLL;
SUB_INC1	1	O	Pulse output for *TRIGGER* input filter	sub-position increment provided continuously
SUB_INC2	1	O	Pulse output for *STATE* input filter (when *TRIGGER* and *STATE* are used for 2 independent devices)	sub-position increment provided continuously
SUB_INC1c	1	O	Pulse output for time base unit 1 in compensation mode (can stop in automatic end mode)	sub-position increment related to *TRIGGER* input
SUB_INC2c	1	O	Pulse output for time base unit 2 in compensation mode (can stop in automatic end mode)	sub-position increment related to *STATE* input (when *TRIGGER* and *STATE* are used for 2 independent devices)
TS_CLK	1	I	Time stamp clock	used for generation of the time stamps; this clock is used to generate the SUB_INC1,2 pulses
CMU_CLK0	1	I	CMU clock 0	used for rapid pulse correction of SUB_INC1,2
SYS_CLK	1	I	System clock	High frequency clock

Table 70 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
RESET_N	1	I	Asynchronous reset signal	Low active; After Reset he DPLL is available only after performing the RAM reset procedures by the DPLL hardware.

LOW_RES	1	I	low resolution of TBU_TS0 selected; shows which of the 27 bits of TBU_TS0 are connected to the DPLL	LOW_RES=0: T BU_TS0(DPLL)= lower 24 Bits of TBU_TS0(TBU);
				LOW_RES=1:
				TBU_TS0(DPLL)= higher 24 Bits of TBU_TS0(TBU);
				In the case LOW_RES=1 the TS0_HRT and/or TS0_HRS bits can be set [5)]

TBU_TS0	24	I	Actual time stamp from TBU; is needed to decide, if a calculated action is already in the past	24 bit time input, with a resolution of the time stamp clock
TBU_TS1	24	I	Actual position/value stamp 1; for calculation of position stamps (*TRIGGER*/*STATE*)	24 bit pos./val. input, with a resolution of the SUB_INC1 pulses
TBU_TS2	24	I	Actual position/value stamp 2; to be implemented for an additional independent position	ditto for SUB_INC2 for calculation of position stamps (*STATE*) for SMC[6)]=RMO[5)]=1
TDIR	1	I	Direction of *TRIGGER* input values (TDIR=0 does mean a forward direction and TDIR=1 a backward direction)	direction information from multiple sensors valid only for SMC[6)]=1 or IDDS=1
SDIR	1	I	Direction of *STATE* input values (SDIR=0 does mean a forward direction and SDIR=1 a backward direction)	direction information from multiple sensors valid only for SMC[6)]=1
DIR1	1	O	Direction information of SUB_INC1 (count forwards for DIR1=0 and backwards for DIR1=1)	count direction of TBU_CH1_BASE; DIR1 changes always after the evaluation of the corresponding valid *TRIGGER* slope and after incrementing/decrementing of the address pointer
DIR2	1	O	Direction information of SUB_INC2 (count forwards for DIR2=0 and backwards for DIR2=1)	count direction of TBU_CH2_BASE; DIR2 changes always after the evaluation of the corresponding valid *STATE* slope and after incrementing/decrementing of the address pointer
STA_T	8	O	Status of TRIGGER state machine	Output to MCS0. Signals accessible via µC interface as well (DPLL_STA)

Table 70 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
CNT_T	3	O	Count TRIGGER	Output to MCS0.This reflects the count of active *TRIGGER* slopes (mod8).Signals accessible via µC interface as well (DPLL_STA)
STA_S	8	O	Status of STATE state machine	Output to MCS0.Signals accessible via µC interface as well (DPLL_STA)
CNT_S	3	O	Count STATE	Output to MCS0.This reflects the count of active *STATE* slopes (mod8).Signals accessible via µC interface as well (DPLL_STA)
INC_CNT1	24	O	Increment counter of pulse generator 1 (automatic end mode)	Output to MCS0.Signals accessible via µC interface as well (DPLL_INC_CNT1)
INC_CNT2	24	O	Increment counter of pulse generator 2 (automatic end mode)	Output to MCS0.Signals accessible via µC interface as well (DPLL_INC_CNT2)

[1] see DPLL_CTRL_x register, x=2,3,4; see **Section 28.20.12.3**, **Section 28.20.12.4**, **Section 28.20.12.5**

[2] see DPLL_STATUS register; see **Section 28.20.12.30**

[3] see DPLL_ACT_STA register; see **Section 28.20.12.7**

[4] see DPLL input signal description; see **Section 28.20.1**

[5] see DPLL_CTRL_0 register; see **Section 28.20.12.1**

[6] see DPLL_CTRL_1 register; see **Section 28.20.12.2**

For references above the following hints are used: [1] see DPLL_CTRL_x register, x=2,3,4; see **Register DPLL_CTRL_2**,**Register DPLL_CTRL_3**,**Register DPLL_CTRL_4**[2] see DPLL_STATUS register; see **Register DPLL_STATUS**[3] see DPLL_ACT_STA register; see **Register DPLL_ACT_STA**[4] see DPLL_CTRL_0 register; see **Register DPLL_CTRL_0**[5] see DPLL_CTRL_1 register; see **Register DPLL_CTRL_1**[6] see DPLL input signal description; see **Section 28.20.1**

28.20.5 DPLL Architecture

28.20.5.1 Purpose of the module

The DPLL generates a predefined number of incremental signal pulses within the period between two events of an input *TRIGGER* or *STATE* signal, when the corresponding pulse generator is enabled. The resolution of the pulses is restricted by the frequency of the time stamp clock (TS_CLK). Changes in the period length of the predicted time period of the current increment will result in a change of the pulse frequency in order to get the same number of pulses. This adoption can be performed by DPLL hardware, software or with support of DPLL hardware in different modes.

The basic part of a DPLL is to make a prediction of the current period between two *TRIGGER* and/or *STATE* signal edges. Disturbances and systematic failures must be considered as well as changes of increment duration caused by acceleration and deceleration of the supervised process. Therefore, a good estimation is to be done using some measuring values from the past. When the process to be predicted takes a steady and differentiable course not only the current increment but also some more increments for the future can be predicted. In utilization of such calculations actions for the future can be predicted.

28.20.5.2 Explanation of the prediction methodology

As already shown in **Section 28.20.1** the DPLL has to perform different tasks. The basic function for all these tasks is the prediction of the current increment which is based on a relation between increments in the past. Because the relation between two succeeding intervals at a fixed position remains also valid in the case of acceleration or deceleration the prediction of the duration of the current time interval is done by a similarity transformation. Having a good estimation of the current time interval, all the other tasks can be done easily by calculations explained in **Section 28.20.6**.

28.20.5.3 Clock topology

All registers are read using the system clock *SYS_CLK*. The SUB_INC1,2 pulses generated have in the normal case the highest frequency not higher then *CMU_CLK0* or the half of *TS_CLK* respectively. For individual pulses the frequency can be doubled. All operations can be performed using the system clock.

28.20.5.4 Clock generation

The clock is generated outside the DPLL.

28.20.5.5 Typical frequencies

For the system clock a reasonable clock frequency should be applied to give the DPLL module sufficient computational power to calculate all needed values (prediction of next increment, actions) in time. The typical system clock frequency is in the range from 40 MHz up to 150 MHz.

28.20.5.6 Time stamps and systematic corrections

The time stamps for the input signals *TRIGGER* and *STATE* have 24 bits each. These bits represent the value of the 24 bit free running counter running with a clock frequency selected by the configuration of the TBU. Using a typical frequency of 20 MHz the time stamp represents a relative value of time with a resolution of 50 ns.

The input signals have to be filtered. The filter is not part of the DPLL. The time stamps can have a delay caused by the filter algorithm used. There are delayed and undelayed filter algorithms available and the delay value can depend on a time or a position value.

Systematic deviations of *TRIGGER* inputs can be corrected by a profile, which also considers systematic missing *TRIGGER*s. The increments containing missing *TRIGGER*S are divided into the corresponding number of nominal increments whereas the duration of a nominal increment is the greatest divider of all increments duration.

For each increment this number of enclosed nominal increments is stored in a profile as NT value for *TRIGGER*. When the increment is a nominal increment the NT value is 1.

For the *TRIGGER* input the value NT is stored in the ADT_T field in RAM region 2c.

In the case of **AMT**[4] = 1 the **ADT_T[i]** values in the RAM region 2c must also contain the adapting information for the *TRIGGER* signal, which considers for each increment a systematic physical deviation **PD** from the perfect increment value with a resolution according to the chosen value of MLT+1, which describes the number of SUB_INC1 pulses for a nominal increment.

The value **PD** for the *TRIGGER* describes the amount of missing or surplus pulses with a sint13 value, to be added to MLT+1 directly. The correction value is in this way also applicable in the case of missing *TRIGGER* inputs for the synchronization gaps. In this case the amount of provided SUB_INC1 pulses for a nominal increment (MLT+1) is multiplied (MLT+1) + PD is multiplied by NT.

The NT value of the current increment is stored in the variable SYN_T (see **NUTC** register in **Section 28.20.12.14**).

In the case of **RMO**[4]= 1 for **SMC**[5]=0 (emergency mode) the time stamp of *STATE* is used to generate the output signal SUB_INC1.

More inaccuracy should be accepted in emergency mode because usually there are only fewer events available for FULL_SCALE according to the value SNU[5].

For the *STATE* signal the systematic deviations of the increments can be corrected in the same way as for *TRIGGER* by profile and adaptation information as described below.

Systematic deviations of *STATE* inputs can be corrected by a profile, which also considers systematic missing *STATE* events. The increments containing missing *STATEs* are divided into the corresponding number of nominal increments whereas the duration of a nominal increment is the greatest divider of all increments duration.

For each increment this number of enclosed nominal increments is stored in a profile as NS value for *STATE*. When the increment is a nominal increment the NS value is 1.

For the *STATE* input the value NS is stored in the ADT_S field in RAM region 1c3.

In the case of **AMS**[4] = 1 the **ADT_S[i]** values in the RAM region 1c3 must contain the adapting information for the STATE signal, which considers for each increment a systematic physical deviation **PD_S** from the perfect increment value with a resolution according to the chosen value of MLS1, which describes the number of SUB_INC1 pulses for a nominal increment (see below).

The number of pulses SUB_INC1 for a nominal *STATE* increment in emergency mode (for SMC=0) is given by the value of MLS1= (MLT + 1)* (TNU + 1) /(SNU + 1) in order to get the same number of pulses in FULL_SCALE for normal and emergency mode. This value has to be configured by the CPU.

The value **PD_S** for the *STATE* describes the amount of missing or surplus pulses with a sint16 value, to be added to MLS1 directly. The correction value is in this way also applicable in the case of missing *STATE* inputs for the synchronization gaps. In this case the amount of provided SUB_INC1 pulses for a nominal increment MLS1 is multiplied by NS first before the PD_S value is added.

The current NS value is stored in the variable SYN_S (see **NUSC** register in **Section 28.20.12.15**).

28.20.5.7 DPLL Architecture overview

As shown in **Figure 123** the DPLL can process different input signals. The signal *TRIGGER* is the normal input signal which gives the detailed information of the supervised process. It can be for instance the information of water or other liquid level representing the volume of the liquid, where each millimeter increasing results in a *TRIGGER* signal generation. In order to get a predefined filling level, without overflow also the inertia of the system must be taken into account. Hence, some delay for closing the inlet valve and also the remaining water amount in the pipe must be considered in order to start the closing action earlier as the filling level will be reached.

A second input signal *STATE* sends an additional (redundant) information for instance at some centimeters and because of intervals with different distances it gives also information about the system state with the direction of the water flow (in or out), while the *TRIGGER* signal must not contain information concerning the flow direction. In some applications the inactive slope of *TRIGGER* can be utilized to transmit a direction information. In the case of faults in the *TRIGGER* signal the *STATE* signal is to be processed in order to reach the desired value nevertheless, maybe with some loss of accuracy.

The measuring scale can have some systematic failures, because not all millimeter or centimeter distances measured mean the same value. This could be due to changes in the thickness of the measuring cylinder or the inaccurate position of the marks. These systematic failures are well known by the system and for improvement of the prediction the signals *ADT_T* and *ADT_S* for the correction of the systematic failures of *TRIGGER* and *STATE* respectively are stored in the internal RAM.

The input signals *TRIGGER* and *STATE* are represented as a time stamp signal each, which is stored in the 24 bit TS-part of the corresponding signal.

Information concerning the delay of this signal by filtering of disturbances is stored in the 24 bit FT-part of the signal.

In order to establish the relation of time stamps to the actual time the TBU_TS0[6] value is also provided showing the actual time value used for prediction of actions in the future.

After reaching the desired water level the water is filled in a bottle by draining. After that the water filling is repeated. The water level at draining is observed by the same sensor signals (the same number of *TRIGGER* pulses), but the duration of the draining could be different from the filling time. Both times together form the FULL_SCALE region, while one of them is a HALF_SCALE region, which can differ in time but not in the number of pulses, especially for *TRIGGER*.

For synchronization purposes some *TRIGGER* marks can be omitted in order to set the system to a proper synchronization value (maybe before the upper filling value is reached).

In emergency situations, when the *TRIGGER* signals are missed the *STATE* signal is used instead of.

The PMTR_i [4] signals announce the request for a position minus time calculation for up to 24 events.

All 24 events can be activated using the 24 AENi[1] (action enable) bits. Each of these enable bits are asked by the routing engine for a read access. The corresponding read request is generated by the AENi bit while CAIPx is zero. CAIP1 and CAIP2 are two bits of the DPLL_STATUS register for 12 actions each with the meaning “calculation of actions in progress”, controlled by the state machine (see **Section 28.20.2**) for scheduling the operations.

When such a request is serviced by the ARU (in the case CAIPx=0) the values for position and time are written in the corresponding RAM 1a region (0x0200… 0x025C for the position value and 0x0260… 0x02BC for the delay value), the control bits for the corresponding action are set accordingly. When a new PMTR value arrives, an old value is overwritten without notice and the shadow bit of ACT_N[i] is cleared while the ACT_N[i] (new action) bit in the DPLL_ACT_STA register is set. The ACT_N[i] is cleared, when the currently calculated action value is in the past. Overwriting of old information is possible without data inconsistency because the read request to ARU is suppressed during action calculations by the CAIP1,2 bits. In this way always the last possible PMTR value is used consistently.

28.20.5.8 DPLL Architecture description

The DPLL block diagram of **Figure 123** will now be explained in detail in combination with some example configurations of the control registers. There are different configuration bits available which can adopt the DPLL to the use case (see **Section 28.20.12**).

Let for example in HALF_SCALE the *TRIGGER* number TNU[4] be 0x3B (which is for TNU+1 = 60 decimal that does mean 120 events in FULL_SCALE) and the number of SUB_INC1 pulses between two *TRIGGER*s MLT[4] be 0x257 (this means 600 pulses per *TRIGGER* event). Than the FULL_SCALE region can be divided into 72000 parts each of them associated with its own SUB_INC1 pulse. For a run through FULL_SCALE all 72000 pulses should appear but maybe with a different pulse frequency between two *TRIGGER* events. For this example after each 600 pulses at the *SUB_INC1* output the next *TRIGGER* event is to be expected with the corresponding new time stamp.

Missing SUB_INC1 pulses due to acceleration have to be taken into account within the next increment. Not one pulse has to be missed or added because of calculation inaccuracy in average for a sufficient number of FULL_SCALE periods. This means that not one pulse is sent in addition and all missing pulses are to be caught up on afterwards.

For the systematic arrangement of *TRIGGER* inputs **the profile** (as already mentioned in **Section 28.20.5.6** is stored in the RAM region 2c (see **Section 28.20.14.3**). In this field the relative position of gaps can be stored in the NT value and also physical deviations in the PD value.

For the consideration of systematic missing *TRIGGER*s the actual NT value of the profile is stored in the SYN_T bits of the NUTC register (see **Section 28.20.12.14**).

In normal mode the physical deviation values PD in the ADT_T field could be used to balance the local systematic inaccuracy of the *TRIGGER* signal. The value of PD (see **Section 28.20.14.3**) is the pulse difference in the corresponding increment and does mean the number of sub pulses to be added to the nominal number of pulses. PD is a signed integer value using 13 bits: up to +/-4096 pulses can be added for each increment.

The NT value of the profile ADT_T has the value 1, when a nominal increment is assumed. An integer number greater than 1 shows the number of nominal increments to be considered for a gap. For the actual increment after synchronization the corresponding NT value is stored in SYN_T of the NUTC register.

Using the *STATE* input there are similar configuration bits available (see **Section 28.20.12**).

Let for example in HALF_SCALE the *STATE* number SNU[5] be 0xB (which is for SNU+1 =12 decimal and while SYSF[5] = 0 that does mean 24 events in FULL_SCALE). In order to get the same number of SUB_INC1 pulses for FULL_SCALE as above for *TRIGGER*s the value (MLT+1)=600 is divided by 2*(SNU+1)=24 and multiplied with 2*(TNU+1)=120. The result 3000 must be stored in MLS1 by the CPU (see **Section 28.20.12.75**).

For the systematic arrangement of *STATE* inputs **the profile** (as already mentioned in **Section 28.20.5.6** is stored in the RAM region 1c3 (see **Section 28.20.12.90**). In this field the relative position of gaps can be stored in the NS value and also physical deviations in the PD_S value.

For the consideration of systematic missing *TRIGGER*s the actual NS value of the profile is stored in the SYN_S bits of the NUSC register (see **Register DPLL_NUSC**).

In emergency mode the physical deviation values PD_S in the ADT_S field could be used to balance the local systematic inaccuracy of the *STATE* signal. The value of PD_S (see **Section 28.20.12.90**) is the pulse difference in the corresponding increment and does mean the number of sub pulses to be added to the nominal number of pulses per increment. PD_S is a signed integer value using 16 bits: up to +/-32768 pulses can be added for each increment.

In emergency mode the physical deviation values PD_S in the ADT_S field could be used to balance the local systematic inaccuracy of the STATE signal. The value of PD_S (see **Section 28.20.12.90**) is the pulse difference in the corresponding nominal increment and does mean the number of sub pulses to be added to the nominal number of pulses. . PD_S is a signed integer value using 16 bits: up to +/-32768 pulses can be added for each increment.

The NS value of the profile ADT_S has the value 1, when a nominal increment is assumed. An integer number greater than 1 shows the number of nominal increments to be considered for a gap. For the actual increment after synchronization the corresponding NS value is stored in SYN_S of the NUSC register.

28.20.5.9 Block diagrams of time stamp processing.

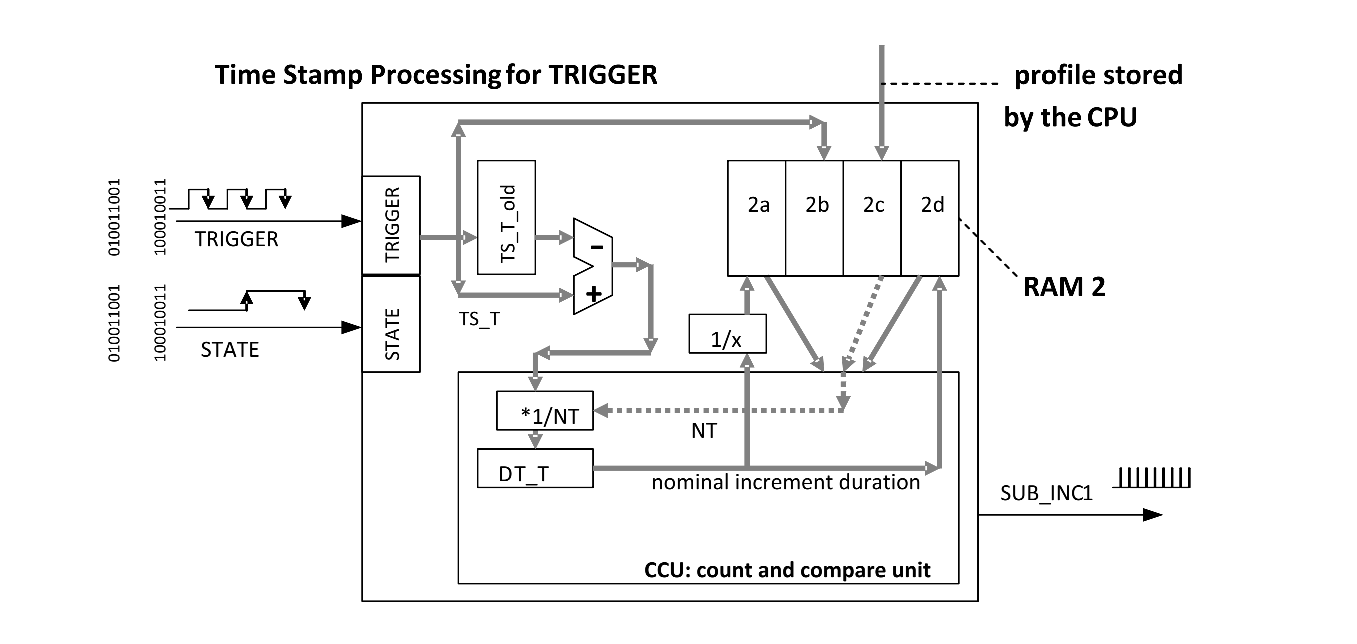

Time Stamp Processing for TRIGGER
profile stored by the CPU
010011001
100010011
TRIGGER
010011001
100010011
STATE
TRIGGER
STATE
TS_T_old
TS_T
-
+
2a
2b
2c
2d
RAM 2
1/x
*1/NT
NT
DT_T
nominal increment duration
CCU: count and compare unit
SUB_INC1

Figure 124 Time Stamp Processing Trigger

As shown in the block diagram above the time stamp difference of two succeeding input events is calculated. For the prediction of the current increment duration such values from the past are used. For this purpose the measured and calculated values of the last FULL_SCALE period are stored in the RAM. For the *TRIGGER* input there are 4 different RAM parts in the RAM region 2:

2a stores the reciprocals of each nominal increment duration RDT_T

2b stores the time stamps of each valid input event TSF_T

2c is used for the profile ADT_T and

2d for the nominal increment durations DT_T.

Because the prediction is based on the relations of increments in the past this relation can be calculated easily by the multiplication of increment duration values with the reciprocal value of another increment. In order not to be forced to distinguish between gaps and "normal" increments duration also for gaps only the nominal duration and the correspondent reciprocal values are stored in the RAM field. This is possible by consideration of the NT value in the profile: the measured increment duration is divided by NT.

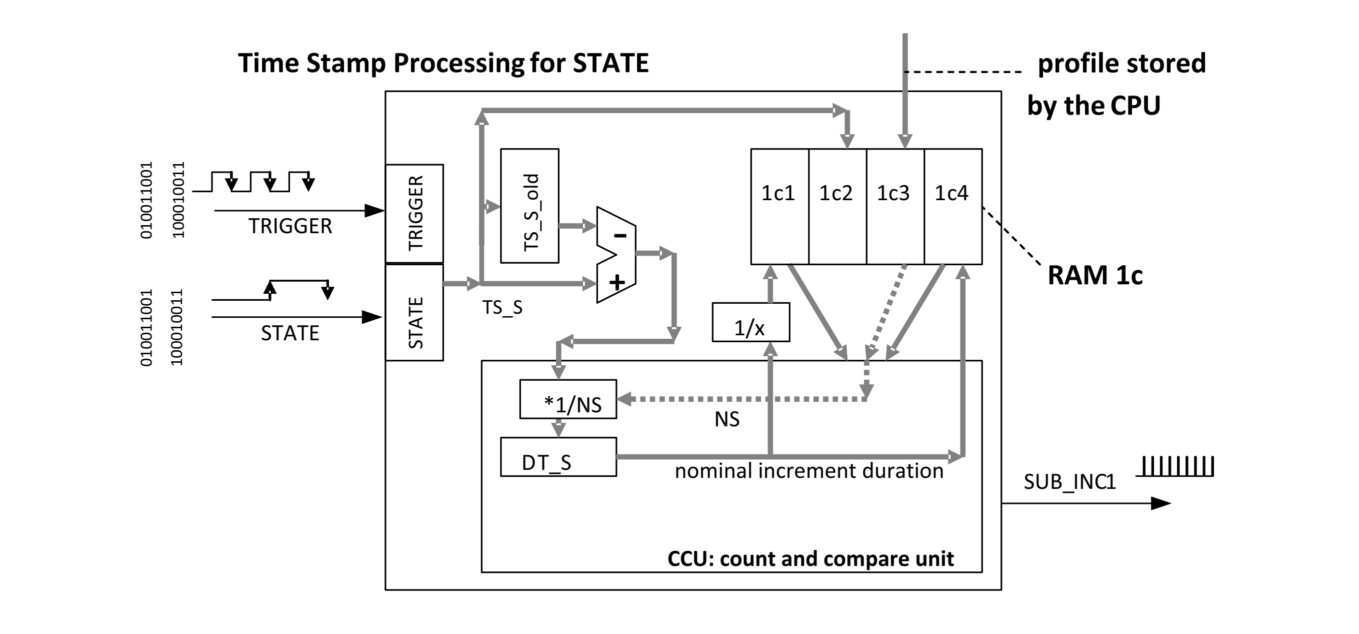


Figure 125 Time Stamp Processing State

For the *STATE* input there are also 4 different RAM parts in the RAM region 1c:

1c1 stores the reciprocals of each nominal increment duration RDT_S

1c2 stores the time stamps of each valid input event TSF_S

1c3 is used for the profile ADT_S and

1c4 for the nominal increment durations DT_S.

The calculations are performed similar as for the *TRIGGER* input. The NS value in the profile shows the appearance of a gap.

28.20.5.10Register and RAM address overview

The address map of the DPLL is divided into register and memory regions as defined in **Table 71**. The addresses from 0x0000 to 0x00FC are reserved for registers, from 0x0100 to 0x01FC is reserved for action registers to serve the ARU at immediately read request.

The RAM is divided into 3 independent accessible parts 1a, 1b+c and 2.

The part 1a from 0x0200 to 0x037C is used for PMTR values got from ARU and intermediate calculation values; there is no write access from the CPU possible, while the DPLL is enabled.

The RAM 1b part from 0x0400 to 0x05FC is reserved for RAM variables and the RAM part 1c from 0x0600 to 0x09FC is used for the *STATE* signal values.

The RAM region 2 from 0x4000 to 0x7FFC is reserved for the *TRIGGER* signal values. RAM region 1a has a size of 288 bytes, Ram 1b+c uses 1,125 Kbytes while RAM region 2 is configurable from 1,5 to 12 Kbytes, depending on the number of *TRIGGER* events in FULL_SCALE. The AOSV_2 register is used to determine the beginning of each part.

Table 71 gives the DPLL Address map overview.

Registers are used to control the DPLL and to show its status. Also parameters are stored in registers when useful. The table below shows the addresses for status and control registers as well as values stored in additional registers. The register meaning explained in the register overview (**Section 28.20.11**) while the bit positions of the status and control registers are described in detail in **Section 28.20.12**.

Time stamps for TRIGGER and *STATE* can have either the same resolution as the TBU_TS0 input or 8 times higher. This is configured in the DPLL_CTRL_1 register (see **Register DPLL_CTRL_1**). While the TBU_TS0 is used for action predictions the higher resolution of *TRIGGER* and *STATE* inputs can be used for a more accurate pulse generation.

The time stamp fields of *TRIGGER* and *STATE* are stored in the corresponding RAM regions in such a way, that for a gap also entries for the virtual increments are provided. This is due to the necessity to calculate time differences between a given number of (real and virtual) input events independent of a gap. Therefore the gap is extended in the RAM fields 2b and 1c2.For all other RAM regions in RAM 2 and RAM 1c the gap is considered as one increment.

For the access to the RAM fields there must be address pointers. When the device starts all address pointers have a zero value and the first measured and calculated values are stored in the beginning of the corresponding RAM field.Because the position of the device is usually unknown at the beginning no profile information can be used. The profile regions must have their own address pointers each which are set by the CPU as soon as the position is known. By setting the appropriate value to the address pointer APT_2C of the *TRIGGER* profile or APS_1C3 of the *STATE* profile respectively the synchronization bits in the DPLL_STATUS register SYT or SYS are set respectively. In the following the gap information can be used.

Because the time stamp fields are extended at the gaps there must be additional address pointers for these regions: APT_2B for *TRIGGER* time stamps and APS_1C2 for *STATE* time stamps. These address pointers must be incremented by NT or NS respectively when a gap appears.

Table 71 Register and RAM address map

Addr. rangeStart	Addr. rangeEnd	Value number	Byte #	Content	Indication	Region	RAM size
0x0000	0x0FC	64	256	Register	used/reserved	0	no RAM
0x100	0x1FC	64	192	ACTION registers	direct read from ARU	0	no RAM
0x0200	0x03FC	128	384	PMTRvalues RAM 1a	CPU R/Pw access, when DPLL disabled; ARU has highest priority	1a with own ports	RAM part 1a:384 bytes
0x0400	0x05FC	128	384	Variables RAM 1b	R and monitored W access by the CPU	1b	RAM part 1b+c:1,125 Kbytes
0x0600	0x09FC	256	768	*STATE* data	R and monitored W access by the CPU	**1c**	
0x0600	0x06FC	64	192	RDT_S[i]	*STATE* reciprocal values	1c1	
0x0700	0x07FC	64	192	TSF_S[i]	*STATE* TS values	1c2	
0x0800	0x08FC	64	192	ADT_S[i]	adapted values of *STATE*	1c3	

Table 71 Register and RAM address map (cont'd)

Addr. rangeStart	Addr. rangeEnd	Value number	Byte #	Content	Indication	Region	RAM size
0x0900	0x09FC	64	192	DT_S[i]	nom. STATE inc.	1c4	
0x4000	0x47FC…0x7FFC	512 …4096	1536 …12288	*TRIGGER* data	R and monitored W access of CPU	**2**	RAM part 2: 1,5…12 Kbytes
0x4000	0x41FC…4FFC	128… 1024	384…3072	RDT_T[i]	*TRIGGER* reciprocal values	2a	
0x4200…5000	0x43FC…5FFC	128…1024	384…3072	TSF_T[i]	*TRIGGER* TS values	2b	
0x4400…6000	0x45FC…6FFC	128… 1024	384…3072	ADT_T[i]	adapted values of *TRIGGER*	2c	
0x4600…7000	0x47FC…7FFC	128… 1024	384…3072	DT_T[i]	nom. *TRIGGER* increments	2d	

28.20.5.10.1 RAM Region 1

RAM region 1 has a size of 1,5 Kbytes and is used to store variables and parameters as well as the measured and calculated values for increments of *STATE*. The RAM 1 region is divided into two independent accessible RAM parts (a and b+c) with own ports. The address information is shown in the table above and the detailed description is performed in the following chapters. The RAM 1a is used to store the PMTR values got from ARU and in addition some intermediate calculation results of actions. RAM region 1b is used for variables needed for the prediction of increments, while RAM 1c is used to store time stamps, profile and duration of all the *STATE* inputs of the last FULL_SCALE region. All variables and values of RAM 1b+c part use a data width of up to 24 bits.

The RAM is to be initialized by the DPLL after HW-reset. All RAM cells must have a zero value after performing the initialize procedure. This is performed when setting The Init_RAM bit in the DPLL_RAM_INI register. The DPLL is only available after finishing this procedure. The initialization progress is shown in the status bits of the same register.

- **RAM Region 1a**: used for storage of PMTR values got from ARU; read and write access by the CPU is only possible, when the DPLL is disabled.
  The CPU Address range: 0x0200 – 0x03FC
- **RAM Region 1b**: usable for intermediate calculations and auxiliary values, data width of 3 bytes used for 24 bit values; A write access to this region results in an interrupt to the CPU, when enabled.
  Address range: 0x0400 – 0x05FC
- **RAM Region 1c**: Values of all STATE increments in FULL_SCALE, data width of 3 bytes used for 24 bit values; A write access to this region results in an interrupt to the CPU, when enabled.
  Address range: 0x0600 – 0x09FC

In RAM region 1c there is a difference in the amount of data. While for the RAM regions 1c1, 1c3 and 1c4 there are **2*(SNU+1-SYN_NS)** entries for SYSF=0 or **2*(SNU+1) -SYN_NS** entries for SYSF=1, for the RAM region 1c2 there are **2*(SNU+1)** entries (see DPLL_CTRL_0 and _1 registers). For the latter also the virtual events are considered, that means the gap is divided into equidistant parts each having the same position share as increments without a gap. For that reason the CPU must extend the stored TSF_S[i] values in the RAM region 1c2 before the APS_1C3 is written. The write access to APS_1C3 sets the SYS bit in the DPLL_Status register in order to show the end of the synchronization process. Only when the SYS bit is set the PMTR values can consider more than the last increment duration for the action prediction by setting NUSE to a corresponding value.

Note: RAM regions 1b and 1c have a common port.

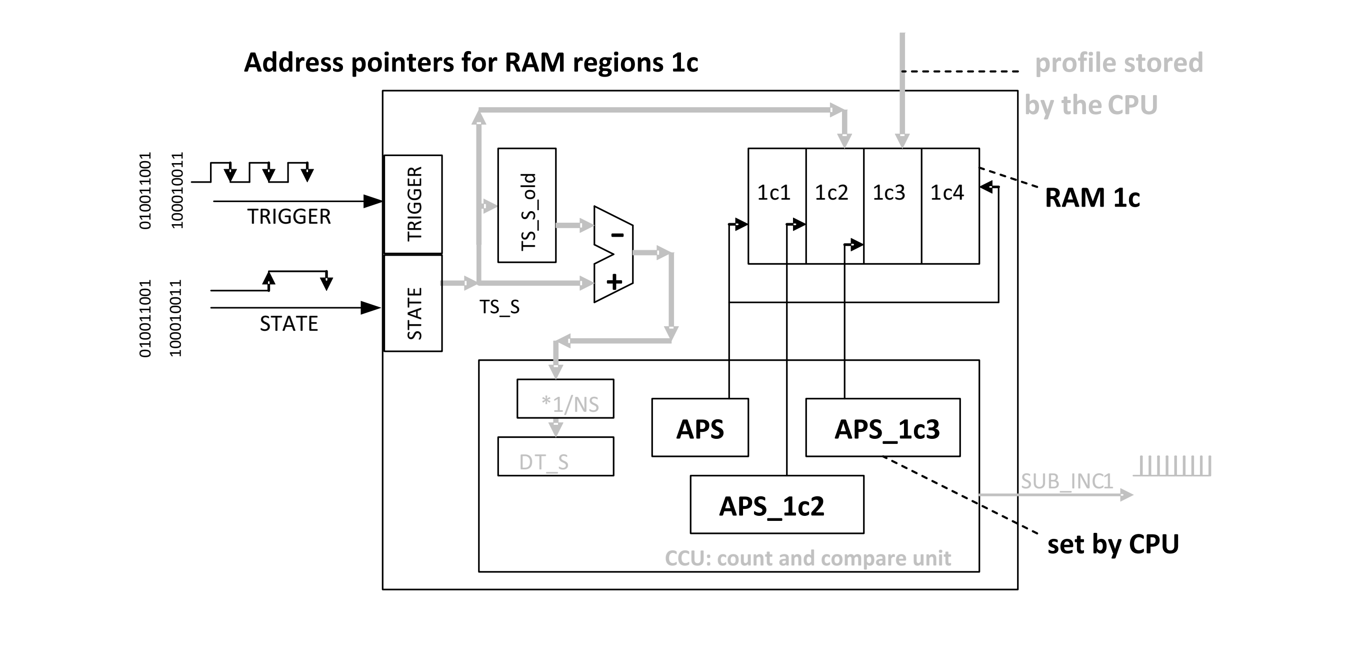


Figure 126 Address Pointer for RAM 1c

The address pointers for RAM region 1c are shown in the diagram above. While the address pointer APS points to the RAM regions 1c1 and 1c4, the address pointer APS_1C2 points to the time stamp field in the region 1c2. This is necessary, because in the time stamp field the gaps are extended to the number of nominal increments (see explanation above and also to the synchronization procedure explained in **Section 28.20.8.6.1**).The address pointer APS_1C3 is set by the CPU when the position is known and therefore the relation to the other address pointers is calculated. This setting of this profile address pointer synchronizes the RAM fields to one another. The synchronization is shown in the DPLL_STATUS register (see **Register DPLL_STATUS**) by the SYS bit.

28.20.5.10.2 RAM Region 2

The RAM region 2 has a configurable size of 1,5 to 12 Kilobytes and is used to store measured and calculated values for increments of *TRIGGER*. The address information is explained in **Section 28.20.11** while the meaning is explained in this chapter.

Because of up to 512 *TRIGGER* events in HALF_SCALE the fields 2a, b c and d must have up to 1024 storage places each. For 3 Bytes word size this does mean up to 12 k Byte of RAM region 2.

In order to save RAM size for configurations with less *TRIGGER* events the RAM is configurable by the offset switch Register OSW (0x001C) and the address offset value register of RAM region 2 AOSV_2 (0x0020). The RAM is to be initialized by the DPLL after HW-reset. All RAM cells must have a zero value after performing the initialize procedure. The DPLL is only available after finishing this procedure.

In RAM region 2 there is a difference in the amount of data. While for the RAM regions 2a, 2c and 2d there are **2*(TNU+1-SYN_NT)** entries, for the RAM region 2b there are **2*(TNU+1)** entries (see DPLL_CTRL_0 and _1 registers). For the latter also the virtual events are considered, that means the gap is divided into equidistant parts each having the same position share as increments without a gap. For that reason the CPU must extend the stored TSF_T[i] values in the RAM region 2b before the APT_2C is written.

The write access to APT_2C sets the SYT bit in the DPLL_Status register in order to show the end of the synchronization process. Only when the SYT bit is set the PMTR values can consider more than the last increment duration for the action prediction by setting NUTE to a value greater than one.

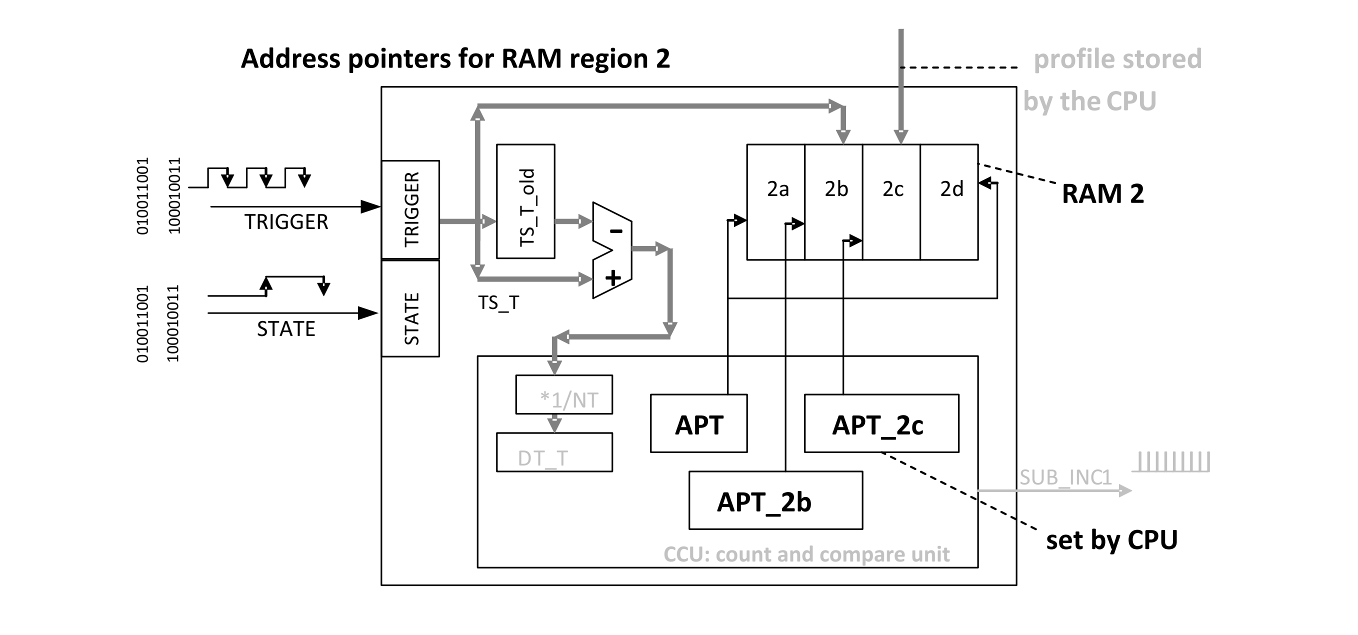


Figure 127 Address Pointer for RAM 2

The address pointers for RAM region 2 are shown in the diagram above. While the address pointer APT points to the RAM regions 2a and 2d, the address pointer APT_2B points to the time stamp field in the region 2b. This is necessary, because in the time stamp field the gaps are extended to the number of nominal increments (see explanation above and also to the synchronization procedure explained in **Section 28.20.8.6.1**).

The address pointer APT_2C is set by the CPU when the position is known and therefore the relation to the other address pointers is calculated. This setting of this profile address pointer synchronizes the RAM fields to one another. The synchronization is shown in the DPLL_STATUS register (see **Register DPLL_STATUS**) by the SYT bit.

28.20.5.11Software reset and DPLL deactivation

The DPLL module allows different options of deactivation and/or reset.To stop the operation of the DPLL module it is possible to deactivate the DPLL by setting of DPLL_CTRL_1.DEN = 0. This stops the calculations for the generation of the sub increments and the actions. Some control register areas are only configurable in this mode, some but not all register signals are set into an initial state. The RAM memory is not affected by DPLL deactivation at all. The behavior of the DPLL output signals and registers when deactivated is described in this document.

The deeper option to reconfigure the DPLL is the use of the software reset. When the DPLL module is deactivated setting DPLL_CTRL_1.SWR = 1 performs a reset of all DPLL registers and state controllers. The RAM memory is not affected by the software reset at all. After the software reset the DPLL module remains in deactivated state and the control registers must be configured again before operation (activation by DEN = 1) can start again.

The RAM modules can be reset (written to all zero) by activation of the memory init control bit (INIT_RAM) of the register DPLL_RAM_INI.If the RAM initialization is automatically done after power on reset or not depends on the GTM implementation.

A special case is the configuration of the control bit DPLL_CTRL_11.STATE_EXT. If this bit shall be modified during operation a software reset of the DPLL module is strongly recommended. A RAM initialisation should also be considered depending on the given application case.

28.20.6 Prediction of the current increment duration

28.20.6.1 The use of increments in the past

Past values to be considered for the prediction of TRIGGER

In order to take into account values of increments for *TRIGGER*s in the past, the NUTE value is configured to determine the number of past values. In addition the VTN has a value according to the number of virtual increments in the NUTE region. Because gaps come in to the NUTE region or leave it the VTN value must be updated by the CPU until NUTE is set to HALF_SCALE or FULL_SCALE. For the RAM regions 2a and 2d the value NUTE-VTN is to be considered while for the RAM region 2b only the NUTE value is to be considered. This is due to the fact that the time stamp entries in a gap are extended to the number of nominal increments, but duration entries not.

Past values to be considered for the prediction of STATE

In order to take into account values of increments for *STATE* in the past, the NUSE value is configured to determine the number of past values. In addition the VSN has a value according to the number of virtual increments in the NUSE region. Because gaps come in to the NUSE region or leave it the VSN value must be updated by the CPU until NUSE is set to HALF_SCALE or FULL_SCALE. For the RAM regions 1c1 and 1c4 in the past the value NUSE-VSN is to be considered while for the RAM region 1c2 only the NUSE value is to be considered. This is due to the fact that time stamp entries in a gap are extended to the number of nominal increments, but duration entries not.

28.20.6.2 Increment prediction in Normal Mode and for first PMSM forwards

For the prediction of increments and actions in normal mode the values are calculated as described in the following equations.

Please note, that the ascending order of calculation must be hold in order not to lose results still needed. It is important for *TRIGGER* values to calculate and store in the RAM region 2 all values according to equations up to DPLL-14 before DPLL-1a4…7, DPLL-1b1 and DPLL-1c1, while the last one overwrites DT_T[i] when NUTE (see **Register DPLL_NUTC**) is set to the FULL_SCALE range. Because the old value of DT_T[i] is also needed for equation DPLL-10 and DPLL-11 this value is stored temporarily at DT_T_ACT as shown by equation DPLL-1a or DPLL-1b respectively until all prediction calculations are done and after that equation DPLL-1a4…7, DPLL-1b1 and DPLL-1b1 updates DT_T[i]: update DT_T[i] after calculations of equation DPLL-14. For p=APT calculates in normal mode.

When using filter information of TRIGGER_FT, selected by IDT=1, it must be distinguished by IFP, if this filter information is time or position related.

In order to make possible to perform the automatic resolution corrections of equations DPLL-1a1a the filter unit in TIM module must operate using the time stamp clock.

28.20.6.2.1 Equations DPLL-1a to calculate TRIGGER time stamps

For calculation of time stamps use the filter delay information and an additional TRIGGER input delay value stored in register TIDEL (initial zero)

- TS_T_1=TRIGGER_TS - TIDEL **(DPLL-1a0)**
- TS_T=TS_T_1- FTV_Tx (for IDT=1 and IFP=0) **(DPLL-1a1)**

with

- FTV_Tx = FTV_T/8 (for LOW_RES = 1 and TS0_HRT = 0) **(DPLL-1a1a)**
- FTV_Tx = FTV_T (for LOW_RES = 0 or TS0_HRT = 1) **(DPLL-1a1b)**

and

- TS_T= TS_T_1- FTV_T *(CDT_TX/NMB_T)_old[1)] for (IDT=1 and IFP=1) **(DPLL-1a2)**

this can be also calculated using the value of ADD_IN_CALN:

- TS_T=TS_T_1 - FTV_T *(1/ADD_IN_CALN_old[1)]) for (IDT=1 and IFP=1) **(DPLL-1a3)**

[1)] Consider values, calculated for the last increment; position related filter values are only considered up to at least 1 ms time between two TRIGGER events. The reciprocal value is stored using a 32-bit fractional part, while only the 24 lower bits are used - for explanation see note 4) at DPLL_CTRL_0 register. The value of 1/ADD_IN_CALN_old or (CDT_TX/NMB_T)_old is set to 0xFFFFFF in the case of an overflow.

Note: *CDT_TX is the predicted duration of the last TRIGGER increment and NMB_T the calculated number of SUB_INC1 events in the last increment, because the new calculations are done by equations DPLL-5 and DPLL-21 for the current increment after that. Therefore in equation DPLL-1a3 the value ADD_IN of the last increment is used (see equation DPLL-25). SYN_T_OLD is the number of TRIGGER events including missing TRIGGERs as specified in the NUTC register for the last increment, with the initial value of 1.*

For storage of time stamps in the RAM see also equations DPLL-1a4 ff. after calculation of actions, **Section 28.20.7.5.1**.

28.20.6.2.2 Equation DPLL-1b to calculate DT_T_ACT (nominal value)

- DT_T_ACT= (TS_T - TS_T_OLD)/SYN_T_OLD **(DPLL-1b)**

For the case SYT=0 (still no synchronization to the profile) the values SYN_T and SYN_T_OLD are still assumed as having the value 1.

Correct the current increment duration value got by equation DPLL-1b in the case of physical deviations (ADT=1) by

- DT_T_ACT = DT_T_ACT * (1 - PDC_T+PDC_T^2+PDC_T^3) **(DPLL-1b1)**

with the relative correction value for the last increment (for SMC=0)

- PDC_T = PD_OLD/(MLT+1) **(DPLL-1b2)**

for SMC=1 use

- PDC_T = PD_OLD/(MLS1) (DPLL-1b3)

Note: *The term (1 - PDC_T+PDC_T^2+PDC_T^3) is representing the third order Taylor series of the term 1/(1+PCT_T), which is chosen to reduce the additional time delay due to the more complex computation.*

28.20.6.2.3 Equation DPLL-1c to calculate RDT_T_ACT (nominal value)

- RDT_T_ACT= 1/ DT_T_ACT **(DPLL-1C)**

28.20.6.2.4 Equation DPLL-2a1 to calculate QDT_T_ACT

Relation of the recent last two increment values for APT=p in forward direction (DIR1=0)

- QDT_T_ACT = DT_T_ACT * RDT_T[p-1] **(DPLL-2a1)**

QDT_T_ACT as well as QDT_T[i] have a 24 bit value using a 6 bit integer part and an 18 bit fractional part.

Note: QDT_T_ACT uses a 6 bit integer part and an 18 bit fractional part.

28.20.6.2.5 Equation DPLL-3 to calculate the error of last prediction

When q = NUTE-VTN considers for the error calculation only the last valid prediction values for DIR1=0:
Calculate the error of the last prediction when using only RDT_T_FS1, DT_T[p-q], DT_T[p-q-1] and DT_T[p-1] for the prediction of DT_T[p]:

- EDT_T=DT_T_ACT-(DT_T[p-1]*QDT_T[p-q]) **(DPLL-3)**

with

- QDT_T[p-q] = DT_T[p-q] * RDT_T[p-q-1] for FST=0 **(DPLL-2b1)**
- QDT_T[p-q] = DT_T[p-q] * RDT_T_FS1 for FST=1 **(DPLL-2b2)**

and FST has the meaning: NUTE=FULL_SCALE (see NUTC register)
while
QDT_T_ACT as well as QDT_T[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Note: QDT_T[p-q] uses a 6 bit integer part and a 18-bit fractional part.

28.20.6.2.6 Equation DPLL-4 to calculate the weighted average error

for SYT=1 calculate:

- MEDT_T := (EDT_T + MEDT_T)/2 **(DPLL-4)**

28.20.6.2.7 Equations DPLL-5 to calculate the current increment value

nominal increment value (for ADT=0):

- CDT_TX _nom = (DT_T_ACT + MEDT_T)* QDT_T[p-q+1] **(DPLL-5a1)**

nominal increment value (for ADT=1):

- CDT_TX _nom_corr = CDT_TX _nom*(1 + CDC_T) **(DPLL-5a2)** with for SMC=0

- CDC_T = PD/ (MLT+1) **(DPLL-5a3)**

or for SMC=1 use

- CDC_T = PD/ (MLS1) **(DPLL-5a4)**

and with (for q>1):

- QDT_T[p-q+1] = DT_T[p-q+1] * RDT_T[p-q] **(DPLL-2c)**

and for q=1 use equation DPLL-2a1.

while
QDT_T_ACT as well as QDT_T[i] have a 24 bit value using a 6 bit integer part and an 18 bit fractional part.

The CDT_TX_nom value is limited by the relation

- CTN_MIN < CDT_TX_nom < CTN_MAX. **(DPLL-5c)**

When the calculated value exceeds one of the limit, it is replaced by the corresponding limit value.

The expected duration to the next TRIGGER event is (for ADT=0)

- CDT_TX = CDT_TX _nom * SYN_T **(DPLL-5b)**

The expected duration to the next TRIGGER event is (for ADT=1)

- CDT_TX = CDT_TX _nom_corr * SYN_T **(DPLL-5b)**

Note: QDT_T[p-q+1] uses a 6-bit integer part and a 18 bit fractional part.

Note: In the case of an overflow in equations DPLL-5a or b set the value to 0xFFFFFF and the corresponding CTO bit in the DPLL_STATUS register. In the case of negative values set CDT_TX to 0x0 without any effect to the CTO bit.

28.20.6.3 Increment prediction in Emergency Mode and for second PMSM forwards

Please note, that the ascending order of calculations for *STATE* and storage of the values in the RAM region 1c must be hold in order not to lose results still needed. The same considerations as done for DT_T_ACT are valid for DT_S_ACT (equation DPLL-6a4…7, DPLL-6b1 and DPLL-6b1): update TD_S[i] only after calculations of equation DPLL-14.

When using filter information of STATE_FT, selected by IDS=1, it must be distinguished by IFP, if this filter information is time or position related.

In order to make possible to perform the automatic resolution corrections of equations DPLL-6a1a the filter unit in TIM must operate using the time stamp clock.

28.20.6.3.1 Equations DPLL-6a to calculate STATE time stamps

For calculation of time stamps use the filter delay information, the additional STATE input delay value stored in the register SIDEL (initial zero) and use p=APS while DIR2=0:

- TS_S_1=STATE_TS - SIDEL **(DPLL-6a0)**
- TS_S=TS_S_1 - FTV_Sx (for IDS=1 and IFP=0) **(DPLL-6a1)**

with

- FTV_Sx = FTV_S/8 (for LOW_RES = 1 and TS0_HRS = 0) **(DPLL-6a1a)**
- FTV_Sx = FTV_S (for LOW_RES = 0 or TS0_HRS = 1) **(DPLL-6a1b)**

and

- TS_S=TS_S$_1$ - FTV_S *(CDT_SX/NMB_S)_old[1] (for IDS=1 and IFP=1) **(DPLL-6a2)**

this can be also calculated using the value of ADD_IN_CALE:

- TS_S=TS_S$_1$ - FTV_S *(1/ADD_IN_CALE)_old[1] (for IDS=1 and IFP=1) **(DPLL-6a3)**

with

see also equations DPLL-6a4 ff. at **Section 28.20.6.2** for TRIGGER.

[1] Consider values, calculated for the last increment; position related filter values are only considered up to at least 1 ms time between two STATE events. The reciprocal value is stored using a 32 bit fractional part, while only the 24 lower bits are used - for explanation see note 4) at DPLL_CTRL_0 register. The value of 1/ADD_IN_CALE_old or (CDT_SX/NMB_S)_old is set to 0xFFFFFF in the case of an overflow.

Note: *CDT_SX is the predicted duration of the last STATE increment and NMB_S the calculated number of SUB_INC1 events in the last increment, because the new calculations are done by equations DPLL-10 and DPLL-22 respectively for the current increment after that. Therefore in equation DPLL-6a3 the value ADD_IN of the last increment is used (see equation DPLL-26). SYN_S_OLD is the number of increments including missing STATEs as specified in the NUSC register for the last increment with the initial value of 1. The update to the RAM region 1c4 is done after all related calculations (see equation DPLL-6b1 for this reason).*

28.20.6.3.2 Equation DPLL-6b to calculate DT_S_ACT (nominal value)

- DT_S_ACT= (TS_S - TS_S_OLD)/SYN_S_OLD **(DPLL-6b)**

For the case SYS=0 (still no synchronization to the profile) the values SYN_S and SYN_S_OLD are still assumed as having the value 1.

Correct the current increment duration value got by equation DPLL-6b in the case of physical deviations (ADS=1) by

- DT_S_ACT = DT_S_ACT * (1 - PDC_S+PDC_S2+PDC_S3) **(DPLL-6b1)**

with the relative correction value for the last increment (for SMC=0)

- PDC_S = PD_S_OLD/(MLS1) **(DPLL-6b2)**

for SMC=1 use

- PDC_S = PD_S_OLD/(MLS2) **(DPLL-6b3)**

Note: *The term (1 - PDC_S+PDC_S2+PDC_S3) is representing the third order Taylor series of the term 1/(1+PDC_S), which is chosen to reduce the additional time delay due to the more complex computation.*

28.20.6.3.3 Equation DPLL-6c to calculate RDT_S_ACT (nominal value)

- RDT_S_ACT = 1/ DT_S_ACT **(DPLL-6c)**

28.20.6.3.4 Equation DPLL-7a1 to calculate QDT_S_ACT

for APS=p in forward direction (DIR2=0)

- QDT_S_ACT = DT_S_ACT * RDT_S[p-1] **(DPLL-7a1)**

QDT_S_ACT as well as QDT_S[i] have a 24 bit value using a 6 bit integer part and an 18 bit fractional part.

Note: QDT_S_ACT uses a 6 bit integer part and a 18 bit fractional part.

28.20.6.3.5 Equation DPLL-8 to calculate the error of last prediction

with q= NUSE-VSN when using QDT_S[p-q] and DT_S[p-1] for the prediction of DT_S[p]

- EDT_S = DT_S_ACT - (DT_S[p-1] * QDT_S[p-q]) **(DPLL-8)**

and with

- QDT_S[p-q] = DT_S[p-q] * RDT_S[p-q-1] for FSS=0 **(DPLL-7b1)**
- QDT_S[p-q] = DT_S[p-q] * RDT_S_FS1 for FSS=1 **(DPLL-7b2)**

and FSS has the meaning: NUSE=FULL_SCALE (see NUSC register)

QDT_S_ACT as well as QDT_S[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Note: QDT_S[p-q] uses a 6 bit integer part and a 18 bit fractional part.

28.20.6.3.6 Equation DPLL-9 to calculate the weighted average error

for SYS=1 calculate:

- MEDT_S := (EDT_S + MEDT_S)/2 **(DPLL-9)**

28.20.6.3.7 Equations DPLL-10 to calculate the current increment (nominal value)

nominal increment value (for ADS=0):

- CDT_SX _nom= (DT_S_ACT + MEDT_S)* QDT_S[p-q+1] **(DPLL-10a1)**

or nominal increment value (for ADS=1):

- CDT_SX _nom_corr= CDT_SX _nom*(1+CDC_S) **(DPLL-10a2)**

with for SMC=0

- CDC_S = PD / (MLS1) **(DPLL-10a3)**

for SMC=1 use

- CDC_S = PD / (MLS2) **(DPLL-10a4)**

and with

- QDT_S[p-q+1] = DT_S[p-q+1] * RDT_S[p-q] (for q>1) **(DPLL-7c)**
  see equation DPLL-7a1 for q=1

while

QDT_S_ACT as well as QDT_S[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

The CDT_SX_nom value is limited by the relation

- CSN_MIN < CDT_SX_nom < CSN_MAX **(DPLL-10c)**

When the calculated value exceeds one of the limit, it is replaced by the corresponding limit value.

The expected duration to the next *STATE* event is (for ADT=0)

- CDT_SX = CDT_SX _nom * SYN_T **(DPLL-10b)**

The expected duration to the next *STATE* event is (for ADT=1)

- CDT_SX = CDT_SX _nom_corr * SYN_S **(DPLL-10b)**

Note: QDT_S[p-q+1] uses a 6 bit integer part and a 18 bit fractional part.

Note: In the case of an overflow in equations DPLL-10a or b set the value to 0xFFFFFF and the corresponding CSO bit in the DPLL_STATUS register. In the case of negative values set CDT_SX to 0x0 without any effect to the CSO bit. All 5 steps above (DPLL-6 to DPLL-10) are only needed in emergency mode. For the normal mode the calculations of equations DPLL-6 and DPLL-7 are done solely in order to get the values needed for a sudden switch to emergency mode.

28.20.6.4 Increment prediction in Normal Mode and for first PMSM backwards

28.20.6.4.1 Equations DPLL-2a2 to calculate QDT_T_ACT backwards

- QDT_T_ACT = DT_T_ACT * RDT_T[p+1] **(DPLL-2a2)**

QDT_T_ACT as well as QDT_T[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

28.20.6.4.2 Equation DPLL-3a to calculate of the error of last prediction

When q = NUTE-VTN and DIR1=1 using only QT_T[p+q] and DT_T[p+1] for the prediction of DT_T[p]

- EDT_T=DT_T_ACT-(DT_T[p+1]*QDT_T[p+q] **(DPLL-3a)**

with

- QDT_T[p+q] = DT_T[p+q] * RDT_T[p+q+1] for FST=0 **(DPLL-2b3)**
- QDT_T[p+q] = DT_T[p+q] * RDT_T_FS1 for FST=1 **(DPLL-2b4)**

and FST has the meaning: NUTE=FULL_SCALE (see NUTC register)

QDT_T_ACT as well as QDT_T[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

28.20.6.4.3 Equation DPLL-4 to calculate the weighted average error

For SYT=1 calculate: MEDT_T := (EDT_T + MEDT_T) / 2 (DPLL-4)

28.20.6.4.4 Equation DPLL-5 to calculate the current increment value

nominal increment value (for ADT=0):

- CDT_TX _nom = (DT_T_ACT + MEDT_T)* QDT_T[p+q-1] **(DPLL-5a5)**

nominal increment value (for ADT=1):

- CDT_TX _nom_corr = CDT_TX _nom * (1 + CDC_T) **(DPLL-5a6)**

with for SMC=0

- CDC_T = PD / (MLT + 1) **(DPLL-5a3)**

for SMC=1 use

- CDC_T = PD / (MLS1) **(DPLL-5a3)**

and with

- QDT_T[p+q-1] = DT_T[p+q-1] * RDT_T[p+q] (for q>1) **(DPLL-2c1**

for q=1 use equation DPLL-2a1.

while

QDT_T_ACT as well as QDT_T[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

The CDT_TX_nom value is limited by the relation

- CTN_MIN < CDT_TX_nom < CTN_MAX. **(DPLL-5c)**

When the calculated value exceeds one of the limit, it is replaced by the corresponding limit value.
and the expected duration to the next TRIGGER event (for ADT=0)

- CDT_TX = CDT_TX _nom * SYN_T **(DPLL-5b)**

the expected duration to the next TRIGGER event (for ADT=1)

- CDT_TX = CDT_TX _nom_corr * SYN_T **(DPLL-5b)**

Note: *In the case of an overflow in equations DPLL-5a1 or b set the value to 0xFFFFFF and the corresponding CTO bit in the DPLL_STATUS register. In the case of negative values set CDT_TX(_nom) to 0x0.*

28.20.6.5 Increment prediction in Emergency Mode and for second PMSM backwards

28.20.6.5.1 Equation DPLL-7a2 to calculate QDT_S_ACT backwards

- QDT_S_ACT = DT_S_ACT * RDT_S[p+1] **(DPLL-7a2)**

QDT_S_ACT as well as QDT_S[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

28.20.6.5.2 Equation DPLL-8a to calculate the error of the last prediction

While q= NUSE-VSN, use only QDT_S[p+q] and DT_S[p+1] for the prediction of DT_S[p]

- EDT_S = DT_S_ACT - (DT_S[p+1] * QDT_S[p+q]) **(DPLL-8a)**

with

- QDT_S[p-q] = DT_S[p+q] * RDT_S[p+q+1] for FSS=0 **(DPLL-7b3)**
- QDT_S[p-q] = DT_T[p+q] * RDT_S_FS1 for FSS=1 **(DPLL-7b4)**

and FSS has the meaning: NUSE=FULL_SCALE (see NUSC register)

QDT_S_ACT as well as QDT_S[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

28.20.6.5.3 Equation DPLL-9 to calculate the weighted average error

For SYS=1 calculate:

- MEDT_S := (EDT_S + MEDT_S)/2 **(DPLL-9)**

28.20.6.5.4 Equations DPLL-10 to calculate the current increment value

nominal increment value (for ADS=0):

- CDT_SX _nom= (DT_S_ACT + MEDT_S)* QDT_S[p+q-1] **(DPLL-10a5)**

or nominal increment value (for ADS=1):

- CDT_SX _nom_corr= CDT_SX _nom*(1+CDC_S) **(DPLL-10a6)**

with for SMC=0

- CDC_S = PD / (MLS1) **(DPLL-10a3)**

for SMC=1 use

- CDC_S = PD / (MLS2) **(DPLL-10a4)**

and with

- QDT_S[p+q-1] = DT_S[p+q-1] * RDT_S[p+q] (for q>1) **(DPLL-7c1)**

  for q=1 use equation DPLL-7a.

while

QDT_S_ACT as well as QDT_S[i] have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

The CDT_SX_nom value is limited by the relation

- CSN_MIN < CDT_SX_nom < CSN_MAX. **(DPLL-10c)**

When the calculated value exceeds one of the limit, it is replaced by the corresponding limit value.

and calculate the expected duration to the next *STATE* event (for ADT=0)

- CDT_SX = CDT_SX _nom * SYN_T **(DPLL-10b)**

and calculate the expected duration to the next *STATE* event (for ADT=1)

- CDT_SX = CDT_SX _nom_corr * SYN_S **(DPLL-10b)**

Note: In the case of an overflow in equations DPLL-10a1 or b set the value to 0xFFFFFF and the corresponding CSO bit in the DPLL_STATUS register. In the case of negative values set CDT_SX(_nom) to 0x0.

All 5 steps above (DPLL-6 to DPLL-10) are only needed in emergency mode. For the normal mode the calculations of equations DPLL-6 and DPLL-7 are done solely in order to get the values needed for a sudden switch to emergency mode.

28.20.7 Calculations for actions

As already shown for the calculation of the current interval by equations DPLL-1 to DPLL-10 for the prediction of actions a similar calculation is to be done, as shown by the equations DPLL-11. to DPLL-14. The calculation of actions is also needed when the DPLL is used for synchronous motor control applications (SMC=1, see DPLL_CTRL_1 register). For action prediction purposes the measured time periods of the past (one FULL_SCALE back, when the corresponding NUTE or NUSE values are set properly by the CPU) are used. The calculation can be explained by the following assumptions, which are considerably simple:

Take the corresponding increments for prediction in the past and put the sum of it in relation to the increment (DT_T[k], DT_S[k], with k >= 0, which is represented by the time stamp difference) which is exactly one FULL_SCALE period in the past (DPLL-11 or DPLL-13 respectively). Make a prediction for the coming sum of increments using the current measured increment (DT_T_ACT or DT_S_ACT respectively, that means DPLL-1 or DPLL-6 respectively) and add a weighted average error (DPLL-3 and DPLL-4 or DPLL-8 and DPLL-9 respectively, calculated for one increment prediction) before multiplication with the relation of equation DPLL-11 or DPLL-13 respectively in order to get the result as described by equations DPLL-12 or DPLL-14 respectively.

In order to avoid division operations instead of the increment (DT_T[k], DT_S[k], with k > 0) in the past its reciprocal value (RDT_T[k], RDT_S[k], with k > 0) is used, which is stored also in RAM. For the calculation of actions perform always a new refined calculation as long as the resulting time stamp is not in the past. In the other case the TSAC/PSAC values (time/position stamp of action calculated) is set to the time/position stamp of the last input event (TRIGGER/STATE), the ACT_N[i] bit in the DPLL_ACT_STA register is reset, while the corresponding

ACT_N[i] bit in the DPLL_ACT_STA_shadow register is set. Each new PMTR_i value will set this ACT_N[i] bit again and reset the correspondent shadow bit until a new calculation is performed.

Please make sure that the prediction parameters are chosen such that under all conditions (acceleration/deceleration) the values of PDT_T, PDT_S and DTA respectively do not exceed the value 0xFFFFFF. This requirement can limit the predicted position range in the case of very low speed.

Action updates at highest speed

Up to 32 action values can be calculated. For the shortest increment duration (23,4 µs) not all of them can be updated with each active input event. Please notice the following conditions and parameters for an estimation of possible results.

All time estimation values are given for a system clock frequency of 100 MHz and the assumption, that the calculation of the DPLL is not impeded by a remote read or write access to the DPLL RAMs. Each RAM access is to be considered by an additional delay of about 40 ns ($t_{remote\_RAM\_acces}$, to be precised later). When using a different system clock frequency the calculation duration is extended accordingly.

1. Typical time needed for basic operations (RAM update, pointer calculation and SUB_INC generation for normal, emergency mode or one PMSM: t_{basic_0} = 9,9 µs.
2. Typical time needed basic operations (RAM update, pointer calculation and SUB_INC generation for two PMSMs): t_{basic_1} = 11,0 µs.
3. Typical time needed to calculate one action: t_{action_i} = 3,7 µs.

Please notice that the above mentioned values are observed worst case values, when the two state machines of TRIGGER and STATE are both in operation.

These values allow the calculation of at least 3 action values for each input event for all specified increments duration. The complete time needed for the basic operation, n action calculations and k remote RAM access operations can be calculated as follows:

$t_{complete} = t_{basic\_0/1} + n * t_{action}\_i + k * t_{remote\_RAM\_access}$.

Typical applications

Normal and emergency mode

For a typical application with the shortest increment duration of 100 µs in normal or emergency mode the calculation of up to 24 action values can be performed for each active input event.

One PMSM

For one PMSM and a typical shortest increment time of 39 µs there is the calculation of up to 7 action values possible for each input event.

Two PMSMs with restricted action calculations

When only one PMSM uses the action calculation service an the shortest increment duration is 39 µs, there can up to 7 actions served for each valid input event.

Two PMSMs with unrestricted action calculations

When 2 PMSMs are used and both use the action calculation service at a minimal increment duration of 39 µs there are up to 7 action calculations possible for each of the two engines - that means up to 14 action calculations per increment in average.

28.20.7.1 Action calculations for TRIGGER forwards

valid for RMO=0 or for SMC=1 with

p=APT_2B, t=APT, m=NA[i] (part w), mb=NA[i](part b)/1024, NUTE-VTN=q, NUTE=n

28.20.7.1.1 Equation DPLL-11a1 to calculate the time prediction for an action

For DIR1=0 and q>m calculate:

- PDT_T[i] = (TSF_T[p+m-n] - TSF_T[p-n] + mb* DT_Tx[t-q+1]) * RDT_T[t-q] **(DPLL-11a1)**

with

- DT_Tx[t-q+1] = DT_T[t-q+1] for TS0_HRT=0 **(DPLL-11b2)**

or

- DT_Tx[t-q+1] = DT_T[t-q+1]/8 for TS0_HRT=1 **(DPLL-11b3)**

and while the multiplication with mb does mean the fractional part of NA[i].

For SMC=0 and RMO=0 calculate for DIR1=0 all 32 actions in forward direction, if requested; in the case SMC=1 calculate up to 16 actions 0 to 15 in dependence of the *TRIGGER* input.

28.20.7.1.2 Equation DPLL-11a2 to calculate the time prediction for an action

For SYT=1, NUTE= 2*(TNU+1), q>m and DIR1=0 equation DPLL-11a2 is equal to

- PDT_T[i] = (TSF_T[p+m] - TSF_T[p]+ mb* DT_Tx[t-q+1]) * RDT_T[t] **(DPLL-11a2)**

with

- DT_Tx[t-q+1] = DT_T[t-q+1] for TS0_HRT=0 **(DPLL-11b2)**

or

- DT_Tx[t-q+1] = DT_T[t-q+1]/8 for TS0_HRT=1 **(DPLL-11b3)**

28.20.7.1.3 Equation DPLL-11b to calculate the time prediction for an action

for DIR1=0, NUTE-VTN=q, q (< or =) m, n>1 and t=APT:

- PDT_T[i] = (m+mb)*DT_Tx[t-q+1] * RDT_T[t-q] **(DPLL-11b)**

with

- DT_Tx[t-q+1] = DT_T[t-q+1] for TS0_HRT=0 **(DPLL-11b2)**

or

- DT_Tx[t-q+1] = DT_T[t-q+1]/8 for TS0_HRT=1 **(DPLL-11b3)**

Note: Make the calculations above before updating the TSF_T[i] values according to equations DPLL-1c3 ff.

28.20.7.1.4 Equation DPLL-11c to calculate the time prediction for an action

for n=1 (this is always valid for SYT=0)

- PDT_T[i] = (m+mb)* DT_T_ax* RDT_T[t-1] **(DPLL-11c)**

with

- DT_T_ax = DT_T_ACT for TS0_HRT=0 **(DPLL-1a4a)** or

- DT_T_ax = DT_T_ACT/8 for TS0_HRT=1 **(DPLL-1a4b)**

Note: For the relevant last increment add the fractional part of DT_T_ACT as described in NA[i].

28.20.7.1.5 Equation DPLL-12 to calculate the duration value until action

DTA[i] = (DT_T_ACT + MEDT_T)* PDT_T[i] **(DPLL-12)**

Note: All 5 steps in equations DPLL-11 to DPLL-12 are only calculated in normal mode.

28.20.7.2 Action calculations for TRIGGER backwards

valid for RMO=0 or for SMC=1 with

p=APT_2B, t=APT, m=NA[i] (part w), mb=NA[i](part b)/1024, q= NUTE-VTN and n=NUTE

For SMC=0 and RMO=0 calculate for DIR1=1 all 32 actions in backward direction for special purposes; in the case SMC=1 calculate up to 16 actions 0 to 15 in dependence of the *TRIGGER* input.

28.20.7.2.1 Equation DPLL-11a3 to calculate the time prediction for an action

For DIR1=1 and q>m calculate:

- PDT_T[i] = (TSF_T[p-m+n] - TSF_T[p+n] + mb* DT_Tx[t+q-1]) * RDT_T[t+q] **(DPLL-11a3)**

with

- DT_Tx[t+q-1] = DT_T[t+q-1] for TS0_HRT=0 **(DPLL-11b4)**

or

- DT_Tx[t+q-1] = DT_T[t+q-1]/8 for TS0_HRT=1 **(DPLL-11b5)**

28.20.7.2.2 Equation DPLL-11a4 to calculate the time prediction for an action

For SYT=1 and NUTE = 2*(TNU+1), q>m, VTN=2*SYN_NT and hence **NUTE-VTN = 2*(TNU+1-SYN_NT)** for DIR1=1 this is equal to

- PDT_T[i] = (TSF_T[p-m] - TSF_T[p]+ mb* DT_Tx[t+q-1]) * RDT_T[t] **(DPLL-11a4)**

with

- DT_Tx[t+q-1] = DT_T[t+q-1] for TS0_HRT=0 **(DPLL-11b4)**

or

- DT_Tx[t+q-1] = DT_T[t+q-1]/8 for TS0_HRT=1 **(DPLL-11b5)**

Note: Make the calculations above before updating the TSF_T[i] values according to equations DPLL-1c3 ff.

28.20.7.2.3 Equation DPLL-11b1 to calculate the time prediction for an action

For NUTE-VTN =q, q (< or =) m the following equation is valid for n>1 and t=APT:

- PDT_T[i] = (m+mb)*DT_Tx[t+q-1] * RDT_T[t+q] **(DPLL-11b1)**

with

- DT_Tx[t+q-1] = DT_T[t+q-1] for TS0_HRT=0 **(DPLL-11b4)**

or

- DT_Tx[t+q-1] = DT_T[t+q-1]/8 for TS0_HRT=1 **(DPLL-11b5)**

28.20.7.2.4 Equation DPLL-11c1 to calculate the time prediction for an action

for n=1 (this is always valid for SYT=0)

- PDT_T[i] = (m+mb)* DT_T_ax * RDT_T[t+1] **(DPLL-11c1)**

with

- DT_T_ax = DT_T_ACT for TS0_HRT=0 **(DPLL-1a4a)**

or

- DT_T_ax = DT_T_ACT/8 for TS0_HRT=1 **(DPLL-1a4b)**

Note: For the relevant last increment add the fractional part of DT_T_ACT as described in NA[i].

28.20.7.2.5 Equation DPLL-12 to calculate the duration value for an action

DTA[i] = (DT_T_ACT + MEDT_T)* PDT_T[i] **(DPLL-12)**

Use the results of equations DPLL-1a, b, DPLL-3 and DPLL-4 for the above calculation

28.20.7.3 Action calculations for STATE forwards

valid for RMO=1 with

p=APS_1C2, t=APS, m=NA[i](part w) mb=NA[i](part b)/1024, NUSE-VSN = q and NUSE=n>m

For SMC=0 and RMO=1 calculate for DIR2=0 all 32 actions in forward direction, if requested; in the case SMC=1 and RMO=1 calculate up to 16 actions 16 to 31 in dependence of the *STATE* input.

Note: All 5 steps of equations DPLL-13 to DPLL-14 are only calculated in emergency mode or for SMC=1 in combination with RMO=1.

28.20.7.3.1 Equation DPLL-13a1 to calculate the time prediction for an action

For DIR2=0 and q>m calculate:

PDT_S[i] = (TSF_S[p+m-n] - TSF_S[p-n] + mb* DT_Sx[t-q+1] * RDT_S[t-q] **(DPLL-13a1)**

with

- DT_Sx[t-q+1]= DT_S[t-q+1] for TS0_HRS=0 **(DPLL-13b2)**

or

- DT_Sx[t-q+1] = DT_S[t-q+1]/8 for TS0_HRS=1 **(DPLL-13b3)**

28.20.7.3.2 Equation DPLL-13a2 to calculate the time prediction for an action

For **SYS=1 and NUSE=2*(SNU+1)**, q>m, SYSF=0, VSN=2*SYN_NS and hence NUSE-VSN = 2*(SNU+1-SYN_NS) equation DPLL-13a1 is equal to

- PDT_S[i] = (TSF_S[p+m] - TSF_S[p]+ mb*DT_Sx[t-q+1]) * RDT_S[t] **(DPLL-13a2)**

with

- DT_Sx[t-q+1] = DT_S[t-q+1] for TS0_HRS=0 **(DPLL-13b2)**

or

- DT_Sx[t-q+1] = DT_S[t-q+1]/8 for TS0_HRS=1 **(DPLL-13b3)**

28.20.7.3.3 Equation DPLL-13b to calculate the time prediction for an action

For NUSE -VTN=q, q (< or =) m and n>1:

- PDT_S[i] = (m+mb)*DT_Sx[t-q+1] * RDT_S[t-q] **(DPLL-13b)**

with

- DT_Sx[t-q+1] = DT_S[t-q+1] for TS0_HRS=0 **(DPLL-13b2)**

or

- DT_Sx[t-q+1] = DT_S[t-q+1]/8 for TS0_HRS=1 **(DPLL-13b3)**

28.20.7.3.4 Equation DPLL-13c to calculate the time prediction for an action

for n=1

- PDT_S[i] = (m+mb)* DT_S_ax * RDT_S[t-1] **(DPLL-13c)**

with

- DT_S_ax = DT_S_ACT for TS0_HRS=0 **(DPLL-6a4a)**

or

- DT_S_ax = DT_S_ACT/8 for TS0_HRS=1 **(DPLL-6a4b)**

28.20.7.3.5 Equation DPLL-14 to calculate the duration value for an action

- DTA[i] = (DT_S_ACT + MEDT_S)* PDT_S[i] **(DPLL-14)**

Use the results of DPLL-7, DPLL-8 and DPLL-9 for the above calculation

28.20.7.4 Action calculations for STATE backwards

valid for RMO=1 with

p=APS_1C2, t=APS, m=NA[i](part w) mb=NA[i](part b)/1024, NUSE-VSN = q and NUSE=n

For SMC=0 and RMO=1 calculate for DIR1=1 all 32 actions in backwards mode for special purposes; in the case SMC=1 and RMO=1 calculate up to 16 actions 16 to 31 in dependence of the *STATE* input.

28.20.7.4.1 Equation DPLL-13a3 to calculate the time prediction for an action

For (DIR2= 1 (SMC=1) or DIR1=1 (SMC=0)) and q>m calculate

- PDT_S[i] = (TSF_S[p-m+n] - TSF_S[p+n] + mb* DT_Sx[t+q-1]) * RDT_S[t+q] **(DPLL-13a3)**

with

- DT_Sx[t+q-1]= DT_S[t+q-1] for TS0_HRS=0 **(DPLL-13b4)**

or

- DT_Sx[t+q-1] = DT_S[t+q-1]/8 for TS0_HRS=1 **(DPLL-13b5)**

28.20.7.4.2 Equation DPLL-13a4 to calculate the time prediction for an action

For SYS=1, **NUSE=2*(SNU+1)**, q>m, SYSF=0, VSN=2*SYN_NS and hence NUSE-VSN = 2*(SNU+1-SYN_NS) equation DPLL-13a3 is equal to

- PDT_S[i] = (TSF_S[p-m] - TSF_S[p]+ mb* DT_Sx[t+q-1]) * RDT_S[t] **(DPLL-13a4)**

with

- DT_Sx[t+q-1]= DT_S[t+q-1] for TS0_HRS=0 **(DPLL-13b4)**

or

- DT_Sx[t+q-1] = DT_S[t+q-1]/8 for TS0_HRS=1 **(DPLL-13b5)**

28.20.7.4.3 Equation DPLL-13b1 to calculate the time prediction for an action

For NUSE-VSN =q, q (< or =) m, NUSE=n and n>1:

- PDT_S[i] = m*DT_Sx[t+q-1] * RDT_S[t+q] **(DPLL-13b1)**

with

- DT_Sx[t+q-1] = DT_S[t+q-1] for TS0_HRS=0 **(DPLL-13b4)**

or

- DT_Sx[t+q-1] = DT_S[t+q-1]/8 for TS0_HRS=1 **(DPLL-13b5)**

28.20.7.4.4 Equation DPLL-13c1 to calculate the time prediction for an action

for n=1

- PDT_S[i] = (m+mb)* DT_S_ax * RDT_S[t+1] **(DPLL-13c1)**

with

- DT_S_ax = DT_S_ACT for TS0_HRS=0 **(DPLL-6a4a)**

or

- DT_S_ax = DT_S_ACT/8 for TS0_HRS=1 **(DPLL-6a4b)**

28.20.7.4.5 Equation DPLL-14 to calculate the duration value until action

DTA[i] = (DT_S_ACT + MEDT_S)* PDT_S[i] (DPLL-14)

Use the results of DPLL-7, DPLL-8 and DPLL-9 for the above calculation

28.20.7.5 Update of RAM in Normal and Emergency Mode

After considering the calculations for up to all 24 actions according to equations (DPLL-11, DPLL-12), only when going back to state 1 or 21 (because of a new TRIGGER or STATE event, that means when no further PMTR values are to be considered) set time stamp values and duration of increments in the RAM.

28.20.7.5.1 Equation DPLL-1a4 to update the time stamp values for TRIGGER

- TSF_T[s]=TS_Tx **(DPLL-1a4)**

using the following equations for the determination of TS_Tx

For TS0_HRT=0:

- TS_Tx=TS_T **(DPLL-1a4w)**
- DT_T_ax = DT_T_ACT **(DPLL-1a4a)**

For TS0_HRT=1:

- TS_Tx(20:0)=TS_T/8 **(DPLL-1a4x)**
- TS_Tx(23:21)=TBU_TS0_T(23:21) **(DPLL-1a4y)**
  for TBU_TS0_T(20:0) > or = TS_Tx(20:0)
- TS_Tx(23:21)=TBU_TS0_T(23:21) -1 **(DPLL-1a4z)**
  for TBU_TS0_T(20:0) < TS_Tx(20:0)
- DT_T_ax = DT_T_ACT/8 **(DPLL-1a4b)**

Note: the combination of values LOW_RES=0 and TS0_HRT=1 is not possible.

Store the time stamp values in the time stamp field according to the address pointer APT_2B=s, but make this update only after the calculations of actions (**Section 28.20.7**) because the old TSF_T[i] values are still needed for these calculations. Please note that the address pointer after a gap is still incremented by SYN_T_OLD in that case (see state machine step 1 in **Section 28.20.8.6**).

28.20.7.5.2 Equation DPLL-1a5-7 to extend the time stamp values for TRIGGER in forward direction

when SYT=1 and SYN_T_OLD=r>1 and DIR1=0

- TSF_T[s-1] = TSF_T[s] -DT_T_ax **(DPLL-1a5)**
- TSF_T[s-2] = TSF_T[s-1] -DT_T_ax **(DPLL-1a6)**

until

- TSF_T[s- r+1] = TSF_T[s- r+2] -DT_T_ax **(DPLL-1a7)**

after the incrementation of the pointer APT_2B by SYN_T_OLD

28.20.7.5.3 Equations DPLL-1a5-7 for backward direction

when SYT=1 and SYN_T_OLD=r>1 and DIR1=0

- TSF_T[s+1] = TSF_T[s] -DT_T_ax **(DPLL-1a5)**
- TSF_T[s+2] = TSF_T[s+1] -DT_T_ax **(DPLL-1a6)**

until

- TSF_T[s- r+1] = TSF_T[s- r+2] -DT_T_ax **(DPLL-1a7)**

after the incrementation of the pointer APT_2B by SYN_T_OLD

28.20.7.5.4 Equations DPLL-1b1 and DPLL-1c1 to update the RAM after calculation

- DT_T[p] = DT_T_ACT **(DPLL-1b1)**

save old reciprocal value from RAM before overwriting:

- RDT_T_FS1= RDT_T[p] **(DPLL-1c1)**

after that store new value in RAM

- RDT_T[p]= RDT_T_ACT **(DPLL-1c2)**

Store increment duration and reciprocal value in RAM region 2 in normal mode after calculation of actions only when a new valid TRIGGER slope is detected and in emergency mode directly after calculation of DT_T_ACT or RDT_T_ACT respectively.

28.20.7.5.5 Equation DPLL-6a4 to update the time stamp values for STATE

- TSF_S[s]=TS_Sx **(DPLL-6a4)**

using the following equations for the determination of TS_Sx

For TS0_HRS=0:

- TS_Sx=TS_S **(DPLL-6a4)**
- DT_S_ax = DT_S_ACT **(DPLL-6a4a)**

For TS0_HRS=1:

- TS_Sx(20:0)=TS_S/8 **(DPLL-6a4x)**
- TS_Sx(23:21)=TBU_TS0_S(23:21) **(DPLL-6a4y)**
  for TBU_TS0_S(20:0) > or = TS_Sx(20:0)
- TS_Sx(23:21)=TBU_TS0_S(23:21) -1 **(DPLL-6a4z)**
  for TBU_TS0_S(20:0) < TS_Sx(20:0)
- DT_S_ax = DT_S_ACT/8 **(DPLL-6a4b)**

Note: the combination of values LOW_RES=0 and TS0_HRS=1 is not possible.

Store the time stamp value in the time stamp field according to the address pointer APS_1C2=s, but make this update only after the calculation of actions (equations DPLL-13a2, **Section 28.20.7.3.2** or DPLL-13a4 **Section 28.20.7.4.2**, if applicable) because the old TSF_S[i] values are still needed for these calculations. Please note, that the address pointer after a gap is still incremented by SYN_S_OLD in that case (see state machine step 21 in **Section 28.20.8.6**).

28.20.7.5.6 Equations DPLL-6a5-7 to extend the time stamp values for STATE

When SYS=1 and SYN_S_OLD=r>1 and DIR2=0 or DIR1=0 respectively calculate

- TSF_S[s-1] = TSF_S[s] - DT_S_ax **(DPLL-6a5)**
- TSF_S[s-2] = TSF_S[s-1] - DT_S_ax **(DPLL-6a6)**

until

- TSF_S[s- r+1] = TSF_S[s- r+2] - DT_S_ax **(DPLL-6a7)**

after incrementation of the pointer APS_2b by SYN_S_OLD

28.20.7.5.7 Equations DPLL-6a5-7 for backward direction

When SYS=1 and SYN_S_OLD=r>1 and DIR2=1 or DIR1=1 respectively calculate

- TSF_S[s+1] = TSF_S[s] - DT_S_ax **(DPLL-6a5)**
- TSF_S[s+2] = TSF_S[s+1] - DT_S_ax **(DPLL-6a6)**

until

- TSF_S[s+r-1] = TSF_S[s+r-2] - DT_S_ax **(DPLL-6a7)**

after the incrementation of the pointer APS_1C2 by SYN_S_OLD

28.20.7.5.8 Equations DPLL-6b1 and DPLL-6c2 to update the RAM after calculation

- DT_S[p] = DT_S_ACT **(DPLL-6b1)**

save old reciprocal value from RAM before overwriting:

- RDT_S_FS1= RDT_S[p] **(DPLL-6c1)**

after that store new value in RAM

- RDT_S[p] = RDT_S_ACT **(DPLL-6c2)**

when a new active STATE slope is detected in emergency mode or in normal mode (SMC=RMO=0) directly after calculation of the values above.

Store increment duration and reciprocal value in RAM region 1c in emergency mode after calculation of actions only when a new active STATE slope is detected and in normal mode directly after calculation of DT_S_ACT or RDT_S_ACT respectively.

28.20.7.6 Time and position stamps for actions in Normal Mode

28.20.7.6.1 Equation DPLL-15 to calculate the action time stamp

- TSAC[i]= DTA[i] - DLA[i] + TS_Tx (for DTA[i] > DLA[i] and DTA[i] - DLA[i] < 0x800000) **(DPLL-15a)**
- TSAC[i]= TS_Tx (for DTA[i] < DLA[i]) **(DPLL-15b)**
- TSAC[i]= 0x7FFFFF + TS_Tx (for DTA[i] > DLA[i] and DTA[i] - DLA[i] > 0x7FFFFF) **(DPLL-15c)**

Note: For TS_Tx see equations (DPLL-1a4 and following and following), ***Section 28.20.7.5.1***

The calculation is done after the calculation of the current expected duration value according to equation DPLL-12 at **Section 28.20.7.2.5**. The time stamp of the action can be calculated as shown above in equation DPLL-15 using the delay value of the action and the current time stamp

28.20.7.6.2 Equations DPLL-17 to calculate the position stamp forwards

for **DIR1**=0 and TS0_HRT=0:

- PSAC[i] = PSA[i] - (DLA[i]*RCDT_TX_NOM)*(MLT+1) **(DPLL-17)**

with

- RCDT_TX_NOM= (1/CDT_TX_NOM) * SYN_T **(DPLL-17a)**

and

- RCDT_TX= 1/CDT_TX **(DPLL-17b)**

for **DIR1**=0 and TS0_HRT=1:

- PSAC[i] = PSA[i] - (8*DLA[i]*RCDT_TX_NOM)*(MLT+1) **(DPLL-17d)**

with

- RCDT_TX_NOM= (1/CDT_TX_NOM) * SYN_T **(DPLL-17a)**

and

- RCDT_TX= 1/CDT_TX **(DPLL-17b)**

replace (MLT+1) in equations (DPLL-17) and (DPLL-17d) by MLS1 for SMC=1
use the calculated value of (DPLL-17b) also for the generation of SUB_INCi and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D_i

The action is to be updated for each new *TRIGGER* event until the calculated time stamp is in the past.
In this case the values of TSAC[i] and PSAC[i] depend on the DPLL_CTRL_11.ACBU signal. When DPLL_CTRL_11.ACBU = ‘0’: Use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value.
When DPLL_CTRL_11.ACBU = ‘1’:
- for ACB[z][1]=’1’:
is used as input signal to control if “action in the past” shall be checked based on position information. If the position has reached “past” use the calculated position stamp of the actual increment as target position value.
- for ACB[z][1]=’0’:
In this case the PSAC[i] is used as calculated by the DPLL.
- for ACB[z][0]=’1’: is used as input signal to control if “action in past” shall be checked based on time information. If the time has reached “past” use the time stamp of the last input event instead of the calculated TSAC[i] value.
- for ACB[z][0]=’0’:
In this case the TSAC[i] is used as calculated by the DPLL.

Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

28.20.7.6.3 Equations DPLL-17 to calculate the position stamp backwards

For **DIR1**=1 and TS0_HRT=0:

- PSAC[i] = PSA[i] + (DLA[i]*RCDT_TX_NOM)*(MLT+1) **(DPLL-17c)**

with

- RCDT_TX_NOM= (1/CDT_TX_NOM) * SYN_T **(DPLL-17a)**

and

- RCDT_TX= 1/CDT_TX **(DPLL-17b)**

For **DIR1**=1 and TS0_HRT=1:

- PSAC[i] = PSA[i] + (8*DLA[i]*RCDT_TX_NOM)*(MLT+1) **(DPLL-17e)**

with

- RCDT_TX_NOM= (1/CDT_TX_NOM) * SYN_T **(DPLL-17a)**

and

- RCDT_TX= 1/CDT_TX **(DPLL-17b)**

replace (MLT+1) in equations (DPLL-17c) and (DPLL-17e) by MLS1 for SMC=1
use the calculated value of (DPLL-17b) also for the generation of SUB_INCi and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D_i

The action is to be updated for each new *TRIGGER* event until the calculated time stamp is in the past.
Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.
In this case the values of TSAC[i] and PSAC[i] depend on the DPLL_CTRL_11.ACBU signal. When DPLL_CTRL_11.ACBU = ‘0’: Use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value.
When DPLL_CTRL_11.ACBU = ‘1’:
- for ACB[z][1]=’1’:
is used as input signal to control if “action in the past” shall be checked based on position information. If the position has reached “past” use the calculated position stamp of the actual increment as target position value.
- for ACB[z][1]=’0’:
In this case the PSAC[i] is used as calculated by the DPLL.
- for ACB[z][0]=’1’: is used as input signal to control if “action in past” shall be checked based on time information. If the time has reached “past” use the time stamp of the last input event instead of the calculated TSAC[i] value.
- for ACB[z][0]=’0’:
In this case the TSAC[i] is used as calculated by the DPLL.

Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

28.20.7.7 The use of the RAM

The RAM is used to store the data of the last FULL_SCALE period. The use of single port RAMs is recommended. The data width of the RAM is usual 3 bytes, but could be extended to 4 bytes in future applications. There are 3 different RAMs, each with separate access ports. The RAM 1a is used to store the position minus time requests, got from the ARU. No CPU access is possible to this RAM during operation (when the DPLL is enabled).

Ram 1b is used for configuration parameters and variables needed for calculations. Within RAM 1c the values of the *STATE* events are stored. RAM 1b and RAM 1c do have a common access port and are also marked as RAM 1bc in order to clarify this fact.

RAM 2 is used for values of the *TRIGGER* events.

Because of the access of the DPLL internal state machine at the one side and the CPU at the other side the access priority has to be controlled for both RAMs 1bc and 2. The access priority is defined as stated below. The CPU access procedure via AE-interface goes in a wait state (waiting for data valid) while it needs a colliding RAM access during serving a corresponding state machine RAM access. In order not to provoke unexpected behavior of the algorithms the writing of the CPU to the RAM regions 1b, 1c or 2 will be monitored and results in interrupt requests when enabled.

CPU access is specified at follows:

1. CPU has highest priority for a single read/write access. The DPLL algorithm is stalled during external bus RAM accesses.
2. After serving the CPU access to the RAM the DPLL gets the highest RAM access priority for 8 clock cycles. Afterwards continue with 1.

The RAM address space has to be implemented in the address space of the CPU.

28.20.7.8 Time and position stamps for actions in Emergency Mode

28.20.7.8.1 Equation DPLL-18 to calculate the action time stamp

- TSAC[i]= DTA[i] - DLA[i] + TS_Sx (for DTA[i] > DLA[i] and DTA[i] - DLA[i] < 0x800000) **(DPLL-18a)**
- TSAC[i]= TS_Sx (for DTA[i] < DLA[i]) **(DPLL-18b)**
- TSAC[i]= 0x7FFFFF + TS_Sx (for DTA[i] > DLA[i] and DTA[i] - DLA[i] > 0x7FFFFF) **(DPLL-18c)**

Note: For TS_Sx see equations (DPLL-6a4 and following), ***Section 28.20.7.5.5***

The calculation is done after the calculation of the current expected duration value according to equation DPLL-14 at **Section 28.20.7.3.5**. The time stamp of the action can be calculated as shown in equation DPLL-18 using the delay value of the action and the current time stamp.

28.20.7.8.2 Equations DPLL-20 to calculate the position stamp forwards

for **DIR2**=0 or DIR1=0 respectively and TS0_HRS=0:

- PSAC[i] = PSA[i] - (DLA[i]*RCDT_SX_NOM)*MLS1 **(DPLL-20)**

with

- RCDT_SX_NOM= (1/CDT_SX_NOM) * SYN_S **(DPLL-20a)**

and

- RCDT_SX= 1/CDT_SX **(DPLL-20b)**

for **DIR2**=0 or DIR1=0 respectively and TS0_HRS=1:

- PSAC[i] = PSA[i] - (8*DLA[i]*RCDT_SX_NOM)*MLS1 **(DPLL-20d)**

with

- RCDT_SX_NOM= (1/CDT_SX_NOM) * SYN_S **(DPLL-20a)**

and

- RCDT_SX= 1/CDT_SX **(DPLL-20b)**

replace MLS1 in equations (DPLL-20) and (DPLL-20d) by MLS2 for (SMC=1 and RMO=1)
use the calculated value of (DPLL-17b) also for the generation of SUB_INCi and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_Di.

The action is to be updated for each new *STATE* event until the calculated time stamp is in the past.
In this case the values of TSAC[i] and PSAC[i] depend on the DPLL_CTRL_11.ACBU signal.

When DPLL_CTRL_11.ACBU = ‘0’: Use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value.
When DPLL_CTRL_11.ACBU = ‘1’:
- for ACB[z][1]=’1’:
is used as input signal to control if “action in the past” shall be checked based on position information. If the position has reached “past” use the calculated position stamp of the actual increment as target position value.
- for ACB[z][1]=’0’:
In this case the PSAC[i] is used as calculated by the DPLL.
- for ACB[z][0]=’1’: is used as input signal to control if “action in past” shall be checked based on time information. If the time has reached “past” use the time stamp of the last input event instead of the calculated TSAC[i] value.
- for ACB[z][0]=’0’:
In this case the TSAC[i] is used as calculated by the DPLL.
the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value.
Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

se the calculated value of (DPLL-17b) also for the generation of SUB_INCi and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D_i.

The action is to be updated for each new *STATE* event until the calculated time stamp is in the past.
In this case the values of TSAC[i] and PSAC[i] depend on the DPLL_CTRL_11.ACBU signal.

When DPLL_CTRL_11.ACBU = ‘0’: Use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value.
When DPLL_CTRL_11.ACBU = ‘1’:
- for ACB[z][1]=’1’:
is used as input signal to control if “action in the past” shall be checked based on position information. If the position has reached “past” use the calculated position stamp of the actual increment as target position value.
- for ACB[z][1]=’0’:
In this case the PSAC[i] is used as calculated by the DPLL.
- for ACB[z][0]=’1’: is used as input signal to control if “action in past” shall be checked based on time information. If the time has reached “past” use the time stamp of the last input event instead of the calculated TSAC[i] value.
- for ACB[z][0]=’0’:
In this case the TSAC[i] is used as calculated by the DPLL.
the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value.
Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

28.20.7.8.3 Equations DPLL-20 to calculate the position stamp backwards

For **DIR2**=1 or DIR1=1 respectively and TS0_HRS=0:

- PSAC[i] = PSA[i] + (DLA[i]*RCDT_SX_NOM)*MLS1 **(DPLL-20c)**

with

- RCDT_SX_NOM= (1/CDT_SX_NOM) * SYN_S **(DPLL-20a)**

and

- RCDT_SX= 1/CDT_SX **(DPLL-20b)**

For **DIR2**=1 or DIR1=1 respectively and TS0_HRS=1:

- PSAC[i] = PSA[i] + (8*DLA[i]*RCDT_SX_NOM)*MLS1 **(DPLL-20e)**

with

- RCDT_SX_NOM= (1/CDT_SX_NOM) * SYN_S **(DPLL-20a)**

and

- RCDT_SX= 1/CDT_SX **(DPLL-20b)**

replace MLS1 in equations (DPLL-20c) and (DPLL-20e) by MLS2 for (SMC=1 and RMO=1)
use the calculated value of (DPLL-20b) also for the generation of SUB_INCi and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D.

The action is to be updated for each new *STATE* event until the event is in the past. In this case use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value. Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

28.20.8 Signal processing

28.20.8.1 Time stamp processing

Signal processing does mean the computation of the time stamps in order to calculate at which time the outputs have to appear. For such purposes the time stamp values have to be stored in the RAM and by calculating the difference between old and new values the duration of the last time interval is determined simply. This difference should be also stored in the RAM in order to see the changes between the intervals by changing the conditions and the speed of the observed process.

28.20.8.2 Count and compare unit

The count and compare unit processes all input signals taking into account the configuration values. It uses a state machine and provides the output signals as described above.

28.20.8.3 Sub pulse generation for SMC=0

28.20.8.3.1 Adder for generation of SUB_INCx by the carry c_{out}

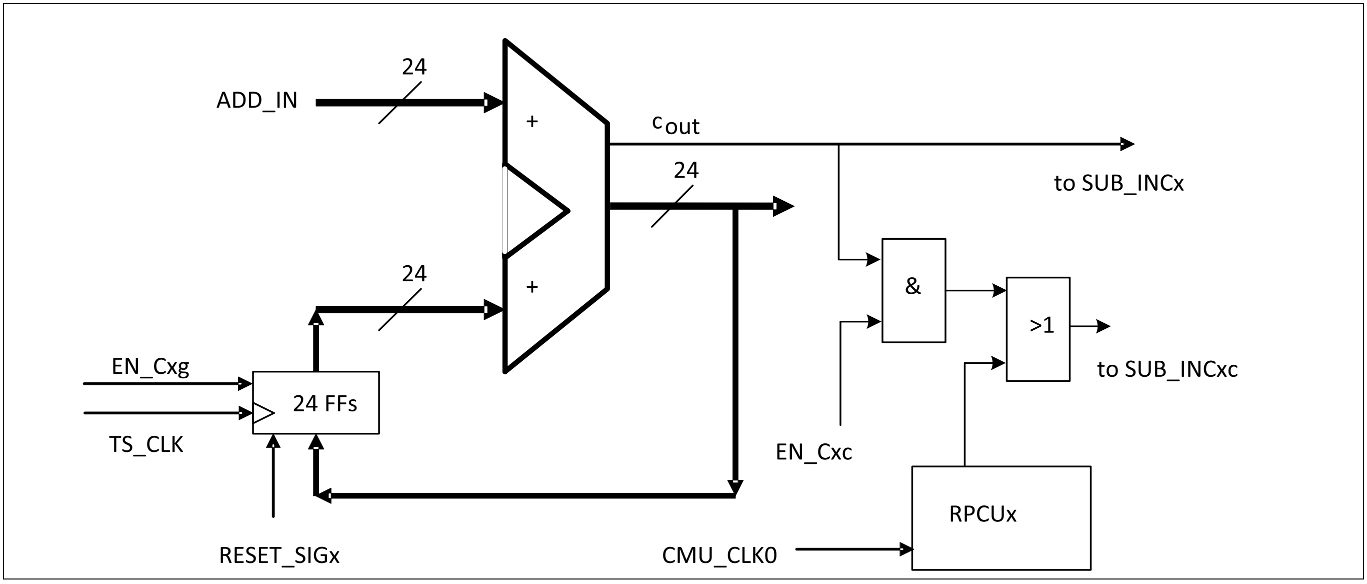


Figure 128 Adder for generation of SUB_INCx by the carry cout

Note: The SUB_INC generation by the circuit above has the advantage, that the resolution for higher speed values is better as for a simple down counter.

After RESET and after EN_Cxg=0 the flip-flops (FF s) should have a zero value. EN_Cxg has to be zero until reliable ADD_IN values are available and the pulse generation starts. This is controlled by the configuration bits SGE1,2 in the DPLL_CONTROL_1 register. The calculated values for the increment prediction using equations DPLL-2c **Chapter 28.20.6.2.7**, DPLL-2c1 **Chapter 28.20.6.4.4**, DPLL-7c **Chapter 28.20.6.3.7** or DPLL-7c1 **Chapter 28.20.6.4.4** respectively are valid only when at least NUTE>1 TRIGGER values or at least NUSE>1 STATE values are available. For NUTE =1 or NUSE=1 respectively the equations DPLL-25 **Chapter 28.20.8.3.4** and DPLL-26 **Chapter 28.20.8.3.6** use the actual increment value subtracted by the weighted average error.

The generation of SUB_INC1 pulses depends on the configuration of the DPLL. In automatic end mode the counter INC_CNT1 resets the enable signal EN_C1 when the number of pulses desired is reached. In this case only the uncompensated output SUB_INC1 remains active in order to provide pulses for the input filter unit. In the case of acceleration missing pulses can be determined at the next TRIGGER/STATE event in normal/emergency mode easily. For the correction strategy COA = 0 those missing pulses are sent out with CMU_CLK0 frequency as soon they are determined. During this time period the EN_Cxg remains cleared. After calculation or providing of a new ADD_IN value the FFs are enabled by EN_Cxg. In this way no pulse is lost. The new pulses are sent out afterwards, when INC_CNT1 is set to the desired value, maybe by adding MLT+1 or MLS1 respectively for the new TRIGGER/STATE event.

Because the used DIV procedure of the algorithms results only in integer values, a systematic failure could appear. The pulse generation at SUB_INC1 will stop in automatic end mode when the INC_CNT1 register reaches zero or all remaining pulses at a new increment will be considered in the next calculation. In this way the loss of pulses can be avoided. When a new TRIGGER/STATE appears the value of SYN_T*(MLT+1) or SYN_S*MLS1 respectively is added to INC_CNT1, when SGE1=1. Therefore for FULL_SCALE 2*(TNU+1)*(MLT+1) pulses SUB_INC1 generated, when INC_CNT1 reaches the zero value. The generation of SUB_INC1 pulses has to be done as fast as possible. The calculations for the ADD_IN value must be done first. Therefore all values needed for calculation are to be fetched in a forecast.

28.20.8.3.2 Equation DPLL-21 to calculate the number of pulses to be sent in normal mode using the automatic end mode condition

For RMO=0, SMC=0 and DMO=0

- NMB_T = (MLT+1)*SYN_T + MP + PD_store + MPVAL1 **(DPLL-21)**

with

PD_store = ADT_T[12:0], prefetched during last increment
SYN_T = ADT_T[18:16], prefetched during last increment
MPVAL1 = pulse correction value for PCM1_SHADOW_TRIGGER=1

while the value for PD_store is zero for AMT=0
and
the value of MP is zero for COA=0

In order to get a higher resolution for higher speed a generator for the sub-pulses is chosen using an adder. All missing pulses MP are considered using equation DPLL-21 and are determined by counting the number of pulses of the last increment. The value SYN_T is stored from the last increment using NT of the ADT_T[i] value at RAM region 2c.

28.20.8.3.3 Equations DPLL-22-24 to calculate the number of pulses to be sent in emergency mode using the automatic end mode condition for SMC=0

For RMO=1, SMC=0 and DMO=0; the value for PD_S_store is zero for AMS=0

- NMB_S = (MLS1 + PD_S_store) *SYN_S+ MP **(DPLL-22)**

with

- MLS1= (MLT+1) * (TNU+1) / (SNU+1) **(DPLL-23)**

PD_S_store = ADT_S[15:0], prefetched during last increment
SYN_S = ADT_S[21:16], prefetched during last increment
MPVAL1 = pulse correction value for PCM1_SHADOW_STATE=1

while the value for PD_S_store is zero for AMS=0
and
the value of MP is zero for COA=0

Please note, that these calculations above in equations DPLL-21 and DPLL-22 are only valid for an automatic end mode (DMO = 0).
For calculation of the number of generated pulses a value of 0.5 is added as shown in equations DPLL-25 or DPLL-26 respectively in order to compensate rounding down errors at the succeeding arithmetic operations. Because in automatic end mode the number of pulses is limited by **INC_CNT1** it is guaranteed, that not more pulses as needed are generated and in the same way missing pulses are caught up for the next increment.

28.20.8.3.4 Equation DPLL-25 to calculate ADD_IN in normal mode for SMC=0

In normal mode (for RMO=0) calculate in the case LOW_RES=TS0_HRT

- ADD_IN_CALN= (NMB_T+0.5) *RCDT_TX **(DPLL-25)**

with

RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b, see **Section 28.20.7.6.3**

In normal mode (for RMO=0) calculate in the case LOW_RES=1 and TS0_HRT=0

- ADD_IN_CALN= (NMB_T+0.5) *(RCDT_TX /8) **(DPLL-25a)**

with

RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b, see **Section 28.20.7.6.3**

For RMO=0 and SMC=0:

- ADD_IN_CAL1 = ADD_IN_CALN **(DPLL-25b)**

LOW_RES=0 and TS0_HRT=1 is not possible. For such a configuration the RCT bit in the DPLL_STATUS register is set together with the ERR bit.

In the automatic end mode (DMO=0) missing pulses should be sent to the input RPCUx (rapid pulse catch up on) in **Figure 128**, to be caught up on with CMU_CLK0 (for COA=0).

When normal and rapid pulses are generated simultaneously, the SUB_INCx frequency is doubled at this moment in order to count two pulses at the TBU_CHx_BASE register. In order to make the frequency doubling possible, the CMU_CLK0 should be having a frequency which does not exceed half the frequency of TS_CLK. In addition the ADD_IN value should never exceed the value 0x800000. This limitation is only necessary for DMO=0 and COA=0 (see DPLL_CTRL_1 register).

For the normal mode replace ADD_IN of the ADDER (see **Figure 128**) by ADD_IN_CAL1 (when calculated, DLM=0) or ADD_IN_LD1 (when provided by the CPU, DLM=1).

The sub-pulse generation in this case is done by the following calculations using a 24 bit adder with a carry out c_{out} and the following inputs:

- ADD_IN
- the second input is the output of the adder, stored one time stamp clock before

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

28.20.8.3.5 Enabling of the compensated output for pulses

The c_{out} of the adder influences directly the *SUB_INC1* output of the DPLL (see **Figure 128**). The compensated output SUB_INCxc is in automatic end mode only enabled by EN_Cxc when **INC_CNTx** >0.

28.20.8.3.6 Equation DPLL-26 to calculate ADD_IN in emergency mode for SMC=0

In emergency mode (RMO=1) calculate in the case LOW_RES=TS0_HRS

- ADD_IN_CALE= (NMB_S+0.5)* RCDT_SX **(DPLL-26)**

while

RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b, see **Section 28.20.7.8.2**

In emergency mode (RMO=1) calculate in the case LOW_RES=1 and TS0_HRS=0

- ADD_IN_CALE= (NMB_S+0.5)* RCDT_SX /8 **(DPLL-26a)**

while

RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b, see **Section 28.20.7.8.2**

For RMO=1 and SMC=0:

- ADD_IN_CAL1 = ADD_IN_CALE **(DPLL-26b)**

LOW_RES=0 and TS0_HRS=1 is not possible. For such a configuration the RCS bit in the DPLL_STATUS register is set together with the ERR bit.

In the automatic end mode (DMO=0) missing pulses should be sent to the input RPCUx (rapid pulse catch up on) in **Figure 128**, to be caught up on with CMU_CLK0 (for COA=0).

When normal and rapid pulses are generated simultaneously, the SUB_INCx frequency is doubled at this moment in order to count two pulses at the TBU_CHx_BASE register. In order to make the frequency doubling possible, the CMU_CLK0 should be having a frequency which does not exceed half the frequency of the system clock. In addition the ADD_IN value should never exceed the value 0x800000 when the TS_CLK frequency exceeds half the frequency of the system clock. This limitation is only necessary for DMO=0 and COA=0 (see DPLL_CTRL_1 register).

For the emergency mode replace ADD_IN of the ADDER (see **Figure 128**) by ADD_IN_CAL1 (when calculated, DLM=0) or ADD_IN_LD1 (when provided by the CPU, DLM=1).

The sub-pulse generation in this case is done by the following calculations using a 24 bit adder with a carry out c_{out} and the following inputs:

- ADD_IN
- the second input is the output of the adder, stored one time stamp clock before

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

28.20.8.4 Sub pulse generation for SMC=1

28.20.8.4.1 Necessity of two pulse generators

The Adder of picture **Figure 128** must be implemented twice in the case of SMC=1: one for SUB_INC1 controlled by the *TRIGGER* input and (while RMO=1) one for SUB_INC2, controlled by the *STATE* input. In the case described in the chapter above for SMC=0 only one Adder is used to generate SUB_INC1 controlled by the *TRIGGER* in normal mode or by *STATE* in emergency mode.

28.20.8.4.2 Equation DPLL-27 to calculate the number of pulses to be sent for the first device using the automatic end mode condition

For SMC=1 and DMO=0

- NMB_T = (MLS1 + PD_store) *SYN_T + MP + MPVAL1 **(DPLL-27)**

with
PD_store = ADT_T[12:0], prefetched during last increment
SYN_T = ADT_T[18:16], prefetched during last increment
MPVAL1 = pulse correction value for PCM1_SHADOW_TRIGGER=1

while the value for PD_store is zero for AMT=0
and
for COA=0 use zero instead of the value of MP

28.20.8.4.3 Equation DPLL-28 to calculate the number of pulses to be sent for the second device using the automatic end mode condition

for RMO=1, SMC=1 and DMO=0

- NMB_S = MLS2 *SYN_S+ MP + PD_S_store + MPVAL2 **(DPLL-28)**

with
PD_S_store = ADT_S[15:0], prefetched during last increment
SYN_S = ADT_S[21:16], prefetched during last increment
MPVAL2 = pulse correction value for PCM2_SHADOW_STATE=1

while the value for PD_S_store is zero for AMS=0
and
for COA=0 use zero instead of the value of MP

Please note, that these calculations above in equations DPLL-27 and DPLL-28 are only valid for an automatic end mode (DMO = 0). In addition the number of generated pulses is added by 0.5 as shown in equations DPLL-30 or DPLL-31 respectively in order to compensate rounding down errors at the succeeding division operation. Because in automatic end mode the number of pulses is limited by **INC_CNTx** it is guaranteed, that not more pulses as needed are generated and in the same way missing pulses are made up for the next increment.

28.20.8.4.4 Equation DPLL-30 to calculate ADD_IN for the first device for SMC=1

The sub-pulse generation in this case is done by the following calculations using a 24 bit adder with a carry out c_{out} and the following inputs:

- ADD_IN
- the second input is the (delayed) output of the adder, stored with each time stamp clock.

Replace ADD_IN by ADD_IN_CAL1 (when calculated, DLM1=0) or ADD_IN_LD1 (when provided by the CPU, DLM1=1) respectively while:

For SMC=1 and LOW_RES=TS0_HRT

- ADD_IN_CAL1= (NMB_T+0.5) * RCDT_TX **(DPLL-30)**

When RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b, see **Section 28.20.7.6.3**

For SMC=1,LOW_RES= 1 and TS0_HRT=0

- ADD_IN_CAL1= (NMB_T+0.5) * (RCDT_TX /8) **(DPLL-30a)**

When RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b, see **Section 28.20.7.6.3**

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

ADD_IN_CAL1 is a 24 bit integer value. The CDT_TX is the expected duration of current *TRIGGER* increment.

The c_{out} of the adder influences directly the *SUB_INC1* output of the DPLL (see **Figure 128**). The SUB_INC1 output is in automatic end mode only enabled by EN_C1 when **INC_CNT1** >0.

28.20.8.4.5 Equation DPLL-31 to calculate ADD_IN for the second device for SMC=1

Replace ADD_IN by ADD_IN_CAL2 (when calculated, DLM2=0) or ADD_IN_LD2 (when provided by the CPU, DLM2=1) respectively while:

for SMC=1, RMO=1 and LOW_RES=TS0_HRS:

- ADD_IN_CAL2= (NMB_S+0.5)* RCDT_SX **(DPLL-31)**

When RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b, see **Section 28.20.7.8.2**

for SMC=1, RMO=1, LOW_RES=1 and TS0_HRS=0:

- ADD_IN_CAL2= (NMB_S+0.5)* (RCDT_SX /8) **(DPLL-31a)**

When RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b, see **Section 28.20.7.8.2**

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

The c_{out} of the adder2 influences directly the *SUB_INC2* output of the DPLL (see **Figure 128**).

The SUB_INC2 output is in automatic end mode only enabled by EN_C2 when **INC_CNT2** >0.

Note: Please note, that after RESET and after EN_Cxc=0 (after stopping in automatic end mode) the flip-flops (FFs) have a zero value and also EN_Cxg has to be zero until reliable ADD_IN values are available and the pulse generation starts. The calculated values for the increment prediction using equations DPLL-2c **Section 28.20.6.2.7**, DPLL-2c1 **Section 28.20.6.4.4**, DPLL-7c **Section 28.20.6.3.7** or DPLL-7c1

Section 28.20.6.4.4 respectively are valid only when NUTE>1 or NUSE>1 respectively. For NUTE=1 or NUSE=1 respectively the equations DPLL-30 (see **Section 28.20.8.4.4**) and DPLL-31 (see **Section 28.20.8.4.5**) use the actual increment value subtracted by the weighted average error.

The generation of *SUB_INCx* pulses depends on the configuration of the DPLL.In automatic end mode the counter **INC_CNTx** resets the enable signal EN_Cxcu when the number of pulses desired is reached. In this case only the uncompensated outputs SUB_INCx remain active in order to provide pulses for the input filter units. A new *TRIGGER* or *STATE* input respectively can reset the FFs and also ADD_IN, especially when EN_Cxc was zero before. In the case of acceleration missing pulses can be determined at the next *TRIGGER/STATE* event easily. For the correction strategy COA = 0 those missing pulses are sent out with CMU_CLK0 frequency as soon they are determined. After that the pulse counter **INC_CNTx** should be always zero and the new pulses are sent out afterwards, when **INC_CNTx** is set to the desired value by adding MLS1 or MLS2 for the new *TRIGGER* or *STATE* event respectively.

Because the used DIV procedure of the algorithms results only in integer values, a systematic failure could appear. The pulse generation will stop when the **INC_CNTx** register reaches zero or all remaining pulses at a new increment will be considered in the next calculation. In this way the loss of pulses can be avoided.

When a new *TRIGGER* appears the value of SYN_T*MLS1 is added to **INC_CNT1**. Therefore for FULL_SCALE 2*(TNU+1)*MLS1 pulses *SUB_INC1* generated, when **INC_CNT1** reaches the zero value. The generation of SUB_INC1 pulses has to be done as fast as possible.

When a new *STATE* appears the value of SYN_S*MLS2 is added to **INC_CNT2**. Therefore for FULL_SCALE 2*(SNU+1)*MLS2 pulses *SUB_INC2* generated, when **INC_CNT2** reaches the zero value. The generation of SUB_INC2 pulses has to be done as fast as possible.

28.20.8.5 Calculation of the Accurate Position Values

All appearing *TRIGGER* and *STATE* signals do have a time stamp and a position stamp assigned after the input filter procedure. For the calculation of the exact time stamp the filter values are considered in the calculations of equations DPLL-1a **Section 28.20.6.2.1** or DPLL-6a **Section 28.20.6.3.1** respectively. A corresponding calculation is to be performed for the calculation of position values. The PSTC and PSSC values can be corrected by the CPU, when needed.

The PSTC and PSSC values can be corrected by the CPU, when needed.
After reset, while FTD=0 and no active TRIGGER slope is detected:

- PSTC = 0 **(DPLL-32a)**

Calculate the new Position value for each valid TRIGGER event:

- PSTC= PSTC_old + NMB_T_TAR_OLD **(DPLL-32b)**

when FTD=1 and SGE1=1
with
PSTC_old is the last PSTC value and
NMB_T_old is the number of pulses which are calculated
and provided for sending out in the last increment.

After reset, while FSD=0 and no active STATE slope is detected:

- PSSC= 0 **(DPLL-33a)**

Calculate the new Position value for each STATE event:

• PSSC= PSSC_old + NMB_S_TAR_OLD	**(DPLL-33b)**
when FSD=1 and SGE1=1 (SMC=0) or SGE2=1 (SMC=1 respectively with PSSC_old is the last PSSC value and NMB_S_old is the number of pulses which are calculated and provided for sending out in the last increment.	

28.20.8.6 Scheduling of the Calculation

After enabling the DPLL with each active *TRIGGER* or *STATE* event respectively a cycle of operations is performed to calculate all the results shown in detail in the table below **Section 28.20.2**. A state machine controls this procedure and consists of two parts, the first is triggered by an active slope of the signal *TRIGGER*, begins at step 1 and ends at step 20 (in normal mode and for SMC=1). The second state machine is controlled by an active slope of the signal *STATE*, begins at step 21 and ends at step 40 (in emergency mode and also for SMC=RMO=1). Depending on the mode used all 20 steps are executed or already after 2 steps the jump into the initial state is performed, as shown in the state machine descriptions below. For each new extended cycle (without this jump) all prediction values for actions in the case SMC=0 are calculated once more (with maybe improved accuracy because of better parameters) and all pending decisions are made using these new values when transmitted to the decision device.

In **Section 28.20.8.6.6** the steps of the state machine are described. Please note, that the elaboration of the steps depends on the configuration bits described in the comments. The steps 4 to 17 are only calculated in normal mode (in the state machine explanation below marked yellow in **Section 28.20.2**), but steps 24 to 37 are only calculated in emergency mode (in the state machine explanation below marked cyan in **Section 28.20.2**) when SMC=0.

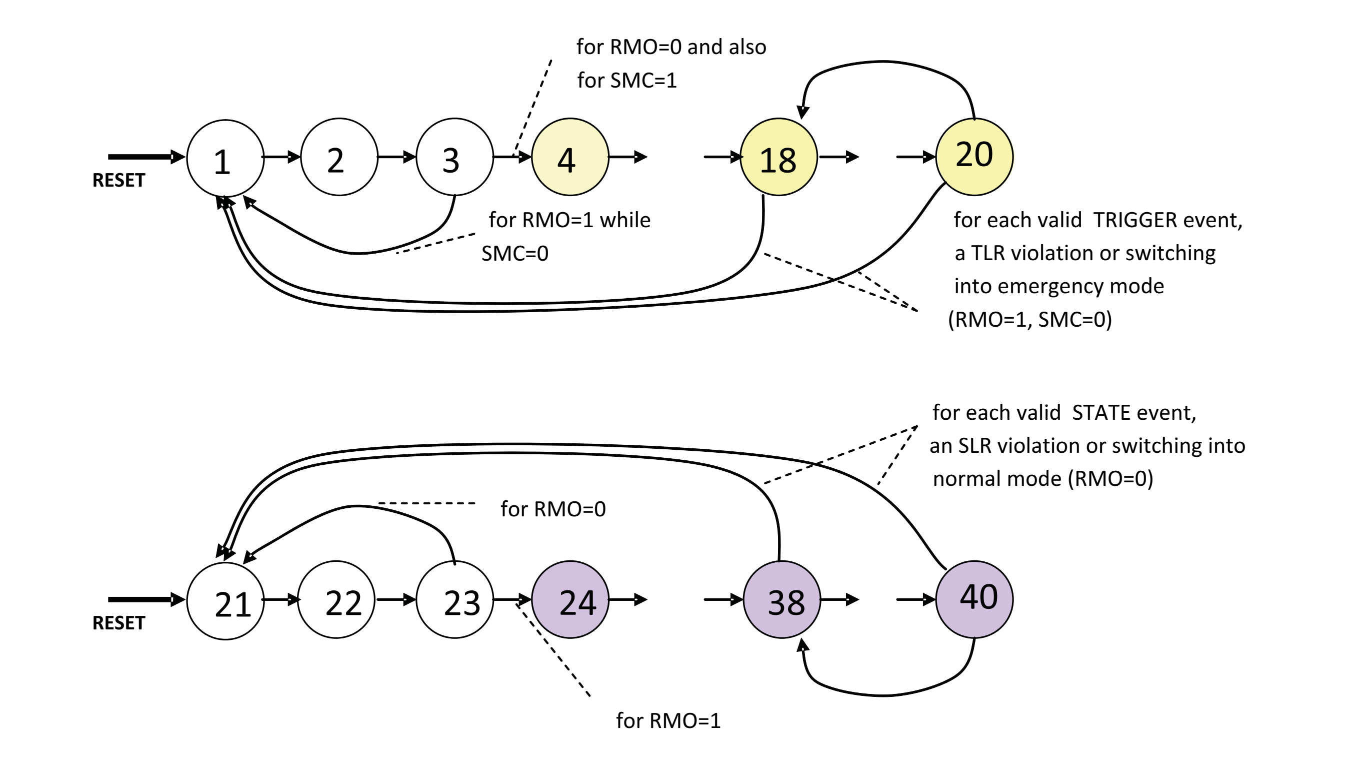


Figure 129 State machine partitioning for normal and emergency mode.

28.20.8.6.1 Synchronization description

TRIGGER

The APT (address pointer for duration and reciprocal duration values of *TRIGGER* increments) is initially set to zero and incremented with each active *TRIGGER* event. Therefore data are stored in the RAM beginning from the first available value. The actual duration of the last increment is stored at DT_T_ACT. For the prediction of the next increment it is assumed, that the same value is valid as long as NUTE is one.

A missing *TRIGGER* is assumed, when at least after TOV* DT_T_ACT no active *TRIGGER* event appears.

The data of equations DPLL-1b1 and DPLL-1c2 **Section 28.20.7.5.4** are written in the corresponding RAM regions and APT is incremented accordingly up to 2*TNU-2*SYN_NT+1.

The APT_2B (address pointer for the time stamp field of *TRIGGER*) is initially set to zero and incremented with each active *TRIGGER* event. When no gap is detected because of the incomplete synchronization process at the beginning, for all *TRIGGER* events the time stamp values are written in the RAM up to 2*(TNU+1) entries, although only 2*(TNU+1-SYN_NT) events in FULL_SCALE appear. When the current position is detected, the synchronization procedure can be performed as described below:

Before the CPU sets the APT_2C address pointer in order to synchronize to the profile, it writes the corresponding increment value for the necessary extension of the RAM region 2b value APT_2B_EXT into the register APT_2B_sync and sets the status bit APT_2C_status. This value can be e.g. 2*SYN_NT, when all gaps in FULL_SCALE already passed the input data stream of *TRIGGER*, or less then this value, when up to now e.g. only a single gap is to be considered in the data stream stored already in the RAM region 2b. The number of virtual increments to be considered depends on the number of inputs already got. After writing APT_2C by the CPU, with the next *TRIGGER* event the APT_2B address pointer is incremented (as usual) and then the additional offset value APT_2B_EXT is added to it once (while APT_2B_STATUS=1 and for forward direction). For that reason the APT_2B_STATUS bit is reset after it. The old APT_2B value before adding the offset is stored in the APT_2B_OLD register as information for the CPU where to start the extension procedure. In the following the CPU fills in the time stamp field around the APT_2B_OLD position taking into account the corresponding number of virtual entries stored in the APT_2B_EXT value and the corresponding NT values in the profile. The extension procedure ends when all gaps considered in the APT_2B_EXT value are treated once. In the consequence all storage locations of RAM region 2b up to now do have the corresponding entries. Future gaps are treated by the DPLL. For a backward direction the APT_2C_ext value is subtracted accordingly.

When the CPU writes the APT_2C address pointer the SYT bit is set simultaneously. For SYT=1 in normal mode (SMC=0) the LOCK1 bit is set with the system clock, when the right number of increments between two synchronization gaps is detected by the DPLL. An unexpected missing *TRIGGER* or an additional *TRIGGER* between two synchronization gaps does reset the LOCK1 bit in normal mode. In that case the CPU must correct the SUB_INC pulse number and maybe correct the APT_2C pointer. For this purpose the LL1I interrupt can be used.

When SYT is set the calculations of equations DPLL-1 to DPLL-5 are performed accordingly and the values are stored in (and distributed to) the right RAM positions.This includes the multiple time stamp storage by the DPLL for a gap according to equations DPLL-1a5 to 7 forwards **Section 28.20.7.5.2** or backwards **Section 28.20.7.5.3**. The APT_2B pointer is for that reason incremented or decremented before this operation considering the virtual increments in addition.

Please note, that for the APT and APT_2C pointers the gap is considered as a single increment.

STATE:

The APS (address pointer for duration and reciprocal duration values of *STATE*) is initially set to zero and incremented with each active *STATE* event. Therefore data are stored in the RAM field beginning at the first location. The actual duration of the last increment is stored at DT_S_ACT. For the prediction of the next increment it is assumed, that the same value is valid as long as NUSE is one.

A missing *STATE* is assumed, when at least after SOV* DT_S_ACT no active *STATE* event appears.

The data of equations DPLL-6b1 and DPLL-6c2 **Section 28.20.7.5.8** is written in the corresponding RAM regions and APS is incremented accordingly up to 2*SNU-2*SYN_NS+1 (for SYSF=0).

The APS_1C2 (address pointer for the time stamp field of *STATE*) is initially set to zero and incremented with each active *STATE* event. When no gap is detected because of the incomplete synchronization process at the beginning, for all *STATE* events the time stamp values are written in the RAM up to 2*(SNU+1) entries, although (e.g. for SYSF=0) only 2*(SNU+1-SYN_NS) events in FULL_SCALE appear. When the current position is detected, the synchronization procedure can be performed as described below:

Before the CPU sets the APS_1C3 address pointer in order to synchronize to the profile, it writes the corresponding increment value APS_1C2_EXT for the necessary extension of the RAM region 1c2 into the register DPLL_APS_SYNC and sets the APS_1C2_STATUS bit there. This value can be e.g. 2*SYN_NS (for SYSF=0) or SYN_NS (for SYSF=1), when all gaps in FULL_SCALE already passed the input data stream of *STATE*. Also less then this value can be considered, when up to now only a single gap is to be considered in the data stream stored already in the RAM region 1c2. The number of increments to be considered depends on the number of inputs already got. After writing APS_1C3 by the CPU, with the next active *STATE* slope the APS_1C2 address pointer is incremented (as usual) and then the additional offset value APS_1C2_EXT is added to it once (while APS_1C2_STATUS=1 and forward direction). For that reason the APS_1C2_STATUS bit is reset after it. The old APS_1C2 value is stored in the APS_1C2_OLD register as information for the CPU where to start the extension procedure. In the following the CPU extends the time stamp field beginning from the APS_1C2_OLD position taking into account the corresponding number of virtual entries according to the APS_1C2_EXT value and also the correspondent NS values in the profile. The extension procedure ends when all gaps considered in the APS_1C2_EXT value are treated once. In the consequence all storage locations of RAM region 1c2 up to now do have the corresponding entries. Future gaps are treated by the DPLL.

For a backward direction the APS_1C2_EXT value is subtracted accordingly.

When the CPU writes the APS_1C3 address pointer the SYS bit is set simultaneously. For SYS=1 in emergency mode (SMC=0 and DMO=1) the LOCK1 bit is set with the system clock, when the right number of increments between two synchronization gaps is detected by the DPLL. An unexpected missing *STATE* or an additional *STATE* between two synchronization gaps does reset the LOCK1 bit in emergency mode. In that case the CPU must correct the SUB_INC1 pulse number and maybe correct the APS_1C3 pointer. For this purpose the LL1I interrupt can be used.

When SYS is set the calculations of equations DPLL-5 to DPLL-10 are performed accordingly and the values are stored in (and distributed to) the right RAM positions.This includes the multiple time stamp storage by the DPLL for a gap according to equations DPLL-6a5 to 7 forwards **Section 28.20.7.5.6** or backwards **Section 28.20.7.5.7**. The APS_1C2 pointer is for that reason incremented or decremented before this operation considering the virtual increments in addition.Please note, that for the APS and APS_2c pointers the gap is considered as a single increment.

SMC=1:

For SMC=1 it is assumed, that the starting position is known by measuring the characteristic of the device. In this way the APT and APT_2C as well the APS and APS_1C3 values are set properly, maybe with an unknown repetition rate. When no gap is to be considered for *TRIGGER* or *STATE* signals the APT_2B and APS_1C2 address pointers are set equal to APT or APS respectively. It is assumed, that all missing *TRIGGER*s and missing *STATE*s can be also considered from the beginning, when a valid profile with the corresponding adapted values is written in the RAM regions 1c3 and 2c respectively. In that case the TSF_T[i] and TSF_S[i] must be extended by the DPLL according to the profile. Thus the SYT and SYS bits could be set from the beginning and the LOCK1 and LOCK2 bits are set after recognition of the corresponding gaps accordingly. When no gap exists (SYN_NT=0 or SYN_NS=0), the LOCK bits are set immediately. The CPU can correct the APT_2C and APS_1C3 pointer according to the recognized repetition rate later once more without the loss of Lock1,2.

28.20.8.6.2 Operation for direction change in normal and emergency mode (SMC=0)

When for SMC=0 in normal mode a backwards condition is detected for the TRIGGER input signal (e.g. when THMI is not violated), the LOCK1 bit in the DPLL_STATUS register is reset, the NUTE value in NUTC register is set to 1 (the same for NUSE in NUSC). The address pointers APT_2C as described below (and after that decremented for each following active slope of *TRIGGER* as long as the DIR1 bit shows the backward direction).

Please notice, that in the case of the change of the direction the ITN and ISN bit in the DPLL_STATUS register are reset.

For this transition to the backward direction no change of address pointer APT and APT_2B is necessary.

profile update for TRIGGER when changing direction

The profile address pointer APT_2C is changed step by step in order to update the profile information in SYN_T, SYN_T old and PD_store:

- decrement APT_2C, load SYN_T
- decrement APT_2C, load SYN_T
- decrement APT_2C, load SYN_T, PD_store, update SYN_T_OLD
- decrement APT_2C, **make calculations**, load SYN_T and PD_store, update SYN_T_OLD and PD_store_old and wait for a new *TRIGGER* event

Note: *The update of SYN_T_OLD and the loading of PD_store can be performed in all steps above. The value of APT_2B needs not to be corrected. For a direction change from backwards to forwards make the same corrections by incrementing APT_2C.*

Make calculations does mean: the operation of the state machine starts with the calculations of NMB_T and INC_CNT1 using the actual APT_2C address pointer value, see **Section 28.20.2**.

The TBU_TB1 value is to be corrected by the number of pulses sent out in the wrong direction mode during the last and current increment. This correction is done by sending out SUB_INC1 pulses for decrementing TBU_TB1 (while DIR1=1).

Save inc_cnt1 value at direction change to inc_cnt1_save.

Calculate the new inc_cnt1 value as follows:

1. Stop sending pulses and save inc_cnt1 at the moment of direction change as inc_cnt1_save.
2. Set inc_cnt1 to the target value of the last increment
   nmb_t_tar_old
3. Add the target number of trigger which were calculated for the current increment when this value was already added to inc_cnt1 before the direction change is detected
   + nmb_t_tar
4. Subtract the value of still not sent pulses (remaining value at inc_cnt1_save)
   - inc_cnt1_save
5. Calculate the new target pulses to be sent considering the new values of SYN_T and PD_store and add them:
   + nmb_t_tar_new

This does mean the following equation:

inc_cnt1 = nmb_t_tar_old + nmb_t_tar
- inc_cnt1_save + nmb_t_tar_new

All pulses summarized at inc_cnt1 are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using PCM1 of DPLL_CTRL1 is possible during direction change.

When PSTC was incremented/decremented at the active slope and after that the direction change was detected at the same input event, correct **PSTC** once by

- nmb_t_tar_old when changed to backwards

+ nmb_t_tar_old when changed to forwards

in order to compensate the former operation. When the direction information is known before an intended change of PSTC, do not change them.

Store the new calculated value **nmb_t_tar_new** at **nmb_t_tar** for the correct calculation of PSTC at the next input event.

Consequences for STATE

With the next active *STATE* event the direction information is already given. The profile pointer APS_1C3 is to be corrected by a two times decrement in order to point to the profile of the next following increment. In the following it is decremented with each *STATE* event while DIR1=1. The SYN_S and PD_S_store values must be updated accordingly, including SYN_S_OLD and PD_S_store_old.

Because the right direction is already known when an input event appears, make the following corrections:

- decrement APS_1C3, load SYN_S and PD_S_store, update SYN_S_OLD and PD_S_store_old
- decrement APS_1C3, **make calculations**, load SYN_S and PD_S_store, update SYN_S_OLD and PD_S_store_old and wait for a new *STATE* event.

The update of SYN_S_OLD and the loading of PD_S_store can be performed in all steps above. The value of APS_1C2 needs not to be corrected.

When a new *STATE* event occurs, all address pointers are decremented accordingly as long as DIR1=1.

In **emergency mode** the pulses are corrected as follows:

Save inc_cnt1 value at direction change to inc_cnt1_save.

Calculate the new inc_cnt1 value as follows:

1. Stop sending pulses and save inc_cnt1 at the moment of direction change as inc_cnt1_save.
2. Set inc_cnt1 to the target value of the current increment **nmb_s_tar**
   Please notice, that in difference to the normal mode, nmb_s_tar is to be used instead of nmb_s_tar_old, because direction information in emergency mode is only given from the TRIGGER input and occurs of a STATE event independently. That means: The calculations at the last STATE event were done for the correct former direction. In addition still no pulse calculations are performed for the current increment, because the direction change is known at the moment of the recent STATE event.
   Later direction changes are considered at the next STATE event:
3. Do not add the calculated number of state pulses because no new STATE event occurred.
4. Subtract the value of still not sent target pulses (remaining value at inc_cnt1_save)
   - inc_cnt1_save
5. Add the new calculated target pulses for the current increment
   + nmb_s_tar_new

when for the calculation all new conditions of PD_S_store and SYN_S are considered.

inc_cnt1 = nmb_s_tar_old - inc_cnt1_save + nmb_s_tar_new

All pulses summarized at inc_cnt1 are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using PCM1 of DPLL_CTRL1 is possible during direction change.

Do not change PSSC and suppress incrementing/decrementing of PSSC at the event directly following to the direction change information.

Store the new calculated value **nmb_s_tar_new** at **nmb_s_tar** for the correct calculation of PSTC at the next input event.

Repeated change to forward direction for TRIGGER

The DIR1 bit remains set as long as the THMI value remains none violated for the following *TRIGGER* events and is reset when for an inactive TRIGGER slope the THMI is violated.

Resetting the DIR1 to 0 results (after repeated reset of LOCK1, ITN, ISN) the opposite correction of the profile address pointer considered.

This does mean two increment operations of the address pointer APS_1C3 including the update of SYN_S and PD_S_store with the automatic update of SYN_S_OLD and PD_S_store_old for STATE and four increment operations of the address pointer APT_2C including the update of SYN_T and PD_store with the automatic update of SYN_T_OLD and PD_store_old for TRIGGER.

The correction of TBU_CH1 is done by sending out the correction pulses with the highest possible frequency at SUB_INC1 while DIR1=0. The number of pulses is calculated as shown above.

Consequences for STATE

see corrections above. After that the address pointers are incremented again with each following active *STATE* event as long as DIR1=0.

28.20.8.6.3 Operation for direction change for TRIGGER (SMC=1)

When for SMC=1 a backwards condition is detected for the *TRIGGER* input signal (TDIR=1, resulting in DIR1=1), the LOCK1 bit in the DPLL_STATUS register is reset, the NUTE value in NUTC register is set to 1. The address pointers APT and APT_2C as well as APT_2B are decremented for each active slope of *TRIGGER* as long as the DIR1 bit shows the backward direction.

Please notice, that in the case of the change of the direction the ITN bit in the DPLL_STATUS register is reset.

Profile update for TRIGGER

Make the same update steps for the profile address pointer as shown in **Section 28.20.8.6.2**: Decrement APT_2C for 2 times with the update of the SYN_T and PD_store values at each step with an automatic update of SYN_T_OLD and PD_store_old:

- decrement APT_2C, load SYN_T, PD_store, update SYN_T_OLD
- decrement APT_2C, **make calculations**, load SYN_T and PD_store, update SYN_T_OLD and PD_store_old and wait for a new *TRIGGER* event.

In the normal case no correction of wrong pulses sent is necessary, because the direction change is detected by the pattern immediately.

Nevertheless a correction is necessary as shown below. In the other case: see treatment of pulses TBU_CH1_BASE in normal mode at **Section 28.20.8.6.2**.

Save inc_cntx value at direction change to inc_cnt1_save.

Calculate the new inc_cnt1 value as follows:

1. Clear inc_cnt1.
2. Set inc_cnt1 to the target value of the last increment
   nmb_t_tar
   Please notice, that in difference to the normal mode, nmb_t_tar is to be used instead of nmb_t_tar_old, because the direction information is known before the calculation takes place.
3. Do not add the calculated number of trigger pulses because it is not calculated yet before the direction change information is known.

4. Subtract the value of still not sent pulses (remaining value at inc_cnt1_save)
   - inc_cnt1_save
5. Add the new calculated target pulses for the current increment
   + nmb_t_tar_new

when for the calculation all new conditions of PD_S_store and SYN_S are considered.

inc_cnt1 = nmb_t_tar_old - inc_cnt1_save + nmb_t_tar_new

All pulses summarized at inc_cnt1 are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using PCM1 of DPLL_CTRL1 is possible during direction change.

Suppress changing of PSTC for the TRIGGER event when a direction change is detected.

Store the new calculated value **nmb_t_tar_new** at **nmb_t_tar** for the correct calculation of PSTC at the next input event.

<u>Repeated change to forward direction for TRIGGER</u>

The DIR1 bit remains set as long as the TDIR bit is set for the following *TRIGGER* events and is reset when for an active *TRIGGER* slope the TDIR is zero.

Resetting the DIR1 to 0 results (after repeated reset of LOCK1 and ITN) the opposite correction of the address pointer use.

This does mean two increment operations of the address pointer including the update of SYN_T and PD_store.

A complex correction of TBU_CH1_BASE and INC_CNT1 is in the normal case not necessary, when all increments are equal (SYN_NT=0) and no adapt information is used. In this case only the MLS1 value is added to INC_CNT1 in order to back count the value for the last increment. In the other case: see treatment of pulses TBU_CH1_BASE and ICN_CNT1 in normal mode at **Section 28.20.8.6.2**.

28.20.8.6.4 Operation for direction change for STATE (SMC=1)

When for SMC=1 a backwards condition is detected for the *STATE* input signal (SDIR=1, resulting in DIR2=1), the LOCK2 bit in the DPLL_STATUS register is reset, the NUSE value in NUSC register is set to 1 and the address pointers APS and APS_1C3_f and APS_1C2 are decremented for each active slope of *STATE* as long as the DIR2 bit shows the backward direction.

Please notice, that in the case of the change of the direction the ISN bit in the DPLL_STATUS register is reset.

For this transition to the backward direction no change of address pointer APS and APS_1C2 is necessary.

<u>Profile update for STATE</u>

Make the same update steps for the profile address pointer as shown in **Section 28.20.8.6.2**: Decrement APS_1C3 for 2 times with the update of the SYN_S, SYN_S_OLD, PD_S_store and PD_S_store_old values at each step:

- decrement APT_1c3, load SYN_S, PD_S_store, update SYN_S_OLD
- decrement APT_1c3, **make calculations**, load SYN_S and PD_S_store, update SYN_S_OLD and PD_S_store_old and wait for a new *STATE* event.

A complex correction of TBU_CH2_BASE and INC_CNT2 is in the normal case not necessary, when all increments are equal (SYN_NS=0) and no adapt information is used. In this case only the MLS2 value is added to INC_CNT2 in order to back count the value for the last increment. In the other case: see treatment of pulses TBU_CH1_BASE and ICN_CNT1 in normal mode at **Section 28.20.8.6.2**.

For the second PMSM the pulses are corrected as follows:

Save inc_cnt2 value at direction change to inc_cnt2_save.

Calculate the new inc_cnt2 value as follows:

1. Clear inc_cnt2.
2. Set inc_cnt2 to the target value of the last increment
   nmb_s_tar
   Please notice, that in difference to the normal mode, nmb_s_tar is to be used instead of nmb_s_tar_old, because no new calculation is performed so far.
3. Do not add the calculated number of state pulses because it is not calculated yet before the direction change information is known.
4. Subtract the value of still not sent pulses (remaining value at inc_cnt2_save)
   - inc_cnt2_save
5. Add the new calculated target pulses for the current increment
   + nmb_s_tar_new

when for the calculation all new conditions of PD_S_store and SYN_S are considered.

inc_cnt2 = nmb_s_tar_old - inc_cnt2_save + nmb_s_tar_new

All pulses summarized at inc_cnt2 are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using PCM2 of DPLL_CTRL1 is possible during direction change.

Do not change PSSC for a STATE event when a direction change is detected.

Store the new calculated value **nmb_s_tar_new** at **nmb_s_tar** for the correct calculation of PSTC at the next input event.

Repeated change to forward direction for STATE

The DIR2 bit remains set as long as the SDIR bit is set for the following *STATE* events and is reset when for an active *STATE* slope SDIR is zero.

Resetting the DIR2 to 0 results (after repeated reset of LOCK2 and FSD) in the opposite correction of the address pointer use.

After a last decrementing of all address pointers the APS_1C3 is incremented 2 times with a repeated update of SYN_S, SYN_S_OLD and PD_S_store after each increment.

28.20.8.6.5 DPLL reaction in the case of non plausible input signals

When the DPLL is synchronized concerning the *TRIGGER* signal by setting the FTD, SYT and LOCK1 bits in the DPLL_STATUS register, the number of active *TRIGGER* events between the gaps is to be checked continuously.

When additional events appear while a gap is expected, the LOCK1 bit is reset and the ITN bit in the DPLL_STATUS register is set.

When an unexpected gap appears (missing *TRIGGER*S), the NUTE value in the NUTC register is set to 1, the LOCK1 bit is reset and the ITN bit in the DPLL_STATUS register is set. The address pointers are incremented with the next active *TRIGGER* slope accordingly.

The TOR Bit in the DPLL_STATUS register is set, when the time to the next active *TRIGGER* slope exceeds the value of the last nominal *TRIGGER* duration multiplied with the value of the TLR register (see chapter **Section 28.20.12.72**). In this case also the TORI interrupt is generated, when enabled.

When in the following the direction DIR1 changes as described in the chapters above the ITN bit in the DPLL_STATUS register is reset, the use of the address pointers APT_2C is switched and the pulse correction takes place as described above.

In all other cases the CPU can interact to leave the instable state. This can be done by setting the APT_2C address pointer which results in a reset of the ITN bit. In the following NUTE can also be set to higher values.

When the DPLL is synchronized concerning the *STATE* signal by setting the FSD, SYS and LOCK1 (for SMC=0) or LOCK2 (for SMC=1) bits in the DPLL_STATUS register, the number of active *STATE* events between the gaps is to be checked continuously.

When additional events appear while a gap is expected or while an unexpected missing *STATE* event appears, the LOCK1,2 bit is reset and the ISN bit in the DPLL_STATUS register is set.

When an unexpected gap appears for RMO=SMC=1 (missing *STATE*s for synchronous motor control), the NUSE value in the NUSC register is set to 1, the LOCK2 bit is reset and the ISN bit in the DPLL_STATUS register is set. The address pointers are incremented with the next active *STATE* slope accordingly.

When the *STATE* locking range SLR is violated[7], the state machine 2 will remain in state 21 and the address pointer APS, APS_1C2 and APS_1C3 will remain unchanged until the CPU sets the APS_1C3 accordingly. In this case also the NUSE value in the NUSC register is set to 1. The DPLL stops the generation of the SUB_INC1,2 pulses respectively and will perform no other actions - remaining in step21 of the second state machine (see **Section 28.20.2**).

[7] The SOR Bit in the DPLL_STATUS register is set, when the time to the next active *STATE* slope exceeds the value of the last nominal *STATE* duration multiplied with the value of the SLR register (see **Section 28.20.12.73**).In this case also the SORI interrupt is generated, when enabled.

When in the following the direction DIR2 changes as described in the chapters above the ISN bit in the DPLL_STATUS register is reset, the use of the address pointers APS_1C3 is switched and the pulse correction takes place as described above. In all other cases the CPU must interact to leave the instable state. This can be done by setting the APS_1C3 address pointers which results in a reset of the ISN bit. In the following NUSE can also be set to higher values.

28.20.8.6.6 State description of the State Machine

Table 72 State description of the State Machine.

Step	Description	Comments

always for DEN=1	**for each inactive TRIGGER slope with TEN=1:**
	check, if the last active TRIGGER slope was passing the PVT check; only in this case perform the following tasks: calculate the time stamp difference ΔT to the last active event, store this value at THVAL;
	when THMI >0 is violated (ΔT < THMI):
	generate TINI interrupt,
	set DIR1=0 (forwards)
	set BWD1=0
	(see DPLL_STATUS register)
	else (only for THMI >0):
	set DIR1= 1 (backwards);
	set BWD1=1 (see DPLL_STATUS register)
	after changing the direction correct the pulses WP sent with wrong direction information and send the pulses for the actual increment in addition with highest possible frequency: WP=NMB_T-DPLL_INC_CNT1;
	correct INC_CNT1 by addition of 2*WP before sending the correction pulses;
	generate the TISI Interrupt.
	check THMA, when THMA is violated, generate the TAXI interrupt; go to step 1
	for each inactive STATE slope with SEN=1:
	set DIR2=DIR1
		for SMC=0;
		set DIR1 always after inc./ decr. the address pointers APT, APT_x;
		go to step 1;
		stop output of SUB_INC1 and correct pulses after changing DIR1 after incr./ decr. of APS_x set DIR2 always after incr./decr. the address pointers APS, APS_x;
		go to step 1

always for DEN=1 and (TEN=1 or SEN=1, respectively)	set DIR1=BWD1=TDIR, set DIR2=BWD2=SDIR; for each change of TDIR go to step 1 after performing the following calculations: correct INC_CNT1 correct the pulses (WP, see above) sent with wrong direction information and send the pulses for the actual increment in addition with highest possible frequency. For each change of SDIR go to step 21 after performing the following calculations: update of SYN_S, PD_S_store according to **Section 28.20.8.6.2** correct INC_CNT1,2 correct the pulses sent with wrong direction information and send the pulses for the actual increment in addition with highest possible frequency.	**for SMC=1**; set the direction bits always after incr./decr. the corresponding address pointers;

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments
1	When DEN = 0 or TEN=0: stay in step 1 until DEN=1, TEN=1 and at least one active *TRIGGER* has been detected (FTD=1); the following steps are performed always (not necessarily in step 1, but also in steps 18 to 20 (when waiting for new PMTR values to be calculated): compare TRIGGER_S with TSL (active slope); **When no active TRIGGER slope appears** and when TS_T_CHECK time is reached: • send missing *TRIGGER* INT, also when a gap is expected according to the profile; set MT=1 (missing *TRIGGER* bit) in the DPLL_STATUS register; do not leave the active step, until a valid active *TRIGGER* appears. **When an active TRIGGER slope appears check PVT** **- when the PVT value is violated:** generate the PWI interrupt, ignore the *TRIGGER* input and wait for the next active TRIGGER slope (ignore each inactive slope); do not store any value **- When the PVT value is fulfilled:** store the actual position stamp at PSTM (value at the TRIGGER event); update the RAM region 2 by equation DPLL-1a-c (see **Section 28.20.7.5**) **store the actual INC_CNT1 value at MP1 as missing pulses***(instead of calculation in step 5)*: store all relevant configuration bits **X** of the DPLL_CTRL(0,1) Registers in **shadow** registers and consider them for all corresponding calculations of steps 2 to 20 accordingly; the relevant bits are explained in the registers itself generate the TASI interrupt; for FTD=0: • set PSTC=PSTM • set FTD (first *TRIGGER* detected) • do not change PSTC,APT, APT_2B • for (RMO=0 or SMC=1) **and SGE1=1**: increment INC_CNT1 by (MLT+1)*) +MPVAL1***) • send SUB_INC1 pulses with highest possible frequency **when SGE1=1** and DPLL_CTRL11_SIP1=0, for SYT=0 and FTD =1: • **dir_crement** APT and APT_2B by one; • **dir_crement** for **SGE1_delay*****)=1: PSTC by NMB_T_TAR**) • for (RMO=0 or SMC=1) **and SGE1=1**: increment INC_CNT1 by (MLT+1)*) +MPVAL1***)	Depending on TSL, TEN, DEN step one is leaving with the next *TRIGGER* input; **Note:** Step 1 is also left in emergency mode when an active *TRIGGER* event appears in order to make a switch back to normal mode possible; _old - values are values valid at the last but one active *TRIGGER* event; for the whole table: use always MLS1 instead of (MLT+1) for the case SMC=1; **dir_crement** does mean: increment for DIR1=0 decrement for DIR1=1 *)replace (MLT+1) by MLS1 for SMC=1 **) NMB_T_TAR is the target value of NMB_T of the last increment (see step 5 ff.) ***) add MPVAL1 once to INC_CNT1, that means only when PCM1=1 ****) **SGE1_delay** is the value of SGE1 delayed by one active TRIGGER event *****) PD_store = 0 for AMT=0 (see DPLL_CTRL_0 register)

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments
1 (cont'd)	for SYT=1 and TOR=0: • **dir_crement** APT, APT_2C, **dir_crement** APT_2B by SYN_T_OLD • **dir_crement** for SGE1_delay****)<b>=1 PSTC by NMB_T_TAR**) • for (RMO=0 or SMC=1) **and SGE1=1**: increment INC_CNT1 by SYN_T*((MLT+1)*)+ PD_store*****) + MPVAL1***) PD_store is 0 for AMT=0 within the DPLL_STATUS register: • set LOCK1 bit accordingly;	
2	calculate TS_T according to equations DPLL-1a; calculate DT_T_ACT = TS_T - TS_T_OLD calculate RDT_T_ACT calculate QDT_TX according to equation DPLL-2	
3	send CDTI interrupt when NTI_CNT is zero or decrement NTI_CNT when not zero; calculate EDT_T and MEDT_T according to equations DPLL-3 and DPLL-4 for (RMO=1 and SMC=0): update SYN_T, PD_store and go back to step 1	Note: There are different behaviors of RM and HW-IP: For the HW-IP the values of SYN_T and PD_store are not updated until a new active TRIGGER slope occurs.
4	calculate CDT_TX according to equation DPLL-5a and b;	for RMO=0 or SMC=1;

5	calculate missing pulses:
	MP1 = INC_CNT1(at the moment of an active TRIGGER slope)
	
	calculate target pulses:
	NMB_T_TAR = (MLT+1)*)*SYN_T + PD_store + MPVAL1 (instead of PD_store use zero in the case AMT=0)
		for RMO=0 or SMC=1;
		
		*)replace (MLT+1) by MLS1 for SMC=1;
		
		add MPVAL1 only for PCM=1 and reset PCM1 after that;

6	sent MP with highest possible frequency and set
	NMB_T = NMB_T_TAR
		for RMO=0 or SMC=1,
		DMO=0 and COA=0

7	calculate the number of pulses to be sent
	NMB_T = NMB_T_TAR + MP (see equations DPLL-21 or DPLL-27 respectively)
		for RMO=0 or SMC=1,
		DMO=0 and COA=1

8	NMB_T = SYN_T*CNT_NUM_1	for RMO=0 or SMC=1,
		DMO=1

9	update SYN_T and PD_store;	Note: There are different behaviors of RM and HW-IP: For the HW-IP the values of SYN_T and PD_store are not updated until a new active TRIGGER slope occurs.

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments

10	calculate ADD_IN_CAL1 according to equation DPLL-25 and DPLL-25b or DPLL-31 and store this value in RAM
	use ADD_IN_CAL1 as ADD_IN value for the case DLM=0
	use ADD_IN_LD1 as ADD_IN for the case DLM=1, but do this update immediately (without waiting for this step 10);
	for DMO=DLM=0 and EN_C1u=0:
	reset the flip-flops in the SUB_INC1 generator;
	start sending SUB_INC1;
		for RMO=0 or SMC=1
		for DLM=0
		for DLM=1

11	calculate TS_T_CHECK = TS_T + DT_T_ACT *(TOV) ;	for RMO=0 or SMC=1;

12	automatic setting of actions masking bits in the DPLL_STATUS register:
	for SMC=0: set CAIP1=CAIP2=1
	for SMC=1: set only CAIP1=1
		steps 12 to 16 are not valid for the combination: (SMC=0 and RMO=1)

13	for all correspondent actions with ACT_N[i]=1 calculate: NA[i] = (PSA[i] - PSTC)/(MLT+1)$^{*)}$for forward direction with w= integer part and b = remainder of the division (fractional part); for backward direction use NA[i] = (PSTC - PSA[i])/(MLT+1)$^{*)}$ and consider in both cases the time base overflow in order to get a positive difference	actions 0…11 for SMC=1 actions 0…23 for SMC=0 depending on ACT_N[i] in **DPLL_ACT_STA** register; replace MLT+1 by MLS1 for SMC=1
14	calculate PDT_T[i] and DTA[i] for up to 24 action values according to equations DPLL-11 and DPLL-12;	actions 0…11 for SMC=1 actions 0…23 for SMC=0
15	calculate TSAC[i] according to equation DPLL-15 and PSAC[i] according to equation DPLL-17	actions 0…11 for SMC=1 actions 0…23 for SMC=0

16	automatic resetting of actions masking bits in the DPLL_STATUS register:
	for SMC=0: set CAIP1=CAIP2=0
	for SMC=1: set only CAIP1=0;
	set the corresponding ACT_N[i] bits in the DPLL_ACT_STA register
		Set ACT_N[i] for all enabled actions concerned:
		0…11 for SMC=1
		0…23 for SMC=0

17	check the relation of the last increment to its predecessor according to the profile and taking into account TOV: set the ITN status bit and reset the corresponding LOCK bit, when not plausible; go to step 18, when no active *TRIGGER* appears **for all following steps 18 to 20:go immediately back to step 1, when an active TRIGGER event occurs, interrupt all calculations there and reset all CAIP in that case; when going back to step 1:** store TS_T in RAM 2b according to APT_2B; update RAM 2a and RAM 2d	for all conditions

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments
18	wait for a new PMTR value; set the corresponding CAIPx values and go to step 19 in that case	go immediately to step 1 and update the RAM according to step 17 when an active *TRIGGER* event occurs
19	make the requested action calculation according to new PMTR values	go immediately to step 1 and update the RAM according to step 17 when an active *TRIGGER* event occurs
20	reset CAIPx and go back to step 18	go immediately to step 1 and update the RAM according to step 17 when an active *TRIGGER* event occurs
21	When DEN = 0 or SEN=0: make sure that the first active slope of STATE is detected; stay in step 1 until DEN=1, SEN=1 and at least one active *STATE* has been detected (FSD=1); the following steps are performed always (not necessarily in step 21, but also in steps 38 to 40 (when waiting for new PMTR values to be calculated): compare STATE_S with SSL (active slope); for each inactive slope: generate a SISI interrupt; • send missing *STATE* INT when TS_S_CHECK time is reached and set MS=1 (missing *STATE* bits) in that case; do not leave step 21 while no active *STATE* appears. **When an active STATE slope appears:** store the actual position stamp at PSSM (value at the STATE event) update RAM by equation DPLL-6a-c (see **Section 28.20.7.5**); **store the actual INC_CNT1/2 at MP1/MP2 respectively as missing pulses** (instead of calculations in step 25) store all relevant configuration bits **X** of the DPLL_CTRL(0,1) Registers in **shadow** registers and consider them for all corresponding calculations of steps 22 to 37 accordingly; the relevant bits are explained in the registers itself for FSD=0: • set PSSC=PSSM • set FSD (first *STATE* detected) • do not increment PSSC • for (RMO=1 and SMC=0) **and SGE1=1**>: increment INC_CNT1 by MLS1+MPVAL1 **) • for (RMO=1 and SMC=1) **nd SGE2=1**: increment INC_CNT2 by MLS2+MPVAL2 **)	Depending on SSL, SEN, DEN step 21 is leaving with the next *STATE* input; for the steps 22-37: for SMC=1 replace: MLS1 by MLS2, LOCK1 by LOCK2; SUB_INC1 by SUB_INC2; CNT_NUM_1 by CNT_NUM_2; MPVAL1 by MPVAL2; EN_C1u by EN_C2u; **dir_crement** does mean: increment for DIR2=0 decrement for DIR2=1 or DIR1 respectively *) target number of pulses of the last increment (see step 25 ff.) **) add MPVAL1 or MPVAL2 only once, that means as long as PCM1 or PCM2 is set respectively ***) **SGE1_delay** is the value of SGE1 delayed by one active STATE event ****) **SGE2_delay** is the value of SGE2 delayed by one active STATE event *****) PD_S_store = 0 for AMS=0 (see DPLL_CTRL_0 register)

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments
21 cont'd	for SYS=0, FSD =1: • **dir_crement** PSSC by NMB_S_TAR*) for (SMC=0**and SGE1_delay***)=1**) or (SMC=1 and **SGE2_delay****)=1**) • increment INC_CNT1 by MLS1+MPVAL1**) (for SMC=0, **SGE1=1** and RMO=1); • increment INC_CNT2 by MLS2+MPVAL2**)(for SMC=1, **SGE2=1** and RMO=1); • **dir_crement** APS and APS_1C2 for SYS=1: • **dir_crement** APS and APS_1C3 • **dir_crement** APS_1C2 by SYN_S_OLD • for RMO=1 and SMC=0: for **SGE1_delay***)=1dir_crement** PSSC by NMB_S_TAR*); for **SGE1=1** increment INC_CNT1 by SYN_S*MLS1 + PD_S_store + MPVAL1**) • for RMO=1 and SMC=1: for**SGE2_delay****)=1dir_crement** PSSC by NMB_S_TAR*); for **SGE2=1** increment INC_CNT2 by SYN_S*(MLS2 + PD_S_store)*****) +MPVAL2**) • within the DPLL_STATUS register: set LOCK1 or 2 bit accordingly;	
22	calculate TS_S according to equations DPLL-6a; calculate DT_S_ACT = TS_S - TS_S_OLD calculate RDT_S_ACT calculate QDT_SX	

23	send CDSI interrupt;
	calculate EDT_S and MEDT_S according to equations DPLL-8 and DPLL-9
	for RMO=0:
	go back to step 21 for RMO=0 and update SYN_S and PD_S_store using the current ADT_S[i] values in that case;
		Note: There are different behaviors of RM and HW-IP: For the HW-IP the values of SYN_S and PD_S_store are not updated until a new active STATE slope occurs.

24	calculate CDT_SX according to equation DPLL-10a and b;	only for RMO=1
25	calculate missing pulses - for TBU_CH1: MP1 = INC_CNT1(active STATE slope) - for TBU_CH2: MP2 = INC_CNT2(active STATE slope) calculate target number of pulses: NMB_S_TAR = (MLS1 + PD_S_store)*SYN_S + PD_S_store+MPVAL1 (for SMC=0) NMB_S_TAR = MLS2*(SYN_S + PD_S_store) + MPVAL2 (for SMC=1) (instead of PD_S_store use zero in the case AMS=0)	only for RMO=1 for SMC=0 instead of MPVAL1 use zero for PCM1=0f or SMC=1 instead of MPVAL2 use zero for PCM2=0; add MPVAL1/2 once to INC_CNT1/2 and reset PCM1/2 after that

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments
26	sent MPx with highest possible frequency and set NMB_S = NMB_S_TAR	only for RMO=1, DMO=0 and COA=0
27	calculate number of pulses to be sent according to DPLL-22 or NMB_S = NMB_S_TAR + MPx	only for RMO=1, DMO=0 and COA=1

28	NMB_S = SYN_S*CNT_NUM_1 (SMC=0)
	NMB_S = SYN_S*CNT_NUM_2 (SMC=1)
		only for RMO=1,
		DMO=1

29	update SYN_S and PD_S_store;	Note: There are different behaviors of RM and HW-IP: For the HW-IP the values of SYN_S and PD_S_store are not updated until a new active STATE slope occurs.

30	calculate ADD_IN_CAL2 according to equation DPLL-26 and DPLL-26b or DPLL-31 respectively and store this value in RAM
	use ADD_IN_CAL2 as ADD_IN value for the case DLM=0
	use ADD_IN_LD2 as ADD_IN for the case DLM=1, but do this update immediately (without waiting for this step 30);
	for RMO=1, DMO=DLM=0 and EN_C1u=0 (EN_C1u=0):
	reset the flip-flops in the SUB_INC1 or SUB_INC2 generator respectively;
	start sending SUB_INC1 / SUB_INC2;
		only for RMO=1
		
		for DLM=0
		for DLM=1

31	calculate TS_S_CHECK = TS_S + DT_S _ACT * (SOV);	only for RMO=1;
32	automatic setting of actions masking bits in the DPLL_STATUS register: CAIP1 and CAIP2 for SMC=0 only CAIP2 for SMC=1	for RMO=1

33	for all actions with ACT_N[i]=0 calculate:
	NA[i] = (PSA[i] - PSSC)/MLS1
	for forward direction with
	w = integer part and
	b = remainder of the division (fractional part)
	for backward direction use
	NA[i] = (PSSC - PSA[i])/(MLS1)
	and consider in both cases the time base overflow in order to get a positive difference
	use MLS2 as divider in the case of SMC=1
		for SMC=0: 24 actions,
		for SMC=1: 12 actions;
		depending on ACT_N[i] in **DPLL_ACT_STA** register

34	calculate PDT_S[i] and DTA[i] for up to 24 action values according to equations DPLL-13 and DPLL-14;	only for RMO=1;
		for SMC=0 actions 0…23
		for SMC=1 actions 12…23

35	calculate TSAC[i] according to equation DPLL-18 and PSAC[i] according to equation DPLL-20	for the relevant actions (see above) and RMO=1

Table 72 State description of the State Machine. (cont'd)

Step	Description	Comments
36	automatic reset of the actions masking bit CAIP in the DPLL_STATUS register: CAIP1=CAIP2=0 for SMC=0 and only CAIP2=0 for SMC=1 set the corresponding ACT_N[i] bits in the DPLL_ACT_STA register	for the relevant actions (see above) and RMO=1 Set ACT_N[i] and reset ACT_WRi for all enabled actions
37	check the duration of the last increment to its predecessor according to the profile and taking into account SOV: set the ISN status bit and reset the corresponding LOCK bit, when not plausible; go to step 38, when no active *STATE* appears **for all following steps 38 to 40:** **go immediately back to step 21, when an active STATE event occurs, interrupt all calculations there and reset all CAIPx in that case;** **when going back to step 21:** store TS_S in RAM 1c2 according to APS_1C2; update RAM 1c1 and RAM 1c4	for all conditions
38	wait for a new PMTR value; set the corresponding CAIPx values and go to step 39 in that case	go immediately to step 21 and update the RAM according to step 37 when an active *STATE* event occurs
39	make the requested action calculation according to new PMTR values	go immediately to step 21 and update the RAM according to step 37 when an active *STATE* event occurs
40	reset CAIP and go back to step 38	go immediately to step 21 and update the RAM according to step 37 when an active *STATE* event occurs

28.20.9 DPLL Interrupt signals

The DPLL provides 27 interrupt lines. These interrupts are shown below.

Table 73 DPLL Interrupt signals

Signal	Description
DPLL_DCGI_IRQ	Direction change
DPLL_SORI_IRQ	STATE is out of range
DPLL_TORI_IRQ	TRIGGER is out of range
DPLL_CDSI_IRQ	STATE duration calculated for last increment
DPLL_CDTI_IRQ	TRIGGER duration calculated for last increment
DPLL_TE4_IRQ	TRIGGER event interrupt 4 request3)
DPLL_TE3_IRQ	TRIGGER event interrupt 3 request3)

Table 73 DPLL Interrupt signals (cont'd)

Signal	Description
DPLL_TE2_IRQ	TRIGGER event interrupt 2 request3)
DPLL_TE1_IRQ	TRIGGER event interrupt 1 request3)
DPLL_TE0_IRQ	TRIGGER event interrupt 0 request3)
DPLL_LL2_IRQ	Loss of lock interrupt for SUB_INC2 request
DPLL_GL2_IRQ	Get of lock interrupt for SUB_INC2 request
DPLL_E_IRQ	Error interrupt request
DPLL_LL1_IRQ	Loss of lock interrupt for SUB_INC1 request
DPLL_GL1_IRQ	Get of lock interrupt for SUB_INC1 request
DPLL_W1_IRQ	Write access to RAM region 1b or 1c interrupt request
DPLL_W2_IRQ	Write access to RAM region 2 interrupt request
DPLL_PW_IRQ	Plausibility window violation interrupt of TRIGGER request
DPLL_TAS_IRQ	TRIGGER active slope while NTI_CNT is zero interrupt request
DPLL_SAS_IRQ	STATE active slope interrupt request
DPLL_MT_IRQ	Missing TRIGGER interrupt request
DPLL_MS_IRQ	Missing STATE interrupt request
DPLL_TIS_IRQ	TRIGGER inactive slope interrupt request
DPLL_SIS_IRQ	STATE inactive slope interrupt request
DPLL_TAX_IRQ	TRIGGER maximum hold time violation interrupt request
DPLL_TIN_IRQ	TRIGGER minimum hold time violation interrupt request
DPLL_PE_IRQ	DPLL enable interrupt request
DPLL_PD_IRQ	DPLL disable interrupt request

Note: TEi_IRQ depends on the TINT value in ADT_T[i][1] and is only active when SYT[2] =1.

[1] see RAM region 2 explanations; see **Section 28.20.14**

[2] see DPLL STATUS register; see **Register DPLL_STATUS**

[3] see TINT value in the corresponding ADT_T[i] section of RAM region 2; see **Section 28.20.14.3**

28.20.10 MCS to DPLL interface

A reduced AEI interface is implemented in the DPLL, which can only be accessed by the MCS Bus Master interface in the same cluster. The purpose of this interface is to enable a faster interchange of data between the MCS and the DPLL, while enabling a certain control over the DPLL internal state machine.

28.20.10.1Architecture and organization

The implemented interface has an address width of 4 bits, while the size of the data interface is 24 bits.

The following table shows the implemented AEI addresses from the MCS side. Label RD refers to the label used for the address when reading from the MCS or writing from the DPLL, whereas Label WR refers to the label used for the address when writing from the MCS or reading from the DPLL.

28.20.10.2General functionality

In order to have a better understanding of the implications when this interface is used, the following working concepts are informally defined here. They refer to the STATE engine operation when DPLL_CTRL_11.STATE_EXT is set.

Update of ram:

Operation which stores TSF_S, DT_S_ACTUAL and RDT_S_ACTUAL back to the RAM and reads the profile.

Calculation of sub-increments:

Calculation of DT_S_ACTUAL, RDT_S_ACTUAL, NMB_S and ADD_IN.

Change of direction:

Update of profile and its increment or decrement (only in the STATE processor).

Calculation of actions (PMT):

Where the calculation of PSAC and TSAC is performed (only in state processor).

If **DPLL_CTRL_11.STATE_EXT is not set**, the DPLL will ignore the data written to this interface from the MCS. The DPLL will not update the interface either and a read done to this interface from the MCS can obtain out-of-date information.

If **DPLL_CTRL_11.STATE_EXT is set**, some modifications are done to the way that the DPLL module works when using the STATE engine.

Up to 128 STATE events can be handled.

RAM1c is not used anymore. Instead, the data needed to perform each of the already described operations is fetched from registers in this interface. The data that would have to be written back to RAM1c is also written to this interface.

In each of the procedures described above, the DPLL will enter in one or more stalled states, in which it will wait for one or more words to be written to MCS2DPLL_DEB15 (STATUS_INFO)

Table 74 Correspondence between STA_S values and their unlocking keywords

Operation	STA_S value	First Keyword	Second Keyword
Update of ram	0b00001_001	0xE	0x1
Calculation of sub-increments	0b00010_000	0xD	0x2
Calculation of actions	0b01110_000	0xC	0x3
Change of direction	0b00000_100	0xB	0x4
Change of direction	0b00000_110	0xB	0x4

The stalled STATE in the DPLL is freed by writing the upper keywords to MCS2DPLL_DEB15. Please note the requirements on DPLL level (described in **Section 28.20.15**) regarding the data on the interface that has to be written by the MCS program. If this data is incorrectly delivered or the STATE state machine is unlocked before delivery, the proper signal processing of the DPLL cannot be assured.

For the particular case of an update of ram after a virtual increment, the data field TSF_S is not calculated completely by the DPLL STATE processing unit. Instead, the values needed in order to fill this data field are provided (**Section 28.20.7.5.6** and **Section 28.20.7.5.7**)

28.20.10.3MCS to DPLL Register overview

The following registers of the MCS2DPLL interface are exclusively accessible by the MCS instance of cluster 0 and cannot be accessed directly via CPU.

Table 75 MCS to DPLL Register overview

Address offset	Common Label	Label RD	Label WR	see Page
0x0	MCS2DPLL_DEB0	DT_S_P	DT_S_P1	650
0x4	MCS2DPLL_DEB1	not used	RDT_S_P1	650
0x8	MCS2DPLL_DEB2	TS_SX	RDT_S_PQ1	651
0xC	MCS2DPLL_DEB3	DT_SX	DT_S_PQ	652
0x10	MCS2DPLL_DEB4	SYN_S_OLD	RDT_S_PQ	652
0x14	MCS2DPLL_DEB5	M_DW	DT_S_PQ1	653
0x18	MCS2DPLL_DEB6	not used	ADT_S_P	654
0x1C	MCS2DPLL_DEB7	RDT_S_P_RD	S_P_RD	655
0x20	MCS2DPLL_DEB8	not used	TSF_S_P	655
0x24	MCS2DPLL_DEB9	not used	TSF_S_P_MQ	656
0x28	MCS2DPLL_DEB10	not used	TSF_S_P_PM_MQ	657
0x2C	MCS2DPLL_DEB11	not used	TSF_S_P_PM	657
0x30	MCS2DPLL_DEB12	not used	ADT_S_P1	658
0x34	MCS2DPLL_DEB13	not used	not used	658
0x38	MCS2DPLL_DEB14	not used	not used	659
0x3C	MCS2DPLL_DEB15	not used	STATUS_INFO	659

28.20.11 DPLL Register Memory overview

The available registers and the size of the RAM area 2 depends on the chosen device.

Please refer to device specific appendix.

28.20.11.1Available DPLL register overview

Table 76 Available DPLL register overview

Register name	Description	see Page
DPLL_CTRL_0	DPLL Control Register 0	511
DPLL_CTRL_1	DPLL Control Register 1	514
DPLL_CTRL_2	DPLL Control Register 2 (actions 0-7 enable)	520
DPLL_CTRL_3	DPLL Control Register 3 (actions 8-15 enable)	521
DPLL_CTRL_4	DPLL Control Register 4 (actions 16-23 enable)	522
DPLL_CTRL_5[1)]	DPLL Control Register 5 (actions 24-31 enable)	523
DPLL_ACT_STA	DPLL ACTION Status Register with connected shadow register	524
DPLL_OSW	DPLL Offset and switch old/new address register	525
DPLL_AOSV_2	DPLL Address offset register for APT in RAM region 2	527

Table 76 **Available DPLL register overview** (cont'd)

Register name	Description	see Page
DPLL_APT	DPLL Actual RAM pointer to RAM regions 2a, b and d	528
DPLL_APS	DPLL Actual RAM pointer to regions 1c1, 1c2 and 1c4	530
DPLL_APT_2C	DPLL Actual RAM pointer to RAM region 2c	531
DPLL_APS_1C3	DPLL Actual RAM pointer to RAM region 1c3	532
DPLL_NUTC	DPLL Number of recent TRIGGER events used for calculations (mod 2*(TNU +1-SYN_NT))	533
DPLL_NUSC	DPLL Number of recent STATE events used for calculations (e.g. mod 2*(SNU +1-SYN_NS) for SYSF=0)	535
DPLL_NTI_CNT	DPLL Number of active TRIGGER events to interrupt	537
DPLL_IRQ_NOTIFY	DPLL Interrupt notification register	538
DPLL_IRQ_EN	DPLL Interrupt enable register	541
DPLL_IRQ_FORCINT	DPLL Interrupt force register	544
DPLL_IRQ_MODE	DPLL Interrupt mode register	546
DPLL_EIRQ_EN	DPLL Error interrupt enable register	546
DPLL_INC_CNT1	DPLL Counter for pulses for TBU_CH1_BASE to be sent in automatic end mode	549
DPLL_INC_CNT2	DPLL Counter for pulses for TBU_CH2_BASE to be sent in automatic end mode when SMC=RMO=1	549
DPLL_APT_SYNC	DPLL old RAM pointer and offset value for TRIGGER	550
DPLL_APS_SYNC	DPLL old RAM pointer and offset value for STATE	551
DPLL_TBU_TS0_T	DPLL TBU_CH0_BASE value at last TRIGGER event	553
DPLL_TBU_TS0_S	DPLL TBU_CH0_BASE value at last STATE event	553
DPLL_ADD_IN_LD1	DPLL direct load input value for SUB_INC1	554
DPLL_ADD_IN_LD2	DPLL direct load input value for SUB_INC2	555
DPLL_STATUS	DPLL Status Register	556
DPLL_ID_PMTR_[z] (z:0…31)[2]	DPLL 9 bit ID information for input signals PMT z 3)	562
DPLL_CTRL_0_SHADOW_TRIGGER	DPLL shadow register of DPLL_CTRL_0	562
DPLL_CTRL_0_SHADOW_STATE	DPLL shadow register of DPLL_CTRL_0	563
DPLL_CTRL_1_SHADOW_TRIGGER	DPLL shadow register of DPLL_CTRL_1	564
DPLL_CTRL_1_SHADOW_STATE	DPLL shadow register of DPLL_CTRL_1	565
DPLL_RAM_INI	DPLL initialization control and status for RAMs	566

[1] This register is only available for device 4.

[2] The registers DPLL_ID_PMTR 24-31 are not available for all devices.

28.20.11.2RAM Region 1a map description

Table 77 RAM Region 1a map description

Memory name	Description	Details on Page
PSA[i] (i:0…NOAC-1)	Position/Value request for action i	643
DLA[i] (i:0…NOAC-1)	Time to react before PSAi	643
NA[i] (i:0…NOAC-1)	Number of TRIGGER/STATE increments to ACTION i	644
DTA[i] (i:0…NOAC-1)	Calculated relative time to ACTION i	645

[1] The values PSA24-31, DLA24-31, NA24-31 and DTA24-31 in RAM 1a are not available for all devices. Please refer to device specific appendix.

28.20.11.3RAM Region 1b map description

Table 78 RAM Region 1b map description

Memory name	Description	see Page
TS_T	Actual signal TRIGGER time stamp register TRIGGER_TS	567
TS_T_OLD	Previous signal TRIGGER time stamp register TRIGGER_TS_OLD	567
FTV_T	Actual signal TRIGGER filter value	568
TS_S	Actual signal STATE time stamp register STATE_TS	569
TS_S_OLD	Previous signal STATE time stamp register STATE_TS_OLD	569
FTV_S	Actual signal STATE filter value	570
THMI	TRIGGER hold time min. value	571
THMA	TRIGGER hold time max. value	572
THVAL	measured last pulse time from active to inactive TRIGGER slope	572
TOV	Time out value of TRIGGER, according to the last nominal increment for a missing TRIGGER	573
TOV_S	Time out value of STATE, according to the last nominal increment for a missing STATE	574
ADD_IN_CAL1	calculated ADD_IN value for SUB_INC1 generation	575
ADD_IN_CAL2	calculated ADD_IN value for SUB_INC2 generation	576
MPVAL1	missing pulses to be added/subtracted directly to SUB_INC1 and INC_CNT1 once	577
MPVAL2	missing pulses to be added/subtracted directly to SUB_INC2 and INC_CNT2 once	578
NMB_T_TAR	target number of TRIGGER pulses	579
NMB_T_TAR_OLD	target number of TRIGGER pulses	580
NMB_S_TAR	target number of STATE pulses	581
NMB_S_TAR_OLD	target number of STATE pulses	582
RCDT_TX	reciprocal value of expected increment duration (T)	583

Table 78 RAM Region 1b map description (cont'd)

Memory name	Description	see Page
RCDT_SX	reciprocal value of expected increment duration (S)	583
RCDT_TX_NOM	reciprocal value of the expected nominal increment duration (T)	584
RCDT_SX_NOM	reciprocal value of the expected nominal increment duration (S)	585
RDT_T_ACT	actual reciprocal value of TRIGGER	586
RDT_S_ACT	actual reciprocal value of STATE	586
DT_T_ACT	Duration of last TRIGGER increment	587
DT_S_ACT	Duration of last STATE increment	588
EDT_T	Absolute error of prediction for last TRIGGER increment	589
MEDT_T	Average absolute error of prediction up to the last TRIGGER increment	589
EDT_S	absolute error of prediction for last STATE increment	590
MEDT_S	Average absolute error of prediction up to the last STATE increment	591
CDT_TX	Expected duration of current TRIGGER increment	592
CDT_SX	Expected duration of current STATE increment	592
CDT_TX_NOM	Expected nominal duration of current TRIGGER increment (without consideration of missing events)	593
CDT_SX_NOM	Expected nominal duration of current STATE increment (without consideration of missing events)	594
TLR	TRIGGER locking range value; the TOR bit in the DPLL_STATUS register is set when violated	594
SLR	STATE locking range value; the SOR bit is set when violated	595
PDT_[i] (i:0…NOAC-1)[1]	predicted time to ACTION i	596
MLS1	Calculated number of sub-pulses between two STATE events (to be set by CPU)	597
MLS2	Calculated number of sub-pulses between two STATE events (to be set by CPU) for the use when SMC=RMO=1	598
CNT_NUM_1	number of sub-pulses of SUB_INC1 in continuous mode, updated by the host only	598
CNT_NUM_2	number of sub-pulses of SUB_INC2 in continuous mode, updated by the host only	599
PVT	Plausibility value of next active TRIGGER slope	600
PSTC	Accurate calculated position stamp of last TRIGGER input;	601
PSSC	Accurate calculated position stamp of last STATE input;	602
PSTM	Measured position stamp at last active TRIGGER input	602
PSTM_OLD	Measured position stamp at last but one active TRIGGER input	602
PSSM	Measured position stamp at last active STATE input	604
PSSM_OLD	Measured position stamp at last but one active STATE input	604
NMB_T	Number of pulses of current increment in normal mode for SUB_INC1 (see equation DPLL-21 or for SMC=1 equation DPLL-27 respectively)	606
NMB_S	Number of pulses of current increment in emergency mod for SUB_INC1 (see equation DPLL-22) or in the case SMC=1 for SUB_INC2 (see equation DPLL-28)	606

1) The values PDT_24 to PDT_31 in RAM1b are not available for all devices.

28.20.11.4RAM Region 1c map description

Table 79 RAM Region 1c map description

Memory name	Description	see Page
RDT_S[i] (i:0…63)	Part of RAM1c1. Reciprocal value of the corresponding successive increment i, for each true nominal increment.	607
TSF_S[i] (i:0…63)	Part of RAM1c2. Time stamp field for state events, for each true nominal increment plus each virtual increment.	608
ADT_S[i] (i:0…63)	Part of RAM1c3. Adapt values for the current STATE increment, for each true nominal increment.	609
DT_S[i] (i:0…63)	Part of RAM1c4. Uncorrected last increment value of STATE for full scale, for each true nominal increment.	610

28.20.11.5Register Region EXT description

Table 80 Register Region EXT description

Register name	Description	see Page
DPLL_TSAC[z] (z:0…NOAC-1)	DPLL calculated action time stamps for action z	611
DPLL_PSAC[z] (z:0…NOAC-1)	DPLL calculated action position stamps for action z	611
DPLL_ACB_[z] (z:0…(NOAC/4)-1)	DPLL control bits for actions ((4*z)…(4*z)+3)	612
DPLL_CTRL_11	DPLL control register	614
DPLL_THVAL2	DPLL immediate THVAL value	622
DPLL_TIDEL	DPLL additional TRIGGER input delay	623
DPLL_SIDEL	DPLL additional STATE input delay	623
DPLL_CTN_MIN	CDT_T_NOM minimum value	624
DPLL_CTN_MAX	CDT_T_NOM maximum value	624
DPLL_CSN_MIN	CDT_S_NOM minimum value	625
DPLL_CSN_MAX	CDT_S_NOM maximum value	625
DPLL_STA	DPLL state machine status information	626
DPLL_INCF1_OFFSET	DPLL ADD_IN_ADDER1 offset for fast pulse generation	630
DPLL_INCF2_OFFSET	DPLL ADD_IN_ADDER2 offset for fast pulse generation	630
DPLL_DT_T_START	DPLL first value of DPLL_DT_T_ACT for the first increment after setting SIP1 from 0 to 1.	631
DPLL_DT_S_START	DPLL first value of DPLL_DT_S_ACT for the first increment after setting SIP2 from 0 to 1.	632
DPLL_STA_MASK	DPLL trigger masks for signals DPLL_STA_T and DPLL_STA_S	632
DPLL_STA_FLAG	DPLL STA_T/S and INC_CNT1/2 flags	633
DPLL_INC_CNT1_MASK	DPLL INC_CNT1 trigger mask	634

Table 80 Register Region EXT description (cont'd)

Register name	Description	see Page
DPLL_INC_CNT2_MASK	DPLL INC_CNT2 trigger mask	635
DPLL_NUSC_EXT1	Extension register number 1 for DPLL_NUSC[1]	635
DPLL_NUSC_EXT2	Extension register number 2 for DPLL_NUSC[1]	636
DPLL_APS_EXT	Extension register for DPLL_APS[1]	637
DPLL_APS_1C3_EXT	Extension register for DPLL_APS_1C3[1]	639
DPLL_APS_SYNC_EXT	Extension register for DPLL_APS_SYNC[1]	640
DPLL_CTRL_EXT	Extension register for DPLL_CTRL[1]	641

[1] These registers will return AEI_STATUS = b#10 if DPLL_CTRL_11.STATE_EXT is not set.

28.20.11.6RAM Region 2 map description

Table 81 RAM Region 2 map description

Memory name	Description	see Page
RDT_T[i] (i:0…AOSV_2B/4-1)	Region 2a. Reciprocal value of the corresponding successive increment i, for each true nominal increment.	646
TSF_T[i] (i:0…AOSV_2B/4-1)	Region 2b. Time Stamp Field for TRIGGER event i, for each true nominal increment plus each virtual increment.	647
ADT_T[i] (i:0…AOSV_2B/4-1)	Region 2c. Adapt values for the current TRIGGER increment i, for each true nominal increment.	647
DT_T[i] (i:0…AOSV_2B/4-1)	Region 2d. Uncorrected last increment value of TRIGGER i, for each true nominal increment.	649

Note: For each of the regions, the maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.

The description of registers is beginning at the register DPLL_CTRL_0.

The description of RAM regions is beginning at RAM 1a (see below): Bits 31 down to 24 in each RAM region are not implemented and therefore always read as zero (reserved). Other bits which are declared as reserved are not protected against writing. Reserved address regions are not protected against writing.

The description of the memory region RAM 1a begins with memory element PSA[i]: The RAM region 1a is writable only for DEN=0 (see DPLL_CTRL_1 register).

The description of memory region RAM1b begins with memory element TS_T.

The description of memory region RAM1c begins with memory element RDT_S.

The description of register region EXT begins with the register DPLL_TSAC[z]: This is an extension of the normal register region above in order to allow up to 32 action calculations and later specification modifications.

The description of the memory region RAM 2 begins with memory element RDT_T.

28.20.12 DPLL Register and Memory description

28.20.12.1Register DPLL_CTRL_0

DPLL Control Register 0

DPLL_CTRL_0

DPLL Control Register 0 **(028000_H)** **Application Reset Value: 003B BA57$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMO	**TEN**	**SEN**	**IDT**	**IDS**	**AMT**	**AMS**	**TNU**								
rw	rw	rw	rw	rw	rw	rw	rw								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SNU					**IFP**	**MLT**									
rw					rw	rw									

Field	Bits	Type	Description
MLT	9:0	rw	**Multiplier for TRIGGER** [1] MLT+1 is number of *SUB_INC1* pulses between two *TRIGGER* events in normal mode (1...1024); *Note: For emergency mode the number of SUB_INC1 pulses between two STATE events is calculated by the CPU using the formula MLS1=(MLT+1)*(TNU+1) / (SNU+1) in order to get the same number of SUB_INC1 pulses for FULL_SCALE. This value is stored in RAM at 0x05C0. Change of MLT by the CPU must result in the corresponding change of MLS1 by the CPU for SMC=0.* *Note: The number of MLT events is the binary value plus 1. The value MLT+1 is replaced by MLS1 in the case of SMC=1 (see DPLL_CTRL_1 register) for all relevant calculations.*
IFP	10	rw	**Input filter position** [1] [2] [3] Value contains position or time related information. 0_B TRIGGER_FT and STATE_FT mean time related values, that means the number of time stamp clocks 1_B TRIGGER_FT and STATE_FT mean position related values, that means the number of SUB_INC1 (or SUB_INC2 in the case SMC=1) pulses, respectively

Field	Bits	Type	Description
SNU	15:11	rw	**STATE number** 4) SNU+1 is number of nominal STATE events in HALF_SCALE (1...32). *Note: The number of nominal STATE events is the decimal value plus 1. This value can only be written when (RMO=0 and SMC=0) or DEN=0. To make sure that this signal is not changed during a mode change, RMO=0 means that the status of RMO=0 must be given before and during writing to the register.* *Set SSL=00 before changing this value and set RMO=1 only after FULL_SCALE with SSL>0.* *Note: This register can only be written when DPLL_CTRL_11.STATE_EXT is not set. If DPLL_CTRL_11.STATE_EXT is set, the signal cannot be written, the read value is zero.*
TNU	24:16	rw	**TRIGGER number** 4) TNU+1 is number of nominal *TRIGGER* events in HALF_SCALE (1...512). *Note: The number of nominal TRIGGER events is the decimal value plus 1.This value can only be written when (RMO=1 and SMC=0) or DEN=0.* To make sure that this signal is not changed during a mode change RMO=0 means that the status of RMO=0 must be given before and during writing to the register. Set TSL=00 before changing this value and set RMO=0 only after FULL_SCALE with TSL>0.
AMS	25	rw	**Adapt mode STATE** 2) Use of adaptation information of STATE. 0_B No adaptation information is used for STATE 1_B Immediate adapting mode; the values for physical deviation PD_S of ADT_S[i] are considered to calculate SUB_INC1 pulses in emergency mode (SMC=0), or SUB_INC2 pulses for SMC=1
AMT	26	rw	**Adapt mode TRIGGER** 1) Use of adaptation information of TRIGGER. 0_B No adaptation information for TRIGGER is used 1_B Immediate adapting mode; the values for physical deviation PD of ADT_T[i] are considered to calculate the SUB_INC1 pulses in normal mode and for SMC=1
IDS	27	rw	**Input delay STATE** 2) Use of input delay information transmitted in FT part of the STATE signal. 0_B Delay information is not used 1_B Up to 24 bits of the FT part contain the delay value of the input signal, concerning the corresponding edge

Field	Bits	Type	Description
IDT	28	rw	**Input delay TRIGGER** 1) Use of input delay information transmitted in FT part of the TRIGGER signal. 0_B Delay information is not used 1_B Up to 24 bits of the FT part contain the delay value of the input signal, concerning the corresponding edge
SEN	29	rw	**STATE enable** 0_B STATE signal is not enabled (no signal considered) 1_B STATE signal is enabled
TEN	30	rw	**TRIGGER enable** 0_B TRIGGER signal is not enabled (no signal considered) 1_B TRIGGER signal is enabled
RMO	31	rw	**Reference mode** 1)2) Selection of the relevant input signal for generation of SUB_INC1. Double synchronous mode for SMC=1: Signal *TRIGGER*is used to generate the *SUB_INC1* signals, and *STATE* is used to generate the *SUB_INC2* signals. *Note: For SMC=0: TRIGGER and STATE are prepared to calculate SUB_INC1. The RMO bit gives a decision only, which of them is used. For changing from normal mode to emergency mode at the following TRIGGER slope (according to the RMO value in the shadow register)*[1]*, the PSSC value is calculated by PSSC = PSSM + correction value (forward direction) or PSSC = PSSM - correction value (backward direction) with the correction value = inc_cnt1 - nmb_t. For changing from emergency mode to normal mode at the following STATE slope (according to the RMO value in the shadow register)*[2]*, the PSTC value is calculated by PSTC = PSTM + correction value (forward direction) or PSTC = PSTM - correction value (backward direction) with the correction value = inc_cnt1 - nmb_s. In case no further TRIGGER or STATE events the CPU has to perform the above corrections.* 0_B Normal mode; the signal TRIGGER is used to generate the SUB_INC1 signals 1_B Emergency mode for SMC=0; signal STATE is used to generate the SUB_INC1 signals

1) stored in an independent shadow register for an active TRIGGER event and for DEN = 1.

2) stored in an independent shadow register for an active STATE event and for DEN = 1. T

3) he time between two active STATE or TRIGGER events must be always greater then 23.4 µs; in addition, the TS_CLK and the resolution must be chosen such that for each nominal increment, the time stamps at the beginning and the end of the increment differ at least in the value of 257.

4) For IFP=1, the time between two active TRIGGER or STATE events must be always greater then 2.34 ms, and the value x of MLT, MLS1 or MLS2 must be chosen such that the number of time stamp pulses between two SUB_INC events must be less than 65536. This is fulfilled when x is greater than 256.

28.20.12.2Register DPLL_CTRL_1

DPLL Control Register 1

DPLL_CTRL_1
DPLL Control Register 1 **(028004$_H$)** **Application Reset Value: B000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSL		**SSL**		**SMC**	**TS0_HRT**	**TS0_HRS**	**SYSF**	**SWR**	**LCD**	**SYN_NT**					
rw		rw		rw	rw	rw	rw	rw	rw	rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYN_NS					**PCM2**	**DLM2**	**SGE2**	**PCM1**	**DLM1**	**SGE1**	**PIT**	**COA**	**IDDS**	**DEN**	**DMO**
rw					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DMO	0	rw	**DPLL mode select** [1] [2] 0$_B$ Automatic end mode; if the number of pulses for an increment is reached, no further pulse is generated until the next active TRIGGER/STATE is received; in the case of getting a new active TRIGGER/STATE before the defined number of pulses is reached, the pulse frequency is changed according to the conditions described below (COA) 1$_B$ Continuous mode; in this mode, a difference between the predefined number of pulses and the actual number of generated pulses can influence the pulse frequency by writing a corresponding pulse number into CNT_NUM_1 or CNT_NUM_2, respectively, in RAM region 1b
DEN	1	rw	**DPLL enable** *Note: The bits 31 down to 0 of the DPLL_STATUS register are cleared, when the DPLL is disabled. Some bits of the control registers can be set only when DEN=0. The protected bits in the DPLL_CTRL_1 register cannot be written when simultaneously DEN is set to 1.* 0$_B$ The DPLL is not enabled; disabling the DPLL will result in a reset state of the DPLL_STATUS register, which remains in this state until DEN=1. No DPLL related interrupt will be generated in that case. 1$_B$ The DPLL is enabled

Field	Bits	Type	Description
IDDS	2	rw	**Input direction detection strategy in the case of SMC=0** *Note: This bit can only be written when the DPLL is disabled and fixed to zero, when not needed for an implementation. Independent of the value of IDDS is the direction information for TRIGGER in the case SMC=0 always considered at the moment when the inactive slope appears.* 0_B The input direction is detected comparing the THMI value with the duration between active and inactive slope of TRIGGER 1_B The input direction is detected using TDIR input signal also in the case SMC=0
COA	3	rw	**Correction strategy in automatic end mode (DMO=0)** 1) 2) For SMC=RMO=1: **COA** is used for SUB_INC1 **and** SUB_INC2. 0_B The pulse frequency of the CMU_CLK0 will be used to make up for missing pulses from last increment; the output of the calculated new pulses will start after resetting the FFs in the pulse generation unit. The frequency of CMU_CLK0 should not exceed half the frequency of the system clock (see Adder for generation SUB_INCx by the carry cout with SMC=0). (see also figure Adder for generation of SUB_INCx by the carry cout, **Figure 128**). 1_B Missing pulses of the last increment are distributed evenly to the next increment, calculations are done when the next active input event appears. The number of missing sub-pulses will be determined by the pulse counter difference between the last two active TRIGGER/STATE events, respectively; the FFs in the pulse generation unit are not reset before sending new pulses.
PIT	4	rw	**Plausibility value PVT to next active TRIGGER is time related** 1) 0_B The plausibility value is position related (PVT contains the number of SUB_INC1 pulses) 1_B The plausibility value is time related (the PVT value is to be multiplied with the duration of the last increment DT_T_ACT and divided by 1024)
SGE1	5	rw	**SUB_INC1 generator enable** 1) 2) 0_B The SUB_INC1 generator is not enabled 1_B The SUB_INC1 generator is enabled

Field	Bits	Type	Description
DLM1	6	rw	**Direct Load Mode for SUB_INC1 generation** 1) 2) 0_B The DPLL uses the calculated ADD_IN_CAL value for the SUB_INC1 generation 1_B The ADD_IN_LD value is used for the SUB_INC1 generation and is provided by the CPU; the value remains valid until the CPU writes a new one; the calculated ADD_IN values are stored as ADD_IN_CAL in the RAM at different locations for normal and emergency mode
PCM1	7	rw	**Pulse Correction Mode for SUB_INC1 generation** 1) 2) 3) 0_B The DPLL does not use the correction value stored in MPVAL1 1_B The DPLL uses the correction value stored in MPVAL1 in normal and emergency mode
SGE2	8	rw	**SUB_INC2 generator enable** 2) 0_B The SUB_INC2 generator is not enabled 1_B The SUB_INC2 generator is enabled
DLM2	9	rw	**Direct Load Mode for SUB_INC2 generation** 2) 0_B The DPLL uses the calculated ADD_IN_CAL value for the SUB_INC2 generation 1_B The ADD_IN_LD value is used for the SUB_INC2 generation and is provided by the CPU; the value remains valid until the CPU writes a new one; the calculated ADD_IN values are stored as ADD_IN_CAL in the RAM at different locations for normal and emergency mode
PCM2	10	rw	**Pulse Correction Mode for SUB_INC2 generation** 2) 3) 0_B The DPLL does not use the correction value stored in MPVAL2 1_B The DPLL uses the correction value stored in MPVAL2

Field	Bits	Type	Description
SYN_NS	15:11	rw	**Synchronization number of STATE** Summarized number of virtual increments in HALF_SCALE. Sum of all systematic missing *STATE* events in HALF_SCALE (for SYSF=0) or FULL SCALE (for SYSF=1) ; the SYN_NS missing *STATES* can be divided up to an arbitrary number of blocks. The pattern of events and missing events in FULL_SCALE is shown in RAM region 1c3 as value NS in addition to the adapted values. The number of stored increments in FULL_SCALE must be equal to 2*(SNU+1-SYN_NS) for SYSF=0 or 2*(SNU+1)-SYN_NS for SYSF=1 . This pattern is written by the CPU beginning from a fixed reference point (maybe beginning of the FULL_SCALE region). The relation to the actual increment is established by setting of the profile RAM pointer APS_1C3 in an appropriate relation to the RAM pointer APS of the actual increment by the CPU. This value can only be written when (RMO=0 and SMC=0) or DEN=0. Set SSL=00 before changing this value and set RMO=1 only after FULL_SCALE with SSL>0. To make shure that this signal is not changed during a mode change SMC=0 means that the status of SMC=0 must be given before and during writing to the register. This register can only be written when DPLL_CTRL_11.STATE_EXT is not set. If DPLL_CTRL_11.STATE_EXT is set, the signal cannot be written, the read value is zero.
SYN_NT	21:16	rw	**Synchronization numberof TRIGGER** Summarized number of virtual increments in HALF_SCALE. Sum of all systematic missing *TRIGGER* events in HALF_SCALE; the SYN_NT missing *TRIGGER* can be divided up to an arbitrary number of blocks. The pattern of events and missing events in FULL_SCALE is shown in RAM region 2c as value NT in addition to the adapted values. The number of stored increments in FULL_SCALE must be equal to 2*(TNU-SYN_NT). This pattern is written by the CPU beginning from a fixed reference point (maybe beginning of the FULL_SCALE region). The relation to the actual increment is established by setting of the profile RAM pointer APT_2C in an appropriate relation to the RAM pointer APT of the actual increment by the CPU. This value can only be written when (RMO=1 and SMC=0) or DEN=0. Set TSL=00 before changing this value and set RMO=0 only after FULL_SCALE with TSL>0. To make shure that this signal is not changed during a mode change SMC=0 means that the status of SMC=0 must be given before and during writing to the register.
LCD	22	rw	**Locking condition definition** This bit can only be written when the DPLL is disabled and fixed to zero, when not needed for an implementation. 0_B Locking condition definition is one times missing TRIGGERs, as expected by the profile in HALF_SCALE (one gap) 1_B Locking condition definition is n-1 times missing TRIGGERs, as expected by the profile in HALF_SCALE (one additional tooth)

Field	Bits	Type	Description
SWR	23	rw	**Software reset** Resets all register and internal states of the DPLL. Setting the SWR bit results only in a software reset when the DPLL is not enabled (DEN=0). 0_B No software reset enabled 1_B Software reset enabled
SYSF	24	rw	**SYN_NS for FULL_SCALE** The value SYN_NS does mean the sum of all systematic missing *STATE* events in HALF_SCALE (for SYSF=0) or FULL SCALE (for SYSF=1). This value can only be written when (RMO=0 and SMC=0) or DEN=0. Set SSL=00 before changing this value, and set RMO=1 only after FULL_SCALE with SSL>0. To make sure that this signal is not changed during a mode change, SMC=0 means that the status of SMC=0 must be given before and during writing to the register. 0_B The SYN_NS value is valid for HALF_SCALE 1_B The SYN_NS value is valid for FULL_SCALE
TS0_HRS	25	rw	**Time stamp high resolution STATE** This bit can only be written when the DPLL is disabled. 0_B The resolution of the used DPLL input TBU_TS0 bits is equal to the STATE input time stamp resolution 1_B The STATE input time stamps have an 8 times higher resolution than the TBU_TS0 DPLL input
TS0_HRT	26	rw	**Time stamp high resolution TRIGGER** This bit can only be written when the DPLL is disabled. 0_B The resolution of the used DPLL input TBU_TS0 bits is equal to the TRIGGER input time stamp resolution 1_B The TRIGGER input time stamps have an 8 times higher resolution than the TBU_TS0 input
SMC	27	rw	**Synchronous Motor Control** This bit can only be written when the DPLL is disabled. 0_B TRIGGER and STATE inputs are used for a control different to SMC 1_B The TRIGGER input reflects a combined sensor signal for SMC, and in the case of RMO=1, also STATE reflects a different combined sensor signal

Field	Bits	Type	Description
SSL	29:28	rw	**STATE slope select** Definition of active slope for signal STATE; each active slope is an event defined by SNU. Set by DEN=0 only. If DPLL_STATUS.FSD = '1': Slope sensitive after detection of first STATE input signal. If DPLL_STATUS.FSD = '0': Level sensitive for first STATE input signal edge. *Note: This value can only be written when (RMO=0 and SMC=0) or DEN=0. To make sure that this signal is not changed during a mode change, SMC=0 means that the status of SMC=0 must be given before and during writing to the register.* 00_B FSD=1: No slope of STATE will be used (this value makes only sense in normal mode); FSD=0: No input signal of STATE will be used (this value makes only sense in normal mode). 01_B FSD=1: Low-to-high slope will be used as active slope, only inputs with a signal value of '1' will be considered; FSD=0: "High" input signal level will be used as active slope, only inputs with a signal value of '1' will be considered. 10_B FSD=1: High-to-low slope will be used as active slope, only inputs with a signal value of '0' will be considered; FSD=0: "Low" input signal level will be used as active slope, only inputs with a signal value of '0' will be considered. 11_B FSD=1: Both slopes will be used as active slopes; FSD=0: Both input signal levels will be used as active slopes.

Field	Bits	Type	Description
TSL	31:30	rw	**TRIGGER slope select** Definition of active slope for signal TRIGGER each active slope is an event defined by TNU. Set by DEN=0 only. If DPLL_STATUS.FTD = '1' "slope sensitive after detection of first TRIGGER input signal" If DPLL_STATUS.FTD = '0' "level sensitive for first TRIGGER input signal edge" *Note: This value can only be written when (RMO=1 and SMC=0) or DEN=0. To make sure that this signal is not changed during a mode change, SMC=0 means that the status of SMC=0 must be given before and during writing to the register.* 00_B FTD=1: No slope of TRIGGER will be used; this value makes only sense in emergency mode; FTD=0: No input signal of TRIGGER will be used; this value makes only sense in normal mode 01_B FTD=1: Low high slope will be used as active slope, only inputs with a signal value of "1" will be considered; FTD=0: "high" input signal level will be used as active slope, only inputs with a signal value of "1" will be considered 10_B FTD=1: High low slope will be used as active slope, only inputs with a signal value of "0" will be considered; FTD=0: "low" input signal level will be used as active slope, only inputs with a signal value of "0" will be considered 11_B FTD=1: Both slopes will be used as active slopes; FTD=0: Both input signal levels will be used as active slopes

1) Stored in an independent shadow register for a valid TRIGGER event and for DEN = 1.

2) Stored in an independent shadow register for a valid STATE event and for DEN = 1.

3) Bit is cleared, when transmitted to shadow register.

28.20.12.3Register DPLL_CTRL_2

DPLL Control Register 2

DPLL_CTRL_2
DPLL Control Register 2 **(028008_H)** **Application Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**WAD7**	**WAD6**	**WAD5**	**WAD4**	**WAD3**	**WAD2**	**WAD1**	**WAD0**
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN7	**AEN6**	**AEN5**	**AEN4**	**AEN3**	**AEN2**	**AEN1**	**AEN0**	**0**							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
AENx (x=0-7)	x+8	rw	**ACTION_x enable** *Note: This bit can be written only if the correspondent WADx bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1).* *Note: For WADi =1, only the corresponding AENi bits are writable. The AENi bits remain unchanged when the corresponding WADi=0.* 0_B The corresponding action is not enabled 1_B The corresponding action is enabled
WADx (x=0-7)	x+16	rw	**Write control bit of Action_x** For WADx =1, only the corresponding AENx bits are writable. The AENx bits remain unchanged when the corresponding WADx=0. 0_B The corresponding AENx bit is not writeable 1_B The corresponding AENx bit is writeable
0	7:0, 31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.4Register DPLL_CTRL_3

DPLL Control Register 3

Note: For all AENi: This bit can be written only if the correspondent WADi Bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1).

Note: For WADi =1 only the corresponding AENi bits are writable. The AENi bits remain unchanged when the corresponding WADi=0.

DPLL_CTRL_3

DPLL Control Register 3 **($02800C_H$)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**WAD15**	**WAD14**	**WAD13**	**WAD12**	**WAD11**	**WAD10**	**WAD9**	**WAD8**
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN15	**AEN14**	**AEN13**	**AEN12**	**AEN11**	**AEN10**	**AEN9**	**AEN8**	**0**							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
AENx (x=8-15)	x	rw	**ACTION_x enable** This bit can be written only if the correspondent WADx bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1). 0_B The corresponding action is not enabled 1_B The corresponding action is enabled
WADx (x=8-15)	x+8	rw	**Write control bit of Action_x** For WADx = 1, only the corresponding AENx bits are writable. The AENx bits remain unchanged when the corresponding WADx = 0. 0_B The corresponding AENx bit is not writeable 1_B The corresponding AENx bit is writeable
0	7:0, 31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.5Register DPLL_CTRL_4

DPLL Control Register 4

Note: For all AENi: This bit can be written only if the correspondent WADi Bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1).

Note: For WADi=1, only the corresponding AENi bits are writable. The AENi bits remain unchanged when the corresponding WADi=0.

DPLL_CTRL_4

DPLL Control Register 4 **(028010_H)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**WAD23**	**WAD22**	**WAD21**	**WAD20**	**WAD19**	**WAD18**	**WAD17**	**WAD16**
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN23	**AEN22**	**AEN21**	**AEN20**	**AEN19**	**AEN18**	**AEN17**	**AEN16**	**0**							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
AENx (x=16-23)	x-8	rw	**ACTION_x enable** This bit can be written only if the correspondent WADx bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1). 0_B The corresponding action is not enabled 1_B The corresponding action is enabled
WADx (x=16-23)	x	rw	**Write control bit of Action_x** For WADx = 1, only the corresponding AENx bits are writable. The AENx bits remain unchanged when the corresponding WADx = 0. 0_B The corresponding AENx bit is not writeable 1_B The corresponding AENx bit is writeable
0	7:0, 31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.6Register DPLL_CTRL_5

DPLL Control Register 5

Note: *For all AENi: This bit can be written only if the correspondent WADi bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1).*

Note: *For WADi=1, only the corresponding AENi bits are writable. The AENi bits remain unchanged when the corresponding WADi=0.*

DPLL_CTRL_5

DPLL Control Register 5 **(028014_H)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**WAD31**	**WAD30**	**WAD29**	**WAD28**	**WAD27**	**WAD26**	**WAD25**	**WAD24**
r								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN31	**AEN30**	**AEN29**	**AEN28**	**AEN27**	**AEN26**	**AEN25**	**AEN24**	**0**							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
AENx (x=24-31)	x-16	rw	**ACTION_x enable** This bit can be written only if the correspondent WADx bit is set in the same access. It can be set for debug purposes by CPU also, when DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN=1). 0_B The corresponding action is not enabled 1_B The corresponding action is enabled
WADx (x=24-31)	x-8	rw	**Write control bit of Action_x** For WADx = 1, only the corresponding AENx bits are writable. The AENx bits remain unchanged when the corresponding WADx = 0. 0_B The corresponding AENx bit is not writeable 1_B The corresponding AENx bit is writeable
0	7:0, 31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.7Register DPLL_ACT_STA

DPLL ACTION Status Register with Connected Shadow Register

DPLL_ACT_STA
DPLL ACTION Status Register with Connected Shadow Register(028018_H) Application Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACT_N															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACT_N															
rw															

Field	Bits	Type	Description
ACT_N	31:0	rw	**New output data values concerning to action i provided** ACT_N[i] is: • Set (for AENi=1 and a new valid PMTR), that means when new action data are to be calculated for the correspondent action. After each calculation of the new actions values, the ACT_N[i] bit updates the corresponding bit in the connected shadow register. The status of the ACT_N[i] bits in the shadow register is reflected by the corresponding DPLL output signal ACT_V (valid bit). • Reset together with the corresponding shadow register bit for AENi=0. • Reset without the corresponding shadow register bit when the calculated event is in the past (the shadow register bit is set, when it was not set before in that case). • The corresponding shadow register bit is reset, when new PMTR data are written or when the provided action data are read (blocking read). • Writeable for debugging purposes together with the corresponding shadow register when DEN=0. *Note: These bits can only be written for test purposes when the DPLL is disabled.* 00000000$_H$No new output data available after a recent PMT request or actual event value is in the past or invalid 00000001$_H$New PMTR data received or calculation is to be precised by taking into account new TRIGGER or STATE values

28.20.12.8Register DPLL_OSW

DPLL Offset and Switch Old/New Address Register

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is not set. If DPLL_CTRL_11.STATE_EXT is set any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_OSW
DPLL Offset and Switch Old/New Address Register(02801C$_H$) **Application Reset Value: 0000 0200$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						**OSS**		**0**						**SWON_T**	**SWON_S**
r						rw		r						r	r

Field	Bits	Type	Description
SWON_S	0	r	**Switch of new STATE** Switch bit for LSB address of STATE. This bit is changed for each write access to TS_S/TS_S_OLD. Using this unchanged address bit SWON_S for any access to TS_S results always in an access to TS_S_OLD. For writing to this address the former old (TS_S_OLD_old) value is overwritten by the new one while the SWON_S bit changes. Thus the former new one is now the old one and the next access is after changing SWON_S directed to this place. Therefore write to TS_S first and after that immediately to FTV_S and PSSM, always before a new TS_S value is to be written. After writing TS_S, FTV_S and PSSM in this order the address pointer AP with LSB(AP)=SWON_S shows for the corresponding address to TS_S_OLD, FTV_S and PSSM while LSB(AP)=/SWON_S results in an access to TS_S, FTV_S_old and PSSM_OLD respectively. The value can be read only. This bit is reset when disabling the DPLL (DEN=0).
SWON_T	1	r	**Switch of new TRIGGER** Switch bit for LSB address of TRIGGER. This bit is changed for each write access to TS_T/TS_T_OLD. Using this unchanged address bit SWON_T for any access to TS_T results always in an access to TS_T_OLD. For writing to this address the former old (TS_T_OLD_old) value is overwritten by the new one while the SWON_T bit changes. Thus the former new one is now the old one and the next access is after changing SWON_T directed to this place. Therefore write to TS_T first and after that immediately to FTV_T and PSTM, always before a new TS_T value is to be written. After writing TS_T, FTV_T and PSTM in this order the address pointer AP with LSB(AP)=SWON_T shows for the corresponding address to TS_T_OLD, FTV_T and PSTM while LSB(AP)=/SWON_T results in an access to TS_T, FTV_T_old and PSTM_OLD respectively. The value can be read only. This bit is reset when disabling the DPLL (DEN=0).
OSS	9:8	rw	**Offset size of RAM region 2** At least 128 and at most 1024 values can be stored in each of the RAM 2 regions a to d accordingly. The value can be set only for DEN=0. The change of the OSS value results in an automatic change of the offset values in the DPLL_AOSV_2 register. This value can only be written when the DPLL is disabled. 00_B Offset size 128 of RAM region 2 01_B Offset size 256 of RAM region 2 10_B Offset size 512 of RAM region 2 11_B Offset size 1024 of RAM region 2
0	7:2, 31:10	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.9Register DPLL_AOSV_2

DPLL Address Offset Register of RAM 2 Regions

DPLL_AOSV_2

DPLL Address Offset Register of RAM 2 Regions(028020$_H$) **Application Reset Value: 1810 0800$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AOSV_2D								**AOSV_2C**							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AOSV_2B								**AOSV_2A**							
r								r							

Field	Bits	Type	Description

AOSV_2A	7:0	r	**Address offset value of the RAM 2A region**
			The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2a. When the APT value is added to this start address, the current RAM cell RDT_Tx is addressed.
			Value is set automatically when OSS in the PPLL_OSW register is set:
			OSS=0x0: AOSV_2A= 0x00
			OSS=0x1: AOSV_2A= 0x00
			OSS=0x2: AOSV_2A= 0x00
			OSS=0x3: AOSV_2A= 0x00

AOSV_2B	15:8	r	**Address offset value of the RAM 2B region**
			The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2b. When the APT value is added to this start address, the current RAM cell TSF_Tx is addressed.
			Value is set automatically when OSS in the PPLL_OSW register is set:
			OSS=0x0: AOSV_2B= 0x02
			OSS=0x1: AOSV_2B= 0x04
			OSS=0x2: AOSV_2B= 0x08
			OSS=0x3: AOSV_2B= 0x10

AOSV_2C	23:16	r	**Address offset value of the RAM 2C region**
			The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2c. When the APT value is added to this start address, the current RAM cell ADT_Tx is addressed.
			Value is set automatically when OSS in the PPLL_OSW register is set:
			OSS=0x0: AOSV_2C= 0x04
			OSS=0x1: AOSV_2C= 0x08
			OSS=0x2: AOSV_2C= 0x10
			OSS=0x3: AOSV_2C= 0x20

Field	Bits	Type	Description
AOSV_2D	31:24	r	**Address offset value of the RAM 2D region** The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2d. When the APT value is added to this start address, the current RAM cell DT_Tx is addressed. Value is set automatically when OSS in the PPLL_OSW register is set: OSS=0x0: AOSV_2D= 0x06 OSS=0x1: AOSV_2D= 0x0C OSS=0x2: AOSV_2D= 0x18 OSS=0x3: AOSV_2D= 0x30 The offset values are needed to support a scalable RAM size of region 2 from 1,5 Kbytes to 12 Kbytes. The values above must be in correlation with the offset size defined in the OSW register. All offset values are set automatically in accordance to the OSS value in the DPLL_OSW register. This value can be set only for DEN=0.

28.20.12.10Register DPLL_APT

DPLL Actual RAM Pointer Address for TRIGGER

DPLL_APT

DPLL Actual RAM Pointer Address for TRIGGER (028024$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**APT_2B**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APT_2B		**WAPT _2B**	**0**	**APT**										**WAPT**	**0**
rw		rw	r	rw										rw	r

Field	Bits	Type	Description
WAPT	1	rw	**Write bit for address pointer APT** Read as zero. 0_B The APT is not writeable 1_B The APT is writeable

Field	Bits	Type	Description
APT	11:2	rw	**Address pointer TRIGGER** Actual RAM pointer address value offset for DT_T[i] and RDT_T[i] in FULL_SCALE for 2*(TNU+1-SYN_NT) TRIGGER events. This pointer is used for the RAM region 2 subsections 2a and 2d. The pointer APT is incremented for each active *TRIGGER* event (simultaneously with APT_2B, APT_2C) for DIR1=0. For DIR1=1 the APT is decremented. The APT offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region The APT pointer value is directed to the RAM position, in which the data values are to be written, which corresponds to the last increment. The APT value is not to be changed, when the direction (shown by DIR1) changes, because it points always to a storage place after the considered increment. Changing of DIR1 takes place always after an active *TRIGGER* event and the resulting increment/decrement. This value can only be written when the WAPT bit is set.
WAPT_2B	13	rw	**Write bit for address pointer APT_2B** Read as zero. 0_B The APT_2B is not writeable 1_B The APT_2B is writeable
APT_2B	23:14	rw	**Address pointer TRIGGER for RAM region 2b** Actual RAM pointer address value for TSF_T[i] Actual RAM pointer address of *TRIGGER* events in FULL_SCALE for 2*(TNU+1) *TRIGGER* periods; this pointer is used for the RAM region 2b. The RAM pointer is initially set to zero. For SYT=1: The pointer APT_2B is incremented by SYN_T_OLD for each active *TRIGGER* event (simultaneously with APT and APT_2C) for DIR1=0 when an active *TRIGGER* input appears. For DIR1=1 (backwards) the APT is decremented by SYN_T_OLD. For SYT=0: APT_2B is incremented or decremented by 1. In addition when the APT_2C value is written by the CPU - in order to synchronize the DPLL- with the next active *TRIGGER* event the APT_2B_EXT value is added/subtracted (while APT_2B_STATUS is one; see DPLL_APT_SYNC register at **Section 28.20.12.24**). This value can only be written when the WAPT_2B bit is set.
0	0, 12, 31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.11Register DPLL_APS

DPLL Actual RAM Pointer Address for STATE

DPLL_APS

DPLL Actual RAM Pointer Address for STATE (028028$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												**APS_1C2**			
r												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APS_1C2		**WAPS _1C2**	**0**					**APS**						**WAPS**	**0**
rw		rw	r					rw						rw	r

Field	Bits	Type	Description
WAPS	1	rw	**Write bit for address pointer APS** Read as zero 0_B The APS is not writeable 1_B The APS is writeable
APS	7:2	rw	**Address pointer STATE** Actual RAM pointer address value for DT_S[i] and RDT_S[i] Actual RAM pointer and synchronization position/value of *STATE* events in FULL_SCALE for up to 64 *STATE* events but limited to 2*(SNU+1-SYN_NS) in normal and emergency mode for SYSF=0 or to 2*(SNU+1)-SYN_NS for SYSF=1 respectively; this pointer is used for the RAM region 1c1 and 1c4. APS is incremented (decremented) by one for each active *STATE* event and DIR2=0 DIR2=1). The APS offset value is added in the above shown bit position with the subsection offset of the RAM region. The APS pointer value is directed to the RAM position, in which the data values are to be written, which correspond to the last increment. The APS value is not to be changed, when the direction (shown by DIR2) changes, because it points always to a storage place after the considered increment. Changing of DIR2 takes place always after an active *STATE* event and the resulting increment/decrement. This value can only be written when the WAPS bit is set.
WAPS_1C2	13	rw	**Write bit for address pointer APS_1C2** Read as zero 0_B The APS_1C2 is not writeable 1_B The APS_1C2 is writeable

Field	Bits	Type	Description
APS_1C2	19:14	rw	**Address pointer STATE for RAM region 1c2** Actual RAM pointer address value for TSF_S[i]. Initial value: zero (0x00). Actual RAM pointer and synchronization position/value of *STATE* events in FULL_SCALE for up to 64 *STATE* events but limited to 2*(SNU+1) in normal and emergency mode; this pointer is used for the RAM region 1c2. For SYS=1: APS_1C2 is incremented (decremented) by SYN_S_OLD for each active *STATE* event and DIR2=0 (DIR2=1). For SYS=0: APT_1c2 is incremented or decremented by 1 respectively. The APS_1C2 offset value is added in the above shown bit position with the subsection offset of the RAM region. In addition when the APS_1C3 value is written by the CPU - in order to synchronize the DPLL- with the next active *STATE* event the APS_1C2_EXT value is added/subtracted (while APS_1C2_STATUS is one; see DPLL_APT_SYNC register at **Section 28.20.12.25**). This value can only be written when the WAPS_1C2 bit is set
0	0, 12:8, 31:20	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.12Register DPLL_APT_2C

DPLL Actual RAM Pointer for Region 2C

Note: The APT_2C pointer values are directed to the RAM position of the profile element in RAM region 2c, which correspond to the current increment. For DIR1=0 (DIR1=1) the pointers APT_2C_x are incremented (decremented) by one simultaneously with APT. For SMC=0 the change of DIR1 takes place always after an active TRIGGER event (by evaluation of the inactive slope) and the resulting increment/decrement. In the case SMC=1 the direction change is known before the input event is processed. The correction of the APT_2C pointer differs: for SMC=0 correct 4 times and for SMC=1 correct only 2 times. The APT_2C_x offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region.

DPLL_APT_2C

DPLL Actual RAM Pointer for Region 2C (02802C$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				**APT_2C**										**0**	
r				rw										r	

Field	Bits	Type	Description
APT_2C	11:2	rw	**Address pointer TRIGGER for RAM region 2c and Actual RAM pointer address value for ADT_T[i]** Actual RAM pointer address value of TRIGGER adapt events in FULL_SCALE for 2*(TNU+1-SYN_NT) TRIGGER periods depending on the size of the used RAM 2; this pointer is used for the RAM region 2 for the subsection 2c only. The RAM pointer is initially set to zero. The APT_2C value is set by the CPU when the synchronization condition was detected. Within the RAM region 2c initially the conditions for synchronization gaps and adapted values are stored by the CPU.
0	1:0, 31:12	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.13Register DPLL_APS_1C3

DPLL Actual RAM Pointer for RAM Region 1C3

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is not set. If DPLL_CTRL_11.STATE_EXT is set any read/write access to this register will return AEI_STATUS = 0b10.

Note: The APS_1C3 pointer value is directed to the RAM position of the profile element in RAM region 1c2, which corresponds to the current increment. When changing the direction DIR1 or DIR2 respectively, this is always known before an active STATE event is processed. This is because of the pattern recognition in SPE (for PMSM) or because of the direction change recognition by TRIGGER. This direction change results in an automatic increment (forwards) or decrement (backwards) when the input event occurs in addition with a 2 times correction.

The APS_1C3_x offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region.

DPLL_APS_1C3

DPLL Actual RAM Pointer for RAM Region 1C3 (028030$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**APS_1C3**						**0**	
r								rw						r	

Field	Bits	Type	Description
APS_1C3	7:2	rw	**Address pointer STATE for RAM region 1c3** Actual RAM pointer address value for ADT_S[i]. Initial value: zero (0x00). Actual RAM pointer and synchronization position/value of *STATE* events in FULL_SCALE for up to 64 *STATE* events but limited to 2*(SNU+1-SYN_NS) in normal and emergency mode for SYSF=0 or to 2*(SNU+1)-SYN_NS for SYSF=1 respectively; this pointer is used for the RAM region 1c3. The RAM pointer is set by the CPU accordingly, when the synchronization condition was detected.
0	1:0, 31:8	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.14Register DPLL_NUTC

DPLL Number of Recent TRIGGER Events Used for Calculations

Note: DPLL Number of recent TRIGGER events used for calculations (mod 2(TNU +1-SYN_NT)).*

DPLL_NUTC

DPLL Number of Recent TRIGGER Events Used for Calculations(028034$_H$)　　Application Reset Value: 0001 2001$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WVTN	**WSYN**	**WNUT**	**0**				**VTN**						**SYN_T_OLD**		
rw	rw	rw	r				rw						rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYN_T			**0**		**FST**	**NUTE**									
rw			r		rw	rw									

Field	Bits	Type	Description
NUTE	9:0	rw	**Number of recent TRIGGER events used for SUB_INC1 and action calculations modulo 2*(TNUmax+1)** Number of recent TRIGGER events used for SUB_INC1 and action calculations modulo 2*(TNUmax+1). NUTE: number of last nominal increments to be considered for the calculations. No gap is considered in that case for this value, but in the VTN value (see below): This value is set by the CPU, but reset automatically to '1' by a change of direction or loss of LOCK. Each other value can be set by the CPU, maybe Full_SCALE, HALF_SCALE or parts of them. For FULL_SCALE set NUTE= 2*(TNU +1) and for HALF_SCALE NUTE= TNU +1. The relation values QDT_Tx are calculated using NUTE values in the past with its maximum value of 2*(TNU +1). The value zero (in combination with the value FST=1) does mean 2^{11} values in the past. *Note: To prevent that inconsistencies between internal pointer in which NUTE is used and the case decision of different prediction method's for prediction of the next event and PMT(position minus time) occur, the NUTE value is stored internally at that point of time when the internal pointers are calculated for the next event cycle.* *Note: This value can only be written when the WNUT bit is set.* 000_H The NUTE value is less then FULL_SCALE 001_H The NUTE value is equal to FULL_SCALE *This value is set by the CPU, but reset automatically to "0" by a change of direction orloss of LOCK.*
FST	10	rw	**FULL_SCALE of TRIGGER** This value is to be set, when NUTE is set to FULL_SCALE. This value can only be written when the WNUT bit is set.
SYN_T	15:13	rw	**Number of real and virtual events to be considered for the current increment** This value reflects the NT value of the last valid increment, stored in ADT_T[i]; to be updated after all calculations in step 17 of **Table 28.20.8.6.6**. This value can only be written when the WSYN bit in this register is set.
SYN_T_OLD	18:16	rw	**Number of real and virtual events to be considered for the last increment** This value reflects the NT value of the last but one valid increment, stored in ADT_T[i]; is updated automatically when writing SYN_T. This value is updated by the SYN_T value when the WSYN bit in this register is set.

Field	Bits	Type	Description
VTN	24:19	rw	**Virtual TRIGGER number** Number of virtual increments in the current NUTE region This value reflects the number of virtual increments in the current NUTE region; for NUTE=1 this value is zero, when the CPU sets NUTE to a value > 1 , it must also set VTN to the correspondent value; for NUTE is set to FULL_SCALE including NUTE=zero (2^{11} modulo 2^{11}) the VTN is to be set to 2* SYN_NT. The VTN value is subtracted from the NUTE value in order to get the corresponding APT value for the past; the VTN value is not used for the APT_2B pointer. VTN is to be updated by the CPU when a new gap is to be considered for NUTE or a gap is leaving the NUTE region; for this purpose the TINT values in the profile can be used to generate an interrupt for the CPU at the corresponding positions; no further update of VTN is necessary when NUTE is set to FULL_SCALE This value can only be written when the WVTN bit is set.
WNUT	29	rw	**Write control bit for NUTE and FST** Read as zero 0_B The NUTE value is not writeable 1_B The NUTE value is writeable
WSYN	30	rw	**Write control bit for SYN_T and SYN_T_OLD** Read as zero 0_B The SYN_T value is not writeable 1_B The SYN_T value is writeable
WVTN	31	rw	**Write control bit for VTN** Read as zero 0_B The VTN value is not writeable 1_B The VTN value is writeable
0	12:11, 28:25	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.15Register DPLL_NUSC

DPLL Number of Recent STATE Events Used for Calculations

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is not set. If DPLL_CTRL_11.STATE_EXT is set any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_NUSC

DPLL Number of Recent STATE Events Used for Calculations(028038$_H$) Application Reset Value: 0000 2081$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WVSN	**WSYN**	**WNUS**	**0**				**VSN**						**SYN_S_OLD**		
rw	rw	rw	r				rw						rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYN_S_OLD			**SYN_S**						**FSS**	**NUSE**					
rw			rw						rw	rw					

Field	Bits	Type	Description
NUSE	5:0	rw	**Number of recent STATE events used for SUB_INCx calculations modulo 2*(SNUmax+1)** No gap is considered in that case for this value, but in the VSN value (see below): This register is set by the CPU but reset automatically to “1” by a change of direction or loss of LOCK. Each other value can be set by the CPU, maybe Full_SCALE, HALF_SCALE or parts of them. The relation values QDT_Sx are calculated using NUSE values in the past with its maximum value of 2*SNU+1. This value can only be written when the WNUS bit is set. *Note: To prevent that inconsistencies between internal pointer in which NUSE is used and the case decision of different prediction method's for prediction of the next event and PMT(position minus time) occur, the NUSE value is stored internally at that point of time when the internal pointers are calculated for the next event cycle.*
FSS	6	rw	**FULL_SCALE of STATE** This value is to be set, when NUSE is set to FULL_SCALE. This value is set by the CPU, but reset automatically to '0' by a change of direction or loss of LOCK. *Note: This value can only be written when the WNUS bit is set.* 0_B The NUSE value is less then FULL_SCALE 1_B The NUSE value is equal to FULL_SCALE
SYN_S	12:7	rw	**Number of real and virtual events to be considered for the current increment** This value reflects the NS value of the last valid increment, stored in ADT_S[i]; to be updated after all calculations in step 37 of **Table 28.20.8.6.6**. This value can only be written when the WSYN bit in this register is set.
SYN_S_OLD	18:13	rw	**Number of real and virtual events to be considered for the last increment** This value reflects the NS value of the last but one valid increment, stored in ADT_S[i]; is updated automatically when writing SYN_S. This value is updated by the SYN_S value when the WSYN bit in this register is set.

Field	Bits	Type	Description
VSN	24:19	rw	**Virtual STATE number** Number of virtual state increments in the current NUSE region. This value reflects the number of virtual increments in the current NUSE region; for NUSE=1 this value is zero, when the CPU sets NUSE to a value > 1 or zero(2^7 modulo 2^7) , it must also set VSN to the correspondent value; the VSN value is subtracted from the NUSE value in order to get the corresponding APS value for the past; the VSN value is not used for the APS_1C2 pointer. VSN is to be updated by the CPU when a new gap is to be considered for NUSE or a gap is leaving the NUSE region; for this purpose the SASI interrupt can be used; no further update of VSN is necessary when NUSE is set to FULL_SCALE This value can only be written when the WVSN bit is set.
WNUS	29	rw	**Write control bit for NUSE** Read as zero. 0= the NUSE value is not writeable 1= the NUSE value is writeable
WSYN	30	rw	**Write control bit for SYN_S and SYN_S_OLD** Read as zero 0= the SYN_S value is not writeable 1= the SYN_S value is writeable
WVSN	31	rw	**Write control bit for VS** Read as zero 0_B The VSN value is not writeable 1_B The VSN value is writeable
0	28:25	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.16Register DPLL_NTI_CNT

DPLL Number of Active TRIGGER Events to Interrupt

DPLL_NTI_CNT
DPLL Number of Active TRIGGER Events to Interrupt($02803C_H$) **Application Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						**NTI_CNT**									
r						rw									

Field	Bits	Type	Description
NTI_CNT	9:0	rw	**Number of TRIGGERs to interrupt** Number of active TRIGGER events to the next DPLL_CDTI interrupt. This value shows the remaining *TRIGGER* events until an active TRIGGER slope results in a DPLL_CDTI interrupt; the value is to be count down for each active *TRIGGER* event.
0	31:10	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.17Register DPLL_IRQ_NOTIFY

DPLL Interrupt Notification Register

Note: All bits in the DPLL_IRQ_NOTIFY register are set permanently until writing a one bit value is performed to the corresponding bit.

DPLL_IRQ_NOTIFY

DPLL Interrupt Notification Register **(028040_H)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				**DCGI**	**SORI**	**TORI**	**CDSI**	**CDTI**	**TE4I**	**TE3I**	**TE2I**	**TE1I**	**TE0I**	**LL2I**	**GL2I**
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EI	**LL1I**	**GL1I**	**W1I**	**W2I**	**PWI**	**TASI**	**SASI**	**MTI**	**MSI**	**TISI**	**SISI**	**TAXI**	**TINI**	**PEI**	**PDI**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PDI	0	rw	**DPLL disable interrupt; announces the switch off of the DEN bit** This event is combined with the PEI interrupt to the common PDI + PEI interrupt line number 1. 0_B The DPLL disable interrupt is not requested 1_B The DPLL disable interrupt is requested
PEI	1	rw	**DPLL enable interrupt** Announces the switch on of the DEN bit. This event is combined with the PDI interrupt to the common PDI + PEI interrupt line number 1. 0_B The DPLL enable interrupt is not requested 1_B The DPLL enable interrupt is requested
TINI	2	rw	**TRIGGER minimum hold time violation interrupt (dt <= THMI > 0)** 0_B No violation of minimum hold time of TRIGGER is detected 1_B A violation of minimum hold time of TRIGGER is detected
TAXI	3	rw	**TRIGGER maximum hold time violation interrupt (dt > THMA > 0)** 0_B No violation of maximum hold time of TRIGGER is detected 1_B A violation of maximum hold time of TRIGGER is detected

Field	Bits	Type	Description
SISI	4	rw	**STATE inactive slope interrupt** 0_B No inactive slope of STATE is detected 1_B An inactive slope of STATE is detected
TISI	5	rw	**TRIGGER inactive slope interrupt** The TISI bit is only set for an inactive slope when the preceding active slope was accepted. In the case of suppression of the last active slope by the plausibility check the next inactive slope is to be ignored. No set of TISI is performed in this case. 0_B No inactive slope of TRIGGER is detected 1_B An inactive slope of TRIGGER is detected
MSI	6	rw	**Missing STATE interrupt** 0_B The missing STATE interrupt is not requested 1_B The missing STATE interrupt is requested
MTI	7	rw	**Missing TRIGGER interrupt** 0_B The missing TRIGGER interrupt is not requested 1_B The missing TRIGGER interrupt is requested
SASI	8	rw	**STATE active slope interrupt** 0_B No active slope of STATE is detected 1_B An active slope of STATE is detected
TASI	9	rw	**TRIGGER active slope interrupt** 0_B No active slope of TRIGGER is detected 1_B An active slope of TRIGGER is detected
PWI	10	rw	**Plausibility window (PVT) violation interrupt of TRIGGER** 0_B The plausibility window is not violated 1_B The plausibility window is violated
W2I	11	rw	**RAM write access to RAM region 2 interrupt** 0_B The RAM write access interrupt is not requested 1_B The RAM write access interrupt is requested
W1I	12	rw	**Write access to RAM region 1b or 1c interrupt** 0_B The RAM write access interrupt is not requested 1_B The RAM write access interrupt is requested
GL1I	13	rw	**Get of lock interrupt, for SUB_INC1** 0_B The lock getting interrupt is not requested 1_B The lock getting interrupt is requested
LL1I	14	rw	**Loss of lock interrupt for SUB_INC1** 0_B The lock loss interrupt is not requested 1_B The lock loss interrupt is requested

Field	Bits	Type	Description
EI	15	rw	**Error interrupt (see status register bit 31)** 0_B The error interrupt is not requested 1_B The error interrupt is requested
GL2I	16	rw	**Get of lock interrupt, for SUB_INC2** 0_B The lock getting interrupt is not requested 1_B The lock getting interrupt is requested
LL2I	17	rw	**Loss of lock interrupt for SUB_INC2** 0_B The lock loss interrupt is not requested 1_B The lock loss interrupt is requested
TE0I	18	rw	**TRIGGER event interrupt 0** 0_B No interrupt on TRIGGER event 0 requested 1_B Interrupt on TRIGGER event 0 requested
TE1I	19	rw	**TRIGGER event interrupt 1** 0_B No interrupt on TRIGGER event 1 requested 1_B Interrupt on TRIGGER event 1 requested
TE2I	20	rw	**TRIGGER event interrupt 2** 0_B No interrupt on TRIGGER event 2 requested 1_B Interrupt on TRIGGER event 2 requested
TE3I	21	rw	**TRIGGER event interrupt 3** 0_B No interrupt on TRIGGER event 3 requested 1_B Interrupt on TRIGGER event 3 requested
TE4I	22	rw	**TRIGGER event interrupt 4** 0_B No interrupt on TRIGGER event 4 requested 1_B Interrupt on TRIGGER event 4 requested
CDTI	23	rw	**Calculation of TRIGGER duration done, only while NTI_CNT is zero** 0_B No interrupt on calculated TRIGGER duration requested or NTI_CNT is not zero 1_B Interrupt on calculated TRIGGER duration requested while NTI_CNT is zero
CDSI	24	rw	**Calculation of STATE duration done** 0_B No interrupt on calculated STATE duration requested 1_B Interrupt on calculated STATE duration requested

Field	Bits	Type	Description
TORI	25	rw	**TRIGGER out of range interrupt** 0_B TRIGGER is not out of range 1_B TRIGGER is out of range, the TOR bit in the DPLL_STATUS register is set to 1
SORI	26	rw	**STATE out of range** The interrupt occurs at line number 0. 0_B STATE is not out of range 1_B STATE is out of range, the SOR bit in the DPLL_STATUS register is set to 1
DCGI	27	rw	**Direction change interrupt** 0_B No direction change of TRIGGER is detected 1_B Direction change of TRIGGER is detected
0	31:28	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.18Register DPLL_IRQ_EN

DPLL Interrupt Enable Register

DPLL_IRQ_EN

DPLL Interrupt Enable Register **(028044_H)** **Application Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				**DCGI_IRQ_EN**	**SORI_IRQ_EN**	**TORI_IRQ_EN**	**CDSI_IRQ_EN**	**CDTI_IRQ_EN**	**TE4I_IRQ_EN**	**TE3I_IRQ_EN**	**TE2I_IRQ_EN**	**TE1I_IRQ_EN**	**TE0I_IRQ_EN**	**LL2I_IRQ_EN**	**GL2I_IRQ_EN**
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EI_IRQ_EN	**LL1I_IRQ_EN**	**GL1I_IRQ_EN**	**W1I_IRQ_EN**	**W2I_IRQ_EN**	**PWI_IRQ_EN**	**TASI_IRQ_EN**	**SASI_IRQ_EN**	**MTI_IRQ_EN**	**MSI_IRQ_EN**	**TISI_IRQ_EN**	**SISI_IRQ_EN**	**TAXI_IRQ_EN**	**TINI_IRQ_EN**	**PEI_IRQ_EN**	**PDI_IRQ_EN**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PDI_IRQ_EN	0	rw	**DPLL disable interrupt enable, when switch-off of the DEN bit** 0_B The DPLL disable interrupt is not enabled 1_B The DPLL disable interrupt is enabled
PEI_IRQ_EN	1	rw	**DPLL enable interrupt enable, when switch-on of the DEN bit** 0_B The DPLL enable interrupt is not enabled 1_B The DPLL enable interrupt is enabled
TINI_IRQ_EN	2	rw	**TRIGGER minimum hold time violation interrupt enable** 0_B Interrupt on minimum hold time violation of TRIGGER is not enabled 1_B Interrupt on minimum hold time violation of TRIGGER is enabled

Field	Bits	Type	Description
TAXI_IRQ_EN	3	rw	**TRIGGER maximum hold time violation interrupt enable** 0_B Interrupt on maximum hold time violation of TRIGGER is not enabled 1_B Interrupt on maximum hold time violation of TRIGGER is enabled
SISI_IRQ_EN	4	rw	**STATE inactive slope interrupt enable** 0_B Interrupt at the inactive slope of STATE is not enabled 1_B Interrupt at the inactive slope of STATE is enabled
TISI_IRQ_EN	5	rw	**TRIGGER inactive slope interrupt enable** 0_B Interrupt at the inactive slope of TRIGGER is not enabled 1_B Interrupt at the inactive slope of TRIGGER is enabled
MSI_IRQ_EN	6	rw	**Missing STATE interrupt enable** 0_B The missing STATE interrupt is not enabled 1_B The missing STATE interrupt is enabled
MTI_IRQ_EN	7	rw	**Missing TRIGGER interrupt enable** 0_B The missing TRIGGER interrupt is not enabled 1_B The missing TRIGGER interrupt is enabled
SASI_IRQ_EN	8	rw	**STATE active slope interrupt enable** 0_B The active slope STATE interrupt is not enabled 1_B The active slope STATE interrupt is enabled
TASI_IRQ_EN	9	rw	**TRIGGER active slope interrupt enable** 0_B The active slope TRIGGER interrupt is not enabled 1_B The active slope TRIGGER interrupt is enabled
PWI_IRQ_EN	10	rw	**Plausibility window (PVT) violation of TRIGGER interrupt enable** 0_B The plausibility violation interrupt is not enabled 1_B The plausibility violation interrupt is enabled
W2I_IRQ_EN	11	rw	**RAM write access to RAM region 2 interrupt enable** 0_B The RAM write access interrupt is not enabled 1_B The RAM write access interrupt is enabled
W1I_IRQ_EN	12	rw	**Write access to RAM region 1b or 1c interrupt enable** 0_B The RAM write access interrupt is not enabled 1_B The RAM write access interrupt is enabled
GL1I_IRQ_EN	13	rw	**Get of lock interrupt enable, when lock arises** 0_B The get of lock interrupt is not enabled 1_B The get of lock interrupt is enabled
LL1I_IRQ_EN	14	rw	**Loss of lock interrupt enable** 0_B The loss of lock interrupt is not enabled 1_B The loss of lock interrupt is enabled
EI_IRQ_EN	15	rw	**Error interrupt enable (see status register)** 0_B The error interrupt is not enabled 1_B The error interrupt is enabled
GL2I_IRQ_EN	16	rw	**Get of lock interrupt enable for SUB_INC2** 0_B The get of lock interrupt is not enabled 1_B The get of lock interrupt is enabled

Field	Bits	Type	Description
LL2I_IRQ_EN	17	rw	**Loss of lock interrupt enable for SUB_INC2** 0_B The loss of lock interrupt is not enabled 1_B The loss of lock interrupt is enabled
TE0I_IRQ_EN	18	rw	**TRIGGER event interrupt 0 enable** 0_B Interrupt on TRIGGER event 0 is not enabled 1_B Interrupt on TRIGGER event 0 is enabled
TE1I_IRQ_EN	19	rw	**TRIGGER event interrupt 1 enable** 0_B Interrupt on TRIGGER event 1 is not enabled 1_B Interrupt on TRIGGER event 1 is enabled
TE2I_IRQ_EN	20	rw	**TRIGGER event interrupt 2 enable** 0_B Interrupt on TRIGGER event 2 is not enabled 1_B Interrupt on TRIGGER event 2 is enabled
TE3I_IRQ_EN	21	rw	**TRIGGER event interrupt 3 enable** 0_B Interrupt on TRIGGER event 3 is not enabled 1_B Interrupt on TRIGGER event 3 is enabled
TE4I_IRQ_EN	22	rw	**TRIGGER event interrupt 4 enable** 0_B Interrupt on TRIGGER event 4 is not enabled 1_B Interrupt on TRIGGER event 4 is enabled
CDTI_IRQ_EN	23	rw	**Interrupt enable for calculation of TRIGGER duration done** 0_B Interrupt on calculated TRIGGER duration is not enabled 1_B Interrupt on calculated TRIGGER duration is enabled
CDSI_IRQ_EN	24	rw	**Interrupt enable for calculation of STATE duration done** 0_B Interrupt on calculated STATE duration is not enabled 1_B Interrupt on calculated STATE duration is enabled
TORI_IRQ_EN	25	rw	**TRIGGER out of range interrupt enable** 0_B Interrupt on TRIGGER out of range is not enabled 1_B Interrupt on TRIGGER out of range is enabled
SORI_IRQ_EN	26	rw	**STATE out of range interrupt enable** 0_B Interrupt on STATE out of range is not enabled 1_B Interrupt on STATE out of range is enabled
DCGI_IRQ_EN	27	rw	**Direction change interrupt enable** 0_B Interrupt on direction change of TRIGGER is not enabled 1_B Interrupt on direction change of TRIGGER is enabled
0	31:28	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.19Register DPLL_IRQ_FORCINT

DPLL Interrupt Force Register

DPLL_IRQ_FORCINT
DPLL Interrupt Force Register **(028048_H)** **Application Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				**TRG_DCGI**	**TRG_SORI**	**TRG_TORI**	**TRG_CDSI**	**TRG_CDTI**	**TRG_TE4I**	**TRG_TE3I**	**TRG_TE2I**	**TRG_TE1I**	**TRG_TE0I**	**TRG_LL2I**	**TRG_GL2I**
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_EI	**TRG_LL1I**	**TRG_GL1I**	**TRG_W1I**	**TRG_W2I**	**TRG_PWI**	**TRG_TASI**	**TRG_SASI**	**TRG_MTI**	**TRG_MSI**	**TRG_TISI**	**TRG_SISI**	**TRG_TAXI**	**TRG_TINI**	**TRG_PEI**	**TRG_PDI**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TRG_PDI	0	rw	**Force Interrupt PDI** This bit is cleared automatically after write. This bit is write protected by bit RF_PROT of register GTM_CTRL. 0_B The corresponding interrupt is not forced 1_B The corresponding interrupt is forced for one clock
TRG_PEI	1	rw	**Force Interrupt PEI** Coding see bit 0.
TRG_TINI	2	rw	**Force Interrupt TINI** Coding see bit 0.
TRG_TAXI	3	rw	**Force Interrupt TAXI** Coding see bit 0.
TRG_SISI	4	rw	**Force Interrupt SISI** Coding see bit 0.
TRG_TISI	5	rw	**Force Interrupt TISI** Coding see bit 0.
TRG_MSI	6	rw	**Force Interrupt MSI** Coding see bit 0.
TRG_MTI	7	rw	**Force Interrupt MTI** Coding see bit 0.
TRG_SASI	8	rw	**Force Interrupt SASI** Coding see bit 0.
TRG_TASI	9	rw	**Force Interrupt TASI** Coding see bit 0.
TRG_PWI	10	rw	**Force Interrupt PWI** Coding see bit 0.
TRG_W2I	11	rw	**Force Interrupt W2I** Coding see bit 0.

Field	Bits	Type	Description
TRG_W1I	12	rw	**Force Interrupt W1I** Coding see bit 0.
TRG_GL1I	13	rw	**Force Interrupt GL1I** Coding see bit 0.
TRG_LL1I	14	rw	**Force Interrupt LL1I** Coding see bit 0.
TRG_EI	15	rw	**Force Interrupt EI** Coding see bit 0.
TRG_GL2I	16	rw	**Force Interrupt GL2I** Coding see bit 0.
TRG_LL2I	17	rw	**Force Interrupt LL2I** Coding see bit 0.
TRG_TE0I	18	rw	**Force Interrupt TE0I** Coding see bit 0.
TRG_TE1I	19	rw	**Force Interrupt TE1I** Coding see bit 0.
TRG_TE2I	20	rw	**Force Interrupt TE2I** Coding see bit 0.
TRG_TE3I	21	rw	**Force Interrupt TE3I** Coding see bit 0.
TRG_TE4I	22	rw	**Force Interrupt TE4I** Coding see bit 0.
TRG_CDTI	23	rw	**Force Interrupt CDTI** Coding see bit 0.
TRG_CDSI	24	rw	**Force Interrupt CDSI** Coding see bit 0.
TRG_TORI	25	rw	**Force Interrupt TORI** Coding see bit 0.
TRG_SORI	26	rw	**Force Interrupt SORI** Coding see bit 0.
TRG_DCGI	27	rw	**Force interrupt DCGI** Coding see bit 0.
0	31:28	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.20Register DPLL_IRQ_MODE

DPLL Interrupt Mode Register

DPLL_IRQ_MODE
DPLL Interrupt Mode Register **(02804C$_H$)** **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														**IRQ_MODE**	
r														rw	

Field	Bits	Type	Description
IRQ_MODE	1:0	rw	**IRQ mode selection** The interrupt modes are described in **Section 28.4.5**. 00$_B$ Level mode 01$_B$ Pulse mode 10$_B$ Pulse-Notify mode 11$_B$ Single-Pulse mode
0	31:2	r	**Reserved** Read as zero, shall be written as zero

28.20.12.21Register DPLL_EIRQ_EN

DPLL Error Interrupt Enable Register

DPLL_EIRQ_EN
DPLL Error Interrupt Enable Register **(028050$_H$)** **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				**DCGI_EIRQ_EN**	**SORI_EIRQ_EN**	**TORI_EIRQ_EN**	**CDSI_EIRQ_EN**	**CDTI_EIRQ_EN**	**TE4I_EIRQ_EN**	**TE3I_EIRQ_EN**	**TE2I_EIRQ_EN**	**TE1I_EIRQ_EN**	**TE0I_EIRQ_EN**	**LL2I_EIRQ_EN**	**GL2I_EIRQ_EN**
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EI_EIRQ_EN	**LL1I_EIRQ_EN**	**GL1I_EIRQ_EN**	**W1I_EIRQ_EN**	**W2I_EIRQ_EN**	**PWI_EIRQ_EN**	**TASI_EIRQ_EN**	**SASI_EIRQ_EN**	**MTI_EIRQ_EN**	**MSI_EIRQ_EN**	**TISI_EIRQ_EN**	**SISI_EIRQ_EN**	**TAXI_EIRQ_EN**	**TINI_EIRQ_EN**	**PEI_EIRQ_EN**	**PDI_EIRQ_EN**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PDI_EIRQ_EN	0	rw	**DPLL disable interrupt enable, when switch off of the DEN bit** 0_B The DPLL disable interrupt is not enabled 1_B The DPLL disable interrupt is enabled
PEI_EIRQ_EN	1	rw	**DPLL enable interrupt enable, when switch on of the DEN bit** 0_B The DPLL enable interrupt is not enabled 1_B The DPLL enable interrupt is enabled
TINI_EIRQ_EN	2	rw	**TRIGGER minimum hold time violation interrupt enable bit** 0_B Minimum hold time violation of TRIGGER interrupt is not enabled 1_B The minimum hold time violation of TRIGGER interrupt is enabled
TAXI_EIRQ_EN	3	rw	**TRIGGER maximum hold time violation interrupt enable bit** 0_B Maximum hold time violation of TRIGGER interrupt is not enabled 1_B The maximum hold time violation of TRIGGER interrupt is enabled
SISI_EIRQ_EN	4	rw	**STATE inactive slope interrupt enable bit** 0_B The interrupt at the inactive slope of STATE is not enabled 1_B The interrupt at the inactive slope of STATE is enabled
TISI_EIRQ_EN	5	rw	**TRIGGER inactive slope interrupt enable bit** 0_B The interrupt at the inactive slope of TRIGGER is not enabled 1_B The interrupt at the inactive slope of TRIGGER is enabled
MSI_EIRQ_EN	6	rw	**Missing STATE interrupt enable** 0_B The missing STATE interrupt is not enabled 1_B The missing STATE interrupt is enabled
MTI_EIRQ_EN	7	rw	**Missing TRIGGER interrupt enable** 0_B The missing TRIGGER interrupt is not enabled 1_B The missing TRIGGER interrupt is enabled
SASI_EIRQ_EN	8	rw	**STATE active slope interrupt enable** 0_B The active slope STATE interrupt is not enabled. 1_B The active slope STATE interrupt is enabled
TASI_EIRQ_EN	9	rw	**TRIGGER active slope interrupt enable** 0_B The active slope TRIGGER interrupt is not enabled 1_B The active slope TRIGGER interrupt is enabled
PWI_EIRQ_EN	10	rw	**Plausibility window (PVT) violation interrupt of TRIGGER enable** 0_B The plausibility violation interrupt is not enabled 1_B The plausibility violation interrupt is enabled
W2I_EIRQ_EN	11	rw	**RAM write access to RAM region 2 interrupt enable** 0_B The RAM write access interrupt is not enabled 1_B The RAM write access interrupt is enabled
W1I_EIRQ_EN	12	rw	**Write access to RAM region 1b or 1c interrupt** 0_B The RAM write access interrupt is not enabled 1_B The RAM write access interrupt is enabled.
GL1I_EIRQ_EN	13	rw	**Get of lock interrupt enable, when lock arises** 0_B The lock getting interrupt is not enabled 1_B The lock getting interrupt is enabled

Field	Bits	Type	Description
LL1I_EIRQ_EN	14	rw	**Loss of lock interrupt enable** 0_B The lock loss interrupt is not enabled 1_B The lock loss interrupt is enabled
EI_EIRQ_EN	15	rw	**Error interrupt enable (see status register)** 0_B The error interrupt is not enabled 1_B The error interrupt is enabled
GL2I_EIRQ_EN	16	rw	**Get of lock interrupt enable for SUB_INC2** 0_B The lock getting interrupt is not requested 1_B The lock getting interrupt is requested
LL2I_EIRQ_EN	17	rw	**Loss of lock interrupt enable for SUB_INC2** 0_B The lock loss interrupt is not requested 1_B The lock loss interrupt is requested
TE0I_EIRQ_EN	18	rw	**TRIGGER event interrupt 0 enable** 0_B No Interrupt on TRIGGER event 0 enabled 1_B Interrupt on TRIGGER event 0 enabled
TE1I_EIRQ_EN	19	rw	**TRIGGER event interrupt 1 enable** 0_B No Interrupt on TRIGGER event 1 enabled 1_B Interrupt on TRIGGER event 1 enabled
TE2I_EIRQ_EN	20	rw	**TRIGGER event interrupt 2 enable** 0_B No Interrupt on TRIGGER event 2 enabled 1_B Interrupt on TRIGGER event 2 enabled
TE3I_EIRQ_EN	21	rw	**TRIGGER event interrupt 3 enable** 0_B No Interrupt on TRIGGER event 3 enabled 1_B Interrupt on TRIGGER event 3 enabled
TE4I_EIRQ_EN	22	rw	**TRIGGER event interrupt 4 enable** 0_B No Interrupt on TRIGGER event 4 enabled 1_B Interrupt on TRIGGER event 4 enabled
CDTI_EIRQ_EN	23	rw	**Enable interrupt when calculation of TRIGGER duration done** 0_B No Interrupt on calculated TRIGGER duration enabled 1_B Interrupt on calculated TRIGGER duration enabled
CDSI_EIRQ_EN	24	rw	**Enable interrupt when calculation of TRIGGER duration done** 0_B No Interrupt on calculated STATE duration enabled 1_B Interrupt on calculated STATE duration enabled
TORI_EIRQ_EN	25	rw	**TRIGGER out of range interrupt** 0_B No Interrupt when TRIGGER is out of range enabled 1_B Interrupt when TRIGGER is out of range enabled
SORI_EIRQ_EN	26	rw	**STATE out of range** 0_B No Interrupt when STATE is out of range enabled 1_B Interrupt when STATE is out of range enabled
DCGI_EIRQ_EN	27	rw	**Direction change interrupt** 0_B No Interrupt when a direction change of TRIGGER is detected 1_B Interrupt when a direction change of TRIGGER is detected
0	31:28	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.22Register DPLL_INC_CNT1

DPLL Counter for Pulses for TBU_CH1_BASE to be Sent in Automatic End Mode

Counter Value of Sent SUB_INC1 Pulses

DPLL_INC_CNT1
DPLL Counter for Pulses for TBU_CH1_BASE to be Sent in Automatic End Mode($0280B0_H$) Application Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**INC_CNT1**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC_CNT1															
rw															

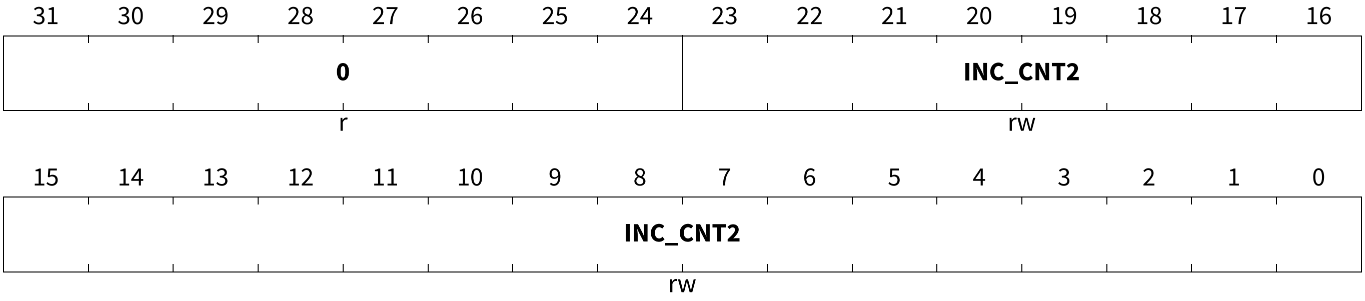
Field	Bits	Type	Description
INC_CNT1	23:0	rw	**Actual number of pulses to be still sent out at the current increment until the next active input signal in automatic end mode** Automatic addition of the number of demanded pulses MLT/MLS1 when getting an active *TRIGGER*/*STATE* input in normal or emergency mode respectively **when SGE1=1**;writeable only for test purposes when DEN=0. In the case of a change of the direction the wrong number of pulses are corrected twice: Add the difference between NMB_T and INC_CNT1 twice to INC_CNT1 before sending out the correction pulses. This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.23Register DPLL_INC_CNT2

DPLL Counter for Pulses for TBU_TS2 to be Sent in Automatic End Mode

Counter Value of sent SUB_INC2 values (for SMC=1 and RMO=1)

DPLL_INC_CNT2
DPLL Counter for Pulses for TBU_TS2 to be Sent in Automatic End Mode(0280B4$_H$) Application Reset Value: 0000 0000$_H$



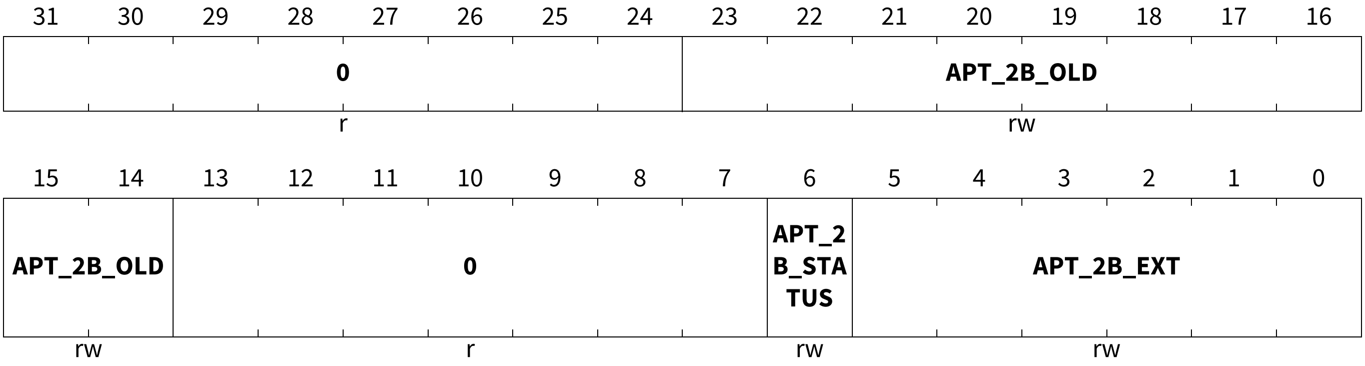
Field	Bits	Type	Description
INC_CNT2	23:0	rw	**Actual number of pulses to be still sent out at the current increment until the next active input signal in automatic end mode** Automatic addition of the number of demanded pulses MLS2 when getting an active *TRIGGER*/*STATE* input in normal or emergency mode respectively **when SGE2=1**; writeable only for test purposes when DEN=0; In the case of a change of the direction the wrong number of pulses are corrected twice: Add the difference between NMB_S and INC_CNT2 twice to INC_CNT2 before sending out the correction pulses. This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.24Register DPLL_APT_SYNC

DPLL Old RAM Pointer and Offset Value for TRIGGER

TRIGGER Time Stamp Field Offset at Synchronization Time

DPLL_APT_SYNC
DPLL Old RAM Pointer and Offset Value for TRIGGER(0280B8$_H$) Application Reset Value: 0000 0000$_H$



Field	Bits	Type	Description
APT_2B_EXT	5:0	rw	**Address pointer 2b extension** This offset value determines, by which value the APT_2B is changed at the synchronization time; set by CPU before the synchronization is performed. This offset value is the number of virtual increments to be inserted in the TSF for an imminent intended synchronization; the CPU sets its value dependent on the gaps until the synchronization time taking into account the considered NUTE value to be set and including the next future increment (when SYN_T_OLD is still 1). When the synchronization takes place, this value is to be added to the APT_2B address pointer (for forward direction, DIR1=0) and the APT_2B_STATUS bit is cleared after it. For backward direction subtract APT_2B_EXT accordingly. This correction is done after updating the RAM TSF with the last TS_T value. When the synchronization is intended and the NUTE value is to be set to FULL_SCALE after it, the APT_2B_EXT value must be set to 2*SYN_NT in order to be able to fill all gaps in the extended TSF_T with the corresponding values by the CPU. When still not all values for FULL_SCALE are available, the APT_2B_EXT value considers only a share according to the corresponding NUTE value to be set after the synchronization.
APT_2B_STATUS	6	rw	**Address pointer 2b status** Set by CPU before the synchronization is performed. The value is cleared when the APT_2B_OLD value is written. 0_B APT_2B_EXT is not to be considered 1_B APT_2B_EXT has to be considered for time stamp field extension
APT_2B_OLD	23:14	rw	**Address pointer TRIGGER for RAM region 2b at synchronization time** This value is set by the current APT_2B value when the synchronization takes place for the first active TRIGGER event after writing APT_2C but before adding the offset value APT_2B_EXT (that means: when APT_2B_STATUS=1). Address pointer APT_2B value at the moment of synchronization, before the offset value is added, that means the pointer with this value points to the last value before the additional inserted gap.
0	13:7, 31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.25Register DPLL_APS_SYNC

DPLL Old RAM Pointer and Offset Value for STATE

Note: STATE Time Stamp Field Offset at Synchronization Time

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is not set. If DPLL_CTRL_11.STATE_EXT is set any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_APS_SYNC

DPLL Old RAM Pointer and Offset Value for STATE(0280BC$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												**APS_1C2_OLD**			
r												rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APS_1C2_OLD		**0**							**APS_1C2_STATUS**	**APS_1C2_EXT**					
rw		r							rw	rw					

Field	Bits	Type	Description
APS_1C2_EXT	5:0	rw	**Address pointer 1c2 extension** This offset value determines, by which value the APS_1C2 is changed at the synchronization time; set by CPU before the synchronization is performed. This offset value is the number of virtual increments to be inserted in the TSF for an imminent intended synchronization; the CPU sets its value dependent on the gaps until the synchronization time taking into account the considered NUSE value to be set and including the next future increment (when SYN_S_OLD is still 1). When the synchronization takes place, this value is to be added to the APS_1C2 address pointer (for forward direction, DIR2=0) and the APT_1c2_status bit is cleared after it. For backward direction subtract APS_1C2_EXT accordingly. When the synchronization is intended and the NUSE value is to be set to FULL_SCALE after it, the APS_1C2_EXT value must be set to SYN_NS (for SYSF=1) or 2*SYN_NS (for SYSF=0) in order to be able to fill all gaps in the extended TSF_S with the corresponding values by the CPU. When still not all values for FULL_SCALE are available, the APS_1C2_EXT value considers only a share according to the NUSE value to be set after the synchronization.
APS_1C2_STATUS	6	rw	**Address pointer 1c2 status** Set by CPU before the synchronization is performed. The value is cleared automatically when the APS_1C2_OLD value is written. 0_B APS_1C2_EXT is not to be considered 1_B APS_1C2_EXT has to be considered for time stamp field extension
APS_1C2_OLD	19:14	rw	**Address pointer STATE for RAM region 1c2 at synchronization time** This value is set by the current APS_1C2 value when the synchronization takes place for the first active STATE event after writing APS_1C3 but before adding the offset value APS_1C2_EXT (that means: when APS_1C2_STATUS=1). Address pointer APS_1C2 value at the moment of synchronization, before the offset value is added, that means the pointer with this value points to the last value before the additional inserted gap.

Field	Bits	Type	Description
0	13:7, 31:20	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.26Register DPLL_TBU_TS0_T

DPLL TBU_TS0 Value at Last TRIGGER Event

Time Stamp Value for the last active TRIGGER

DPLL_TBU_TS0_T

DPLL TBU_TS0 Value at Last TRIGGER Event (0280C0$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TBU_TS0_T**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBU_TS0_T															
rw															

Field	Bits	Type	Description
TBU_TS0_T	23:0	rw	**Value of TBU_TS0 at the last TRIGGER event** For each T_valid the value of TBU_TS0 is stored in this register; the register is writeable only for test purposes when DEN=0. This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.27Register DPLL_TBU_TS0_S

DPLL TBU_TS0 Value at Last STATE Event

Time Stamp Value for the last active STATE

DPLL_TBU_TS0_S

DPLL TBU_TS0 Value at Last STATE Event (0280C4$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TBU_TS0_S**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBU_TS0_S															
rw															

Field	Bits	Type	Description
TBU_TS0_S	23:0	rw	**Value of TBU_TS0 at the last STATE event** For each S_VALID the value of TBU_TS0 is stored in this register;the register is writeable only for test purposes when DEN=0. This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.28Register DPLL_ADD_IN_LD1

DPLL Direct Load Input Value for SUB_INC1

ADD_IN Value in Direct Load Mode for TRIGGER

DPLL_ADD_IN_LD1

DPLL Direct Load Input Value for SUB_INC1 (0280C8$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**ADD_IN_LD1**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_IN_LD1															
rw															

Field	Bits	Type	Description
ADD_IN_LD1	23:0	rw	**Input value for SUB_INC1 generation** Given by CPU. This value can be used in normal und emergency mode (SMC=0) as well as for SMC=1. **For DLM1 = 1:** The value is loaded by the CPU but used by the DPLL only for DLM1=1 (see DPLL_CTRL_1 register). When switching DLM1 to 1, the value in the register is used for the SUB_INC1 generation beginning from the next active *TRIGGER* or *STATE* event respectively independently if new values are written by the CPU or not. When a new value is written the output frequency changes according to the given value beginning immediately from the moment of writing. Do not wait for performing step 10 in the state machine for ADD_IN calculations. If the ADD_IN_LD1 value is zero all pulses are sent with the highest possible frequency. **For DLM1 = 0:** The value loaded by the CPU is stored directly in the internal add_in register which is used to control the sub increment pulse generator directly (see DPLL_CTRL_1 register, DLM1 = 0). When a new ADD_IN_LD1 value is written the output frequency is immediately changed from the moment of writing. The ADD_IN values calculated internally of the DPLL are written to the internal ADD_IN register as well. In the moment when the internal calculation of the ADD_IN values is writing the results into the internal ADD_IN register of the pulse generator the internally calculated ADD_IN values does always have higher priority compared to the values written via the ADD_IN_LD1 register. If the ADD_IN_LD1 value is zero all pulses are sent with the highest possible frequency.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.29Register DPLL_ADD_IN_LD2

DPLL Direct Load Input Value for SUB_INC2

DPLL_ADD_IN_LD2
DPLL Direct Load Input Value for SUB_INC2 (0280CC$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**ADD_IN_LD2**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_IN_LD2															
rw															

Field	Bits	Type	Description
ADD_IN_LD2	23:0	rw	**Input value for SUB_INC2 generation** Given by CPU. This value can be used for SMC=1 while RMO=1. **For DLM2 = 1:** The value is loaded by the CPU but used by the DPLL only for DLM2=1 (see DPLL_CTRL_1 register). When switching DLM2 to 1, the value in the register is used for the SUB_INC2 generation beginning from the next *STATE* event respectively independently if new values are written by the CPU or not. When a new value is written the output frequency changes according to the given value beginning immediately from the moment of writing. Do not wait for performing step 30 in the state machine for ADD_IN calculations. If the ADD_IN_LD2 value is zero all pulses are sent with the highest possible frequency. **For DLM2 = 0:** The value loaded by the CPU is stored directly in the internal add_in register which is used to control the sub increment pulse generator directly (see DPLL_CTRL_1 register, DLM2 = 0). When a new ADD_IN_LD2 value is written the output frequency is immediately changed from the moment of writing. The ADD_IN values calculated internally of the DPLL are written to the internal ADD_IN register as well. In the moment when the internal calculation of the ADD_IN values is writing the results into the internal ADD_IN register of the pulse generator the internally calculated ADD_IN values does always have higher priority compared to the values written via the ADD_IN_LD2 register. If the ADD_IN_LD2 value is zero all pulses are sent with the highest possible frequency.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.30Register DPLL_STATUS

DPLL Status Register

Note: The DPLL_STATUS register is reset, when the DPLL is disabled (switching DEN from 1 to 0).

DPLL_STATUS

DPLL Status Register **($0280FC_H$)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERR	LOCK1	FTD	FSD	SYT	SYS	LOCK2	0	BWD1	BWD2	ITN	ISN	CAIP1	CAIP2	CSVT	CSVS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOW_RES	0		RAM2_ERR	MT	TOR	MS	SOR	PSE	RCT	RCS	CRO	CTO	0	CSO	FPCE
r	r		rw	rw	rw	rw	rw	r	r	r	rw	rw	r	rw	rw

Field	Bits	Type	Description
FPCE	0	rw	**Fast pulse correction error** 0_B No error at fast pulse correction detected 1_B Negative value of MPVAL1/2 used for fast pulse correction mode
CSO	1	rw	**Calculated STATE duration overflow** Bit is set when equations DPLL-10a or DPLL-10b lead to an overflow. 0_B No overflow at equation DPLL-10a or b 1_B Overflow at equation DPLL-10a or b
CTO	3	rw	**Calculated TRIGGER duration overflow** Bit is set when equations DPLL-5a or DPLL-5b lead to an overflow. When one of the above bits is set the corresponding register contains the maximum value 0xFFFFFF. 0_B No overflow at equation DPLL-5a or b 1_B Overflow at equation DPLL-5a or b
CRO	4	rw	**Calculated Reciprocal value overflow** Bit is set when the calculation of RDT_T_ACT or RDT_S_ACT leads to an overflow. An overflow in calculation of reciprocal values can occur, when the condition of Note [3] to the DPLL_CTRL_0 register is violated (see **Section 28.20.12.1**). Such an overflow can occur according to the calculations in equations (DPLL-1c) or (DPLL-6c). The overflow is detected when after the calculation and shifting left 32 bits at least one of the bits 31 to 24 is not zero. In that case the corresponding register is set to 0xFFFFFF. 0_B No overflow at any reciprocal calculation 1_B Overflow for at least one reciprocal calculation
RCS	5	r	**Resolution conflict STATE** 0_B No resolution conflict detected 1_B The TS0_HRS value is set to 1 while LOW_RES=0
RCT	6	r	**Resolution conflict TRIGGER** 0_B No resolution conflict detected 1_B The TS0_HRT value is set to 1 while LOW_RES=0
PSE	7	r	**Prediction space configuration error** 0_B No prediction space error detected 1_B Configured offset value of RAM2 is too small in order to store all TNU+1 values twice in FULL_SCALE

Field	Bits	Type	Description
SOR	8	rw	**STATE out of range** The SOR bit is set, when the time to the next active *STATE* slope exceeds the value of the last nominal *STATE* duration multiplied with the value of the SLR register (see **Section 28.20.12.73**) and is reset, when at the current or last active input event a direction change was detected. The SYS bit is not influenced by setting the SOR bit. 0_B All STATE signal events appear within SLR interval or a direction change was detected 1_B At least one STATE signal event is out of SLR; address pointers APS, APS_1C2 and APS_1C3 are frozen, and the generation of pulses SUB_INC1, 2, respectively, is stopped
MS	9	rw	**Missing STATE detected according to SOV** 0_B No missing STATE detected or a new valid STATE slope occurred 1_B At least one missing STATE detected after the last valid slope
TOR	10	rw	**TRIGGER out of range** The TOR bit is set, when the time to the next active *TRIGGER* slope exceeds the value of the last nominal *TRIGGER* duration multiplied with the value of the TLR register (see **Section 28.20.12.72**) and is reset, when at the current or last active input event a direction change was detected. The SYT bit is not influenced by setting the TOR bit. 0_B All TRIGGER signal events appear within TLR interval or a direction change was detected 1_B At least one TRIGGER signal event is out of TLR; address pointers APT, APT_2B and APT_2C are frozen, and the generation of pulses SUB_INC1 is stopped
MT	11	rw	**Missing TRIGGER detected according to TOV** 0_B No missing TRIGGER detected or a new valid TRIGGER slope occurred 1_B At least one missing TRIGGER detected after the last valid slope
RAM2_ERR	12	rw	**DPLL internal access to not configured RAM2 memory space** 0_B No access to not configured RAM2 memory space 1_B Access to not configured RAM2 memory space
LOW_RES	15	r	**Low resolution of TBU_TS0 is used for DPLL input** This value reflects the input signal LOW_RES. 0_B The lower 24 bits of TBU_TS0 are used as input for the DPLL 1_B The higher 24 bits of TBU_TS0 are used as input for the DPLL
CSVS	16	r	**Current signal value STATE** 0_B The last STATE_S value was 0 1_B The last STATE_S value was 1
CSVT	17	r	**Current signal value TRIGGER** 0_B The last TRIGGER_S value was 0 1_B The last TRIGGER_S value was 1
CAIP2	18	r	**Calculation of upper half actions in progress** 0_B Currently no action calculation, new data requests possible 1_B Action calculation in progress, no new data requests possible

Field	Bits	Type	Description
CAIP1	19	r	**Calculation of lower half actionsin progress** 0_B Currently no action calculation, new data requests possible 1_B Action calculation in progress, no new data requests possible
ISN	20	r	**Increment number of STATE is not plausible** 0_B The number of STATE events between synchronization gaps is plausible, a direction change is detected, or the APS_1C3 pointer is written 1_B After setting LOCK1 in emergency mode (SMC=0 and RMO=1) or LOCK2 for SMC=RMO=1, missing or additional STATE signals detected; bit is cleared when a direction change is detected or the APS_1C3 is written
ITN	21	r	**Increment number of TRIGGER is not plausible** Bit is set when the number of TRIGGERS is different to profile. 0_B The number of TRIGGER events between synchronization gaps is plausible, a direction change is detected, or the address pointer APT_2C is written 1_B After setting LOCK1 in normal mode (for SMC=0 or SMC=1) or in emergency mode (only for SMC=0), missing or additional TRIGGER signals detected; bit is cleared when a direction change is detected or the APT_2C is written
BWD2	22	r	**Backwards drive of SUB_INC2** 0_B Forward direction 1_B Backward direction
BWD1	23	r	**Backwards drive of SUB_INC1** 0_B Forward direction 1_B Backward direction
LOCK2	25	r	**DPLL Lock status concerning SUB_INC2** Locking of SUB_INC2 appears **for RMO=SMC=1:** Bit is set, when SYS is set and the number of events between two missing *STATE*s is as expected by the SYN_S values. LOCK2 is set for SMC=RMO=1: for an active STATE event when SYS is set and SYN_NS=0or when SYS is set and the profile stored in the ADT_Si field matches once between two gaps. LOCK2 is reset: for SMC=RMO=1 when a missing STATE event occurs while SYN_S=1. This does mean an unexpected missing STATE. when the corresponding input signal STATE is out of locking range SLR 0_B The DPLL is not locked concerning STATE for SMC=1 1_B The DPLL is locked concerning STATE for SMC=1
SYS	26	r	**Synchronization condition of STATE fixed** This bit is set when the CPU writes to the APS_1C3 address pointer.
SYT	27	r	**Synchronization condition of TRIGGER fixed** This bit is set when the CPU writes to the APT_2C address pointer.

Field	Bits	Type	Description
FSD	28	r	**First STATE detected** No change of FSD for switching from normal to emergency mode or vice versa. 0_B Still no active STATE event was detected after enabling DPLL 1_B At least one active STATE event was detected after enabling DPLL
FTD	29	r	**First TRIGGER detected** No change of FTD for switching from normal to emergency mode or vice versa. 0_B No active TRIGGER event was detected after enabling DPLL 1_B At least one active TRIGGER event was detected after enabling DPLL

Field	Bits	Type	Description
LOCK1	30	r	**DPLL Lock status concerning SUB_INC1** LOCK1 is set: • in normal mode (for RMO=SMC=0, LCD=0): Bit is set for an active TRIGGER event when SYT is set and the number of events between two gaps is as expected by the profile (NT values in the ADT_T[i] field) or when SYN_NT=0 and SYT=1. • in normal mode (for RMO=SMC=0, LCD=1): Bit is set for an active TRIGGER event when SYT is set and the number of events between two increments without missing TRIGGER (no gap) is as expected by the profile (NT values in the ADT_T[i] field) • in emergency mode (for RMO=1 and SMC=0): Bit is set for an active STATE event, when SYS is set and the received events are in correspondence to the profile (NS values in the ADT_S[i] field) for at least two (four in case of direction change) expected missing STATE events or when SYN_NS=0. • for SMC=1: Bit is set for an active TRIGGER even when SYT is set and SYN_NT=0 or when SYT is set and the profile stored in the ADT_T[i] field matches once between two gaps. LOCK1 is reset for RMO=SMC=0: • when a corresponding missing TRIGGER event occurs while SYN_T=1. This does mean an unexpected missing TRIGGER. • when the corresponding input signal TRIGGER is out of locking range TLR, • when a corresponding direction change is detected: for RMO=1 and SMC=0: • when a corresponding missing STATE event occurs while SYN_S=1. This does mean an unexpected missing STATE. • when the corresponding input signal STATE is out of locking range TLR for SMC=1: • when a corresponding missing TRIGGER event occurs while SYN_T=1. This does mean an unexpected missing TRIGGER. • when the corresponding input signal TRIGGER is out of locking range TLR, • when a corresponding direction change is detected 0_B The DPLL is not locked for TRIGGER (while SMC=RMO=0 or SMC=1) or for STATE (while SMC=0 and RMO=1) 1_B The DPLL is locked for TRIGGER (while SMC=RMO=0 or SMC=1) or for STATE (while SMC=0 and RMO=1)
ERR	31	r	**Error during configuration or operation resulting in unexpected values** 0_B When all bits in position 8 to 0 and 10 and 12 are zero 1_B When at least one bit in position 8 to 0 or 10 or 12 is one

Field	Bits	Type	Description
0	2, 14:13, 24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.31Register DPLL_ID_PMTR_[z]

DPLL ID Information for Input Signal PMT z Register

DPLL_ID_PMTR_z (z=0-31)
DPLL ID Information for Input Signal PMT z Register(028100$_H$+z*4) Application Reset Value: 0000 01FE$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							**ID_PMTR_X**								
r							rw								

Field	Bits	Type	Description
ID_PMTR_X	8:0	rw	**ID information to the input signal PMTR[z] from the ARU** This value can only be written when the action [z] is disabled by the correspondent bit AENz=0 of the registers DPLL_CTRL_2, ...5, respectively, or when the DPLL is disabled (DEN=0).
0	31:9	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.32Register DPLL_CTRL_0_SHADOW_TRIGGER

DPLL Control 0 Shadow Trigger Register

Shadow Register of DPLL_CTRL_0 controlled by an active TRIGGER Slope

DPLL_CTRL_0_SHADOW_TRIGGER
DPLL Control 0 Shadow Trigger Register (0281E0$_H$) Application Reset Value: 0000 0257$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMO	**0**		**IDT**	**0**	**AMT**	**0**									
r	r		r	r	r	r									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					**IFP**	**MLT**									
r					r	r									

Field	Bits	Type	Description
MLT	9:0	r	**Multiplier for TRIGGER** 1) MLT+1 is number of SUB_INC1 pulses between two TRIGGER events in normal mode (1…1024).
IFP	10	r	**Input filter position** 1) Value contains position or time related information.
AMT	26	r	**Adapt mode TRIGGER** 1) Use of adaptation information of TRIGGER.
IDT	28	r	**Input delay TRIGGER** 1) Use of input delay information transmitted in FT part of the TRIGGER signal.
RMO	31	r	**Reference mode** 1) Selection of the relevant the input signal for generation of SUB_INC1.
0	25:11, 27, 30:29	r	**Reserved** Read as zero, shall be written as zero.

1) Only the values characterized by 1) are stored for an active TRIGGER slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_0 are transferred without any input event at the next system clock. This results in the above reset value.

28.20.12.33Register DPLL_CTRL_0_SHADOW_STATE

DPLL Control 0 Shadow STATE Register

Shadow Register of DPLL_CTRL_0 controlled by an active STATE Slope

DPLL_CTRL_0_SHADOW_STATE
DPLL Control 0 Shadow STATE Register (0281E4$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMO		**0**		**IDS**	**0**	**AMS**					**0**				
r		r		r	r	r					r				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		0			**IFP**					**0**					
		r			r					r					

Field	Bits	Type	Description
IFP	10	r	**Input filter position** 1) **Value contains position or time related information.**

Field	Bits	Type	Description
AMS	25	r	**Adapt mode STATE** 1) Use of adaptation information of STATE.
IDS	27	r	**Input delay STATE** 1) Use of input delay information transmitted in FT part of the STATE signal.
RMO	31	r	**Reference mode** 1) Selection of the relevant the input signal for generation of SUB_INC1.
0	9:0, 24:11, 26, 30:28	r	**Reserved** Read as zero, shall be written as zero.

1) Only the values characterized by 1) are stored for an active STATE slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_0 are transferred without any input event at the next system clock.

28.20.12.34Register DPLL_CTRL_1_SHADOW_TRIGGER

DPLL Control 1 Shadow TRIGGER Register

Shadow Register of DPLL_CTRL_1 controlled by an active TRIGGER Slope

DPLL_CTRL_1_SHADOW_TRIGGER
DPLL Control 1 Shadow TRIGGER Register (0281E8$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**PCM1**	**DLM1**	**SGE1**	**PIT**	**COA**	**0**		**DMO**
r								r	r	r	r	r	r		r

Field	Bits	Type	Description
DMO	0	r	**DPLL mode select** 1)
COA	3	r	**Correction strategy in automatic end mode (DMO=0)** 1)
PIT	4	r	**Plausibility value PVT to next valid TRIGGER is time related** 1)
SGE1	5	r	**SUB_INC1 generator enable** 1)
DLM1	6	r	**Direct Load Mode for SUB_INC1 generation** 1)

Field	Bits	Type	Description
PCM1	7	r	**Pulse Correction Mode for SUB_INC1 generation** 1)
0	2:1, 31:8	r	**Reserved** Read as zero, shall be written as zero.

1) Only the values characterized by 1) are stored for an active TRIGGER slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_1 are transferred without any input event at the next system clock.

28.20.12.35Register DPLL_CTRL_1_SHADOW_STATE

DPLL Control 1 Shadow STATE Register

DPLL_CTRL_1_SHADOW_STATE
DPLL Control 1 Shadow STATE Register **(0281EC$_H$)** **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					**PCM2**	**DLM2**	**SGE2**	**PCM1**	**DLM1**	**SGE1**	**0**	**COA**	**0**		**DMO**
r					r	r	r	r	r	r	r	r	r		r

Field	Bits	Type	Description
DMO	0	r	**DPLL mode select** 1)
COA	3	r	**Correction strategy in automatic end mode (DMO=0)** 1)
SGE1	5	r	**SUB_INC1 generator enable** 1)
DLM1	6	r	**Direct Load Mode for SUB_INC1 generation** 1)
PCM1	7	r	**Pulse Correction Mode for SUB_INC1 generation** 1)
SGE2	8	r	**SUB_INC2 generator enable** 1)
DLM2	9	r	**Direct Load Mode for SUB_INC2 generation** 1)
PCM2	10	r	**Pulse Correction Mode for SUB_INC2 generation** 1)
0	2:1, 4, 31:11	r	**Reserved** Read as zero, shall be written as zero.

1) Only the values characterized by 1) are stored for an active STATE slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_1 are transferred without any input event at the next system clock.

28.20.12.36Register DPLL_RAM_INI

DPLL RAM Initialization Register

DPLL_RAM_INI
DPLL RAM Initialization Register **($0281FC_H$)** **Application Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											**INIT_RAM**	**0**	**INIT_2**	**INIT_1BC**	**INIT_1A**
r											rw	r	r	r	r

Field	Bits	Type	Description
INIT_1A	0	r	**RAM region 1a initialization in progress** 0_B No initialization of considered RAM region in progress 1_B Initialization of considered RAM region in progress
INIT_1BC	1	r	**RAM region 1b and 1c initialization in progress** Coding see bit 0.
INIT_2	2	r	**RAM region 2 initialization in progress** Coding see bit 0.
INIT_RAM	4	rw	**RAM regions 1a, 1b and 2 are to be initialized** Setting the INIT_RAM bit results only in a RAM reset when the DPLL is not enabled (DEN=0). Depending on the vendor configuration, the connected RAM regions are initialized to zero in the case of a module HW reset or for setting the RST bit in the GTM_RST register. In the case of no RAM initialization, it must be ensured that all relevant parameters are configured correctly. Otherwise, there is no guarantee to get a predictable behavior. 0_B Do not start initialization of all RAM regions 1_B Start initialization of all RAM regions
0	3, 31:5	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.37Memory DPLL_TS_T

DPLL Actual TRIGGER Time Stamp Value

DPLL_TS_T
DPLL Actual TRIGGER Time Stamp Value (028400$_H$) Reset Value: Table 82

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TRIGGER_TS**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGGER_TS															
rw															

Field	Bits	Type	Description
TRIGGER_TS	23:0	rw	**Time stamp value of the last active TRIGGER input** Measured TRIGGER time stamp. The LSB address is determined using the SWON_T value in the OSW register (see **Section 28.20.12.8**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 82 Reset Values of DPLL_TS_T

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.38Memory DPLL_TS_T_OLD

DPLL Previous TRIGGER Time Stamp Value

DPLL_TS_T_OLD
DPLL Previous TRIGGER Time Stamp Value (028404$_H$) Reset Value: Table 83

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TRIGGER_TS_OLD**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGGER_TS_OLD															
rw															

Field	Bits	Type	Description
TRIGGER_TS_OLD	23:0	rw	**Time stamp value of the last but one active TRIGGER input** Previous measured TRIGGER time stamp. The LSB address is determined using the SWON_T value in the OSW register (see **Section 28.20.12.8**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 83 Reset Values of DPLL_TS_T_OLD

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.39Memory DPLL_FTV_T

DPLL Actual TRIGGER Filter Value

DPLL_FTV_T

DPLL Actual TRIGGER Filter Value (028408$_H$) Reset Value: Table 84

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TRIGGER_FT**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGGER_FT															
rw															

Field	Bits	Type	Description
TRIGGER_FT	23:0	rw	**Filter value of the last active TRIGGER input** Transmitted filter value. The LSB address is determined using the SWON_T value in the OSW register (see **Section 28.20.12.8**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 84 Reset Values of DPLL_FTV_T

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.40Memory DPLL_TS_S

DPLL Actual STATE Time Stamp

DPLL_TS_S
DPLL Actual STATE Time Stamp **(028410_H)** **Reset Value: Table 85**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**STATE_TS**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATE_TS															
rw															

Field	Bits	Type	Description
STATE_TS	23:0	rw	**Time stamp value of the last active STATE input** The LSB address is determined using the SWON_S value in the OSW register (see **Section 28.20.12.8**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 85 Reset Values of DPLL_TS_S

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000_H	Cleared by state machine

28.20.12.41Memory DPLL_TS_S_OLD

DPLL Previous STATE Time Stamp

DPLL_TS_S_OLD
DPLL Previous STATE Time Stamp **(028414_H)** **Reset Value: Table 86**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**STATE_TS_OLD**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATE_TS_OLD															
rw															

Field	Bits	Type	Description
STATE_TS_OLD	23:0	rw	**Time stamp value of the last active STATE input** The LSB address is determined using the SWON_S value in the OSW register (see **Section 28.20.12.8**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 86 Reset Values of DPLL_TS_S_OLD

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.42Memory DPLL_FTV_S

DPLL Actual STATE Filter Value

DPLL_FTV_S

DPLL Actual STATE Filter Value (028418$_H$) Reset Value: Table 87

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**STATE_FT**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATE_FT															
rw															

Field	Bits	Type	Description
STATE_FT	23:0	rw	**Filter value of the last active STATE input** transmitted filter value The LSB address is determined using the SWON_S value in the OSW register (see **Section 28.20.12.8**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 87 Reset Values of DPLL_FTV_S

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.43Memory DPLL_THMI

DPLL TRIGGER Hold Time Minimum Value

DPLL_THMI
DPLL TRIGGER Hold Time Minimum Value (028420$_H$) Reset Value: Table 88

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THMI															
rw															

Field	Bits	Type	Description
THMI	15:0	rw	**Minimal time between active and inactive TRIGGER slope (uint16)** The time value corresponds to the time stamp clock counts: this does mean the clock selected for the TBU_CH0_BASE (see TBU_CH0_CTRL register) set min. value; generate the TINI interrupt in the case of a violation for THMI>0. Typical retention time values after an active slope can be e.g. between 45 µs (forwards) and 90 µs (backwards). When THMI is zero, consider always a THMI violation (forwards).
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 88 Reset Values of DPLL_THMI

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.44Memory DPLL_THMA

DPLL TRIGGER Hold Time Maximum Value

DPLL_THMA
DPLL TRIGGER Hold Time Maximum Value (028424$_H$) Reset Value: Table 89

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THMA															
rw															

Field	Bits	Type	Description
THMA	15:0	rw	**Maximal time between active and inactive TRIGGER slope (uint16)** The time value corresponds to the time stamp clock counts: This does mean the clock selected for the TBU_CH0_BASE (see TBU_CH0_CTRL register) Max. value to be set; generate the TAX interrupt in the case of a violation for THMA>0.
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 89 Reset Values of DPLL_THMA

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.45Memory DPLL_THVAL

DPLL Measured TRIGGER Hold Time Value

Note: In the case of LOW_RES=1 and TBU_HRT=0 the difference between the time stamps of active and inactive slope is multiplied by 8. The register contains this value.

DPLL_THVAL

DPLL Measured TRIGGER Hold Time Value (028428$_H$) Reset Value: Table 90

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**THVAL**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THVAL															
rw															

Field	Bits	Type	Description
THVAL	23:0	rw	**Measured time from the last active slope to the next inactive TRIGGER slope in time stamp clock counts: this does mean the clock selected for the TBU_CH0_BASE (uint16)** The measured value considers all input slope filter delays. From the received input the corresponding filter delays are subtracted before the time stamp difference of active and inactive slope is calculated.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 90 Reset Values of DPLL_THVAL

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.46Memory DPLL_TOV

DPLL Time Out Value of Active TRIGGER Slope

DPLL_TOV

DPLL Time Out Value of Active TRIGGER Slope (028430$_H$) Reset Value: Table 91

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOV_DW						**TOV_DB**									
rw						rw									

Field	Bits	Type	Description
TOV_DB	9:0	rw	**Decision value (fractional part) for missing TRIGGER interrupt**

Field	Bits	Type	Description
TOV_DW	15:10	rw	**Decision value (integer part) for missing TRIGGER interrupt** TOV(15:0) is to be multiplied with the duration of the last increment and divided by 1024 in order to get the time-out time value for a missing TRIGGER event. For the case of LOW_RES=1 (see DPLL_STATUS register) consider for the calculation of the time out value the following cases: LOW_RES=1 and DPLL_CTRL_1/TS0_HRT=1: multiply the TBU_TS0 value by 8 LOW_RES=1 and DPLL_CTRL_1/TS0_HRT=0: multiply the TBU_TS0 value by 8 multiply the estimated time point value (using TS_T, dt_t_ACT and TOV) by 8 LOW_RES=0 and DPLL_CTRL_1/TS0_HRT=0: use TBU_TS0 and the estimated time point value unchanged.
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 91 Reset Values of DPLL_TOV

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	$--00\ 0000_H$	Cleared by state machine

28.20.12.47Memory DPLL_TOV_S

DPLL Time Out Value of Active STATE Slope

DPLL_TOV_S

DPLL Time Out Value of Active STATE Slope (028434$_H$) Reset Value: Table 92

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DW						**DB**									
rw						rw									

Field	Bits	Type	Description
DB	9:0	rw	**Decision value (fractional part) for missing STATE interrupt**

Field	Bits	Type	Description
DW	15:10	rw	**Decision value (integer part) for missing STATE interrupt** TOV_S (15:0) is to be multiplied with the duration of the last increment and divided by 1024 in order to get the time-out time value for a missing STATE event. For the case of LOW_RES=1 (see DPLL_STATUS register) consider for the calculation of the time out value the following cases: LOW_RES=1 and DPLL_CTRL_1/TS0_HRS=1: multiply the TBU_TS0 value by 8 LOW_RES=1 and DPLL_CTRL_1/TS0_HRS=0: multiply the TBU_TS0 value by 8 multiply the estimated time point value (using TS_T, dt_s_ACT and SOV) by 8 LOW_RES=0 and DPLL_CTRL_1/TS0_HRS=0: use TBU_TS0 and the estimated time point value unchanged.
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 92 Reset Values of DPLL_TOV_S

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX_H	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.48Memory DPLL_ADD_IN_CAL1

DPLL Calculated ADD_IN Value for SUB_INC1 Generation

DPLL_ADD_IN_CAL1
DPLL Calculated ADD_IN Value for SUB_INC1 Generation(028438_H) **Reset Value: Table 93**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**ADD_IN_CAL1**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_IN_CAL1															
rw															

Field	Bits	Type	Description
ADD_IN_CAL1	23:0	rw	**Calculated input value for SUB_INC1 generation, calculated by the DPLL** Calculated value. The update of the ADD_IN value by the new calculated value ADD_IN_CAL1 is suppressed for one increment when an unexpected missing TRIGGER (SMC=1 or RMO=0) or an unexpected STATE (RMO=1 and SMC=0) is detected.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 93 Reset Values of DPLL_ADD_IN_CAL1

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.49Memory DPLL_ADD_IN_CAL2

DPLL Calculated ADD_IN Value for SUB_INC2 Generation

DPLL_ADD_IN_CAL2
DPLL Calculated ADD_IN Value for SUB_INC2 Generation($02843C_H$) **Reset Value: Table 94**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**ADD_IN_CAL2**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_IN_CAL2															
rw															

Field	Bits	Type	Description
ADD_IN_CAL2	23:0	rw	**Input value for SUB_INC2 generation, calculated by the DPLL for SMC=RMO=1** Calculated value. The update of the ADD_IN value by the calculated value ADD_IN_CAL2 is suppressed for one increment when an unexpected missing STATE (RMO=SMC=1) is detected.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 94 Reset Values of DPLL_ADD_IN_CAL2

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.50Memory DPLL_MPVAL1

DPLL Missing Pulses to be Added or Subtracted Directly 1

Note: Do not provide negative values which exceed the amount of NT*(MLT+1) or MLS1 respectively; when considered negative PD values the sum of both (MPVAL1 + NT*PD) should not exceed the amount of NT*(MLT+1) or MLS1 respectively.

DPLL_MPVAL1
DPLL Missing Pulses to be Added or Subtracted Directly 1(028440$_H$) **Reset Value: Table 95**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**SIX1**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPVAL1															
rw															

Field	Bits	Type	Description
MPVAL1	15:0	rw	**Missing pulses for direct correction of SUB_INC1 pulses by the CPU (sint16)** Used only for RMO=0 or SMC=1 for the case PCM1=1. Add MPVAL1 once to INC_CNT1 and reset PCM1 after applying once.
SIX1	23:16	rw	**Sign extension for MPVAL1** All bits must be written to either all zeros or all ones. 00$_H$ MPVAL is a positive number 00$_H$ MPVAL1 is a negative number
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 95 Reset Values of DPLL_MPVAL1

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.51Memory DPLL_MPVAL2

DPLL Missing Pulses to be Added or Subtracted Directly 2

DPLL_MPVAL2
DPLL Missing Pulses to be Added or Subtracted Directly 2(028444$_H$) **Reset Value: Table 96**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**SIX2**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPVAL2															
rw															

Field	Bits	Type	Description
MPVAL2	15:0	rw	**Missing pulses for direct correction of SUB_INC2 pulses by the CPU (sint16)** Used only for SMC=RMO=1 for the case PCM2=1. Add MPVAL2 once to INC_CNT2, and reset PCM2 after applying once. Do not provide negative values which exceed the amount of MLS2; when considered negative PD_S values, the sum of both should not exceed the amount of MLS2.
SIX2	23:16	rw	**Sign extension for MPVAL2** All bits must be written to either all zeros or all ones. 00$_H$ MPVAL2 is a positive number FF$_H$ MPVAL2 is a negative number
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 96 Reset Values of DPLL_MPVAL2

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	––00 0000$_H$	Cleared by state machine

28.20.12.52Memory DPLL_NMB_T_TAR

DPLL Target Number of Pulses to be Sent in Normal Mode

DPLL_NMB_T_TAR
DPLL Target Number of Pulses to be Sent in Normal Mode(028448$_H$) **Reset Value: Table 97**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMB_T_TAR															
rw															

Field	Bits	Type	Description
NMB_T_TAR	15:0	rw	**Target Number of pulses for TRIGGER** Calculated number of pulses in normal mode for the current *TRIGGER* increment without missing pulses. Calculated target pulse number. The LSB address is determined using the SWON_T value in the OSW register (see **Section 28.20.12.8**).
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 97 Reset Values of DPLL_NMB_T_TAR

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.53Memory DPLL_NMB_T_TAR_OLD

DPLL Last but One Target Number of Pulses to be Sent in Normal Mode

DPLL_NMB_T_TAR_OLD
DPLL Last but One Target Number of Pulses to be Sent in Normal Mode(02844C$_H$) **Reset Value: Table 98**

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8"><b>0</b></td><td colspan="8"><b>0</b></td></tr>
<tr><td colspan="8">r</td><td colspan="8">rw</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16"><b>NMB_T_TAR_OLD</b></td></tr>
<tr><td colspan="16">rw</td></tr>
</table>

Field	Bits	Type	Description
NMB_T_TAR_OLD	15:0	rw	**Target Number of pulses for TRIGGER** Calculated number of pulses in normal mode for the current *TRIGGER* increment without missing pulses. Calculated target pulse number. The LSB address is determined using the SWON_T value in the OSW register (see **Section 28.20.12.8**).
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 98 Reset Values of DPLL_NMB_T_TAR_OLD

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	––00 0000$_H$	Cleared by state machine

28.20.12.54Memory DPLL_NMB_S_TAR

DPLL Target Number of Pulses to be Sent in Emergency Mode

DPLL_NMB_S_TAR
DPLL Target Number of Pulses to be Sent in Emergency Mode(028450$_H$) **Reset Value: Table 99**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**				**NMB_S_TAR**			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMB_S_TAR															
rw															

Field	Bits	Type	Description
NMB_S_TAR	19:0	rw	**Target Number of pulses for STATE** Calculated number of pulses in emergency mode for the current *STATE* increment without missing pulses. Calculated target pulse number. The LSB address is determined using the SWON_S value in the OSW register (see **Section 28.20.12.8**).
0	23:20	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 99 Reset Values of DPLL_NMB_S_TAR

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	––00 0000$_H$	Cleared by state machine

28.20.12.55Memory DPLL_NMB_S_TAR_OLD

DPLL Last but One Target Number of Pulses to be Sent in Emergency Mode

DPLL_NMB_S_TAR_OLD
DPLL Last but One Target Number of Pulses to be Sent in Emergency Mode(028454$_H$) **Reset Value: Table 100**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**				**NMB_S_TAR_OLD**			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMB_S_TAR_OLD															
rw															

Field	Bits	Type	Description
NMB_S_TAR_OLD	19:0	rw	**Target Number of pulses for STATE** Calculated number of pulses in emergency mode for the current *STATE* increment without missing pulses. Calculated target pulse number. The LSB address is determined using the SWON_S value in the OSW register (see **Section 28.20.12.8**).
0	23:20	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 100 Reset Values of DPLL_NMB_S_TAR_OLD

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.56Memory DPLL_RCDT_TX

DPLL Reciprocal Value of the Expected Increment Duration of TRIGGER

DPLL_RCDT_TX
DPLL Reciprocal Value of the Expected Increment Duration of TRIGGER(028460$_H$) **Reset Value: Table 101**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**RCDT_TX**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCDT_TX															
rw															

Field	Bits	Type	Description
RCDT_TX	23:0	rw	**Reciprocal value of expected increment duration *2^32 while only the lower 24 bits are used** Calculated value; when an overflow occurs in the calculation, the value is set to 0xFFFFFF.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 101 Reset Values of DPLL_RCDT_TX

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.57Memory DPLL_RCDT_SX

DPLL Reciprocal Value of the Expected Increment Duration of STATE

DPLL_RCDT_SX
DPLL Reciprocal Value of the Expected Increment Duration of STATE(028464$_H$) **Reset Value: Table 102**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**RCDT_SX**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCDT_SX															
rw															

Field	Bits	Type	Description
RCDT_SX	23:0	rw	**Reciprocal value of expected increment duration *2^32 while only the lower 24 bits are used** Calculated value; when an overflow occurs in the calculation, the value is set to 0xFFFFFF.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 102 Reset Values of DPLL_RCDT_SX

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.58Memory DPLL_RCDT_TX_NOM

DPLL Reciprocal Value of the Expected Nominal Increment Duration of TRIGGER

DPLL_RCDT_TX_NOM
DPLL Reciprocal Value of the Expected Nominal Increment Duration of TRIGGER(028468_H) Reset Value: Table 103

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**RCDT_TX_NOM**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCDT_TX_NOM															
rw															

Field	Bits	Type	Description
RCDT_TX_NOM	23:0	rw	**Reciprocal value of nominal increment duration *2^32 while only the lower 24 bits are used** Calculated value; when an overflow occurs in the calculation, the value is set to 0xFFFFFF.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 103 Reset Values of DPLL_RCDT_TX_NOM

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.59Memory DPLL_RCDT_SX_NOM

DPLL Reciprocal Value of the Expected Nominal Increment Duration of STATE

Note: RCDT_TX_NOM and RCDT_SX_NOM are calculated by the values RCDT_TX and RCDT_SX to be multiplied with SYN_T or SYN_S respectively.

DPLL_RCDT_SX_NOM
DPLL Reciprocal Value of the Expected Nominal Increment Duration of STATE(02846C$_H$) Reset Value: Table 104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RCDT_SX_NOM							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCDT_SX_NOM															
rw															

Field	Bits	Type	Description
RCDT_SX_NOM	23:0	rw	**Reciprocal value of nominal increment duration *2^32 while only the lower 24 bits are used** Calculated value; when an overflow occurs in the calculation, the value is set to 0xFFFFFF.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 104 Reset Values of DPLL_RCDT_SX_NOM

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	––00 0000$_H$	Cleared by state machine

28.20.12.60 Memory DPLL_RDT_T_ACT

DPLL Reciprocal Value of the Last Increment of TRIGGER

DPLL_RDT_T_ACT
DPLL Reciprocal Value of the Last Increment of TRIGGER(028470$_H$) **Reset Value: Table 105**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RDT_T_ACT							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT_T_ACT															
rw															

Field	Bits	Type	Description
RDT_T_ACT	23:0	rw	**Reciprocal value of last TRIGGER increment *2^32, only the lower 24 bits are used** The LSB is rounded up when the next truncated bit is 1. Calculated value; when an overflow occurs in the calculation, the value is set to 0xFFFFFF and the CRO bit in the DPLL_STATUS register is set (see **Section 28.20.12.30**).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 105 Reset Values of DPLL_RDT_T_ACT

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	––00 0000$_H$	Cleared by state machine

28.20.12.61 Memory DPLL_RDT_S_ACT

DPLL Reciprocal Value of the Last Increment of STATE

DPLL_RDT_S_ACT
DPLL Reciprocal Value of the Last Increment of STATE(028474$_H$) **Reset Value: Table 106**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								RDT_S_ACT							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT_S_ACT															
rw															

Field	Bits	Type	Description
RDT_S_ACT	23:0	rw	**Reciprocal value of last STATE increment *2^32, only the lower 24 bits are used** The LSB is rounded up when the next truncated bit is 1. Calculated value; when an overflow occurs in the calculation, the value is set to 0xFFFFFF and the CRO bit in the DPLL_STATUS register is set (see Section 28.20.12.30).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 106 Reset Values of DPLL_RDT_S_ACT

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.62Memory DPLL_DT_T_ACT

DPLL Duration of the Last TRIGGER Increment

DPLL_DT_T_ACT

DPLL Duration of the Last TRIGGER Increment (028478_H) **Reset Value: Table 107**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DT_T_ACT**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DT_T_ACT															
rw															

Field	Bits	Type	Description
DT_T_ACT	23:0	rw	**Calculated duration of the last TRIGGER increment** Calculated duration of the last increment; Value will be written into the corresponding RAM field, when all calculations for the considered increment are done and APT is valid.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 107 Reset Values of DPLL_DT_T_ACT

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.63Memory DPLL_DT_S_ACT

DPLL Duration of the Last STATE Increment

DPLL_DT_S_ACT
DPLL Duration of the Last STATE Increment (02847C$_H$) Reset Value: Table 108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DT_S_ACT**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DT_S_ACT															
rw															

Field	Bits	Type	Description
DT_S_ACT	23:0	rw	**Calculated duration of the last STATE increment** Calculated increment duration. Value will be written into the corresponding RAM field, when all calculations for the considered increment are done and APS is valid.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 108 Reset Values of DPLL_DT_S_ACT

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.64Memory DPLL_EDT_T

DPLL Difference of Prediction to Actual Value of the Last TRIGGER Increment

DPLL_EDT_T
DPLL Difference of Prediction to Actual Value of the Last TRIGGER Increment(028480$_H$) Reset Value: Table 109

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								EDT_T							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDT_T															
rw															

Field	Bits	Type	Description
EDT_T	23:0	rw	**Signed difference between actual value and a simple prediction of the last TRIGGER increment: sint24** Calculated error value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 109 Reset Values of DPLL_EDT_T

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.65Memory DPLL_MEDT_T

DPLL Weighted Difference of Prediction Errors of TRIGGER

DPLL_MEDT_T
DPLL Weighted Difference of Prediction Errors of TRIGGER(028484$_H$) Reset Value: Table 110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MEDT_T							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEDT_T															
rw															

Field	Bits	Type	Description
MEDT_T	23:0	rw	**Signed middle weighted difference between actual value and prediction of the last TRIGGER increments: sint24** Only calculated for SYT=1. Calculated medium error value, see **Section 28.20.6.2.6**. The value is calculated only after synchronization (SYT=1), and the update is suppressed for one increment when an unexpected missing TRIGGER is detected.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 110 Reset Values of DPLL_MEDT_T

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	$--00\ 0000_H$	Cleared by state machine

28.20.12.66Memory DPLL_EDT_S

DPLL Difference of Prediction to Actual Value of the Last STATE Increment

DPLL_EDT_S
DPLL Difference of Prediction to Actual Value of the Last STATE Increment(028488_H) **Reset Value:** **Table 111**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**EDT_S**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDT_S															
rw															

Field	Bits	Type	Description
EDT_S	23:0	rw	**Signed difference between actual value and prediction of the last STATE increment: sint24** Calculated error value, see **Section 28.20.6.3.5**.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 111 Reset Values of DPLL_EDT_S

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.67Memory DPLL_MEDT_S

DPLL Weighted Difference of Prediction Errors of STATE

DPLL_MEDT_S
DPLL Weighted Difference of Prediction Errors of STATE($02848C_H$) **Reset Value: Table 112**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MEDT_S							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEDT_S															
rw															

Field	Bits	Type	Description
MEDT_S	23:0	rw	**Signed middle weighted difference between actual value and prediction of the last STATE increments: sint24; only calculated for SYS=1** Calculated medium error value, see **Section 28.20.6.3.6**. The value is calculated only after synchronization (SYS=1), and the update is suppressed for one increment when an unexpected missing STATE is detected.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 112 Reset Values of DPLL_MEDT_S

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.68Memory DPLL_CDT_TX

DPLL Prediction of the Actual TRIGGER Increment Duration

DPLL_CDT_TX
DPLL Prediction of the Actual TRIGGER Increment Duration(028490$_H$) **Reset Value: Table 113**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CDT_TX**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDT_TX															
rw															

Field	Bits	Type	Description
CDT_TX	23:0	rw	**Calculated duration of the current TRIGGER increment** Calculated value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 113 Reset Values of DPLL_CDT_TX

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.69Memory DPLL_CDT_SX

DPLL Prediction of the Actual STATE Increment Duration

DPLL_CDT_SX
DPLL Prediction of the Actual STATE Increment Duration(028494$_H$) **Reset Value: Table 114**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CDT_SX**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDT_SX															
rw															

Field	Bits	Type	Description
CDT_SX	23:0	rw	**Calculated duration of the current STATE increment** Calculated value.

Field	Bits	Type	Description
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 114 Reset Values of DPLL_CDT_SX

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.70Memory DPLL_CDT_TX_NOM

DPLL Prediction of the Nominal TRIGGER Increment Duration

DPLL_CDT_TX_NOM
DPLL Prediction of the Nominal TRIGGER Increment Duration(028498$_H$) **Reset Value: Table 115**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CDT_TX_NOM**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDT_TX_NOM															
rw															

Field	Bits	Type	Description
CDT_TX_NOM	23:0	rw	**Calculated duration of the current nominal TRIGGER event** Calculated value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

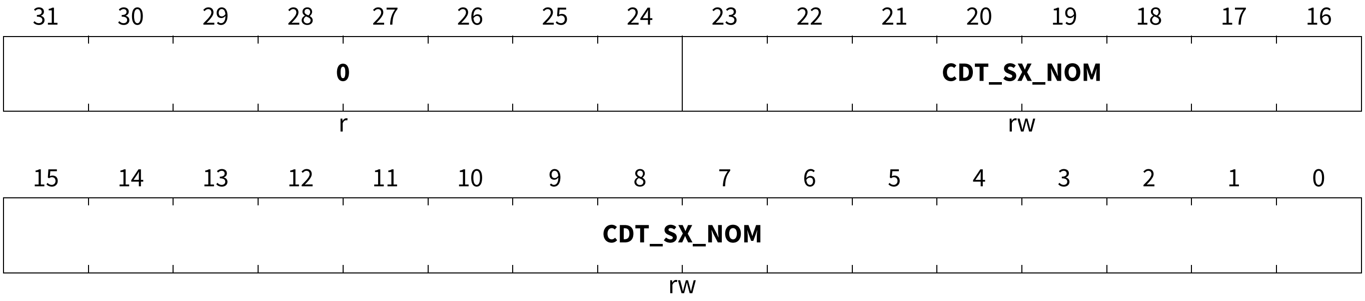
Table 115 Reset Values of DPLL_CDT_TX_NOM

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.71Memory DPLL_CDT_SX_NOM

DPLL Prediction of the Nominal STATE Increment Duration

DPLL_CDT_SX_NOM
DPLL Prediction of the Nominal STATE Increment Duration(02849C$_H$) **Reset Value: Table 116**



Field	Bits	Type	Description
CDT_SX_NOM	23:0	rw	**Calculated duration of the current nominal STATE event** Calculated value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

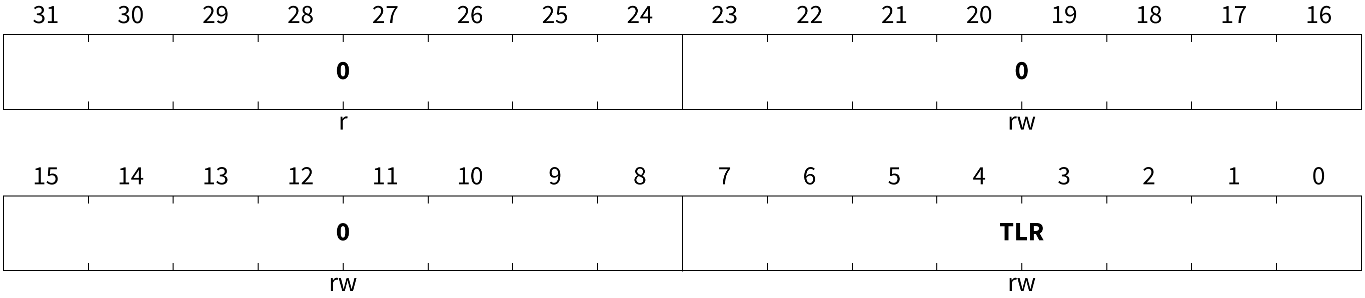
Table 116 Reset Values of DPLL_CDT_SX_NOM

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.72Memory DPLL_TLR

DPLL TRIGGER Locking Range

DPLL_TLR
DPLL TRIGGER Locking Range **(0284A0$_H$)** **Reset Value: Table 117**



Field	Bits	Type	Description
TLR	7:0	rw	**Value is to be multiplied with the last nominal TRIGGER duration in order to get the range for the next TRIGGER event without setting TOR in the DPLL_STATUS register** Multiply value with the last nominal increment duration and check violation; when TLR=0 don't perform the check.
0	23:8	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 117 Reset Values of DPLL_TLR

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.73Memory DPLL_SLR

DPLL STATE Locking Range

DPLL_SLR
DPLL STATE Locking Range (0284A4$_H$) **Reset Value: Table 118**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								**SLR**							
rw								rw							

Field	Bits	Type	Description
SLR	7:0	rw	**Value is to be multiplied with the last nominal STATE duration in order to get the range for the next STATE event without setting SOR in the DPLL_STATUS register** Multiply value with the last nominal increment duration and check violation; when SLR=0 don't perform the check.
0	23:8	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

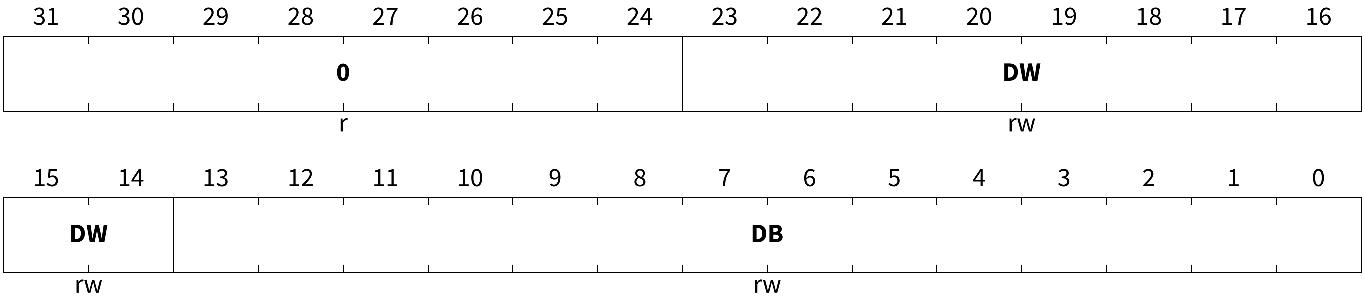
Table 118 Reset Values of DPLL_SLR

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.74Memory DPLL_PDT_[z]

DPLL Projected Increment Sum Relations for Action z

DPLL_PDT_z (z=0-31)
DPLL Projected Increment Sum Relations for Action z(028500$_H$+z*4) **Reset Value: Table 119**



Field	Bits	Type	Description
DB	13:0	rw	**Fractional part of relation between TRIGGER or STATE increments**
DW	23:14	rw	**Integer part of relation between TRIGGER or STATE increments** Definition of relation values between *TRIGGER* or *STATE* increments PDT[i] according to Equations DPLL-11 or DPLL-13 (i = 0-31).[1] The PDT[i] values for actions i=24...31 are only available for device 4 or 5.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

[1] The PDT[z] values for actions i=24...31 are not available for all devices.

Table 119 Reset Values of DPLL_PDT_z (z=0-31)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.75Memory DPLL_MLS1

DPLL Calculated Number of Sub-Pulses between two Nominal STATE Events for SMC = 0

DPLL_MLS1
DPLL Calculated Number of Sub-Pulses between two Nominal STATE Events for SMC = 0(0285C0$_H$) Reset Value: Table 120

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**						**MLS1**	
r								rw						rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MLS1															
rw															

Field	Bits	Type	Description
MLS1	17:0	rw	**Number of pulses between two STATE events** For SMC=0 the value of MLS1 is calculated once by the CPU for fixed values in the DPLL_CTRL_0 register by the formula MLS1 = ((MLT+1)*(TNU+1)/(SNU+1)) and set accordingly. For SMC=1 the value of MLS1 represents the number of pulses between two nominal *TRIGGER* events (to be set and updated by the CPU).
0	23:18	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 120 Reset Values of DPLL_MLS1

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	––00 0000$_H$	Cleared by state machine

28.20.12.76Memory DPLL_MLS2

DPLL Calculated Number of Sub-Pulses between two Nominal STATE Events for SMC = 1 and RMO = 1

DPLL_MLS2
DPLL Calculated Number of Sub-Pulses between two Nominal STATE Events for SMC = 1 and RMO = 1 (0285C4$_H$) **Reset Value: Table 121**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**						**MLS2**	
r								rw						rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MLS2															
rw															

Field	Bits	Type	Description
MLS2	17:0	rw	**Number of pulses between two STATE events (to be set and updated by the CPU)** Using adapt information and the missing *STATE* event information SYN_S, this value can be corrected for each increment automatically.
0	23:18	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 121 Reset Values of DPLL_MLS2

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	––00 0000$_H$	Cleared by state machine

28.20.12.77Memory DPLL_CNT_NUM_1

DPLL Number of Sub-Pulses of SUB_INC1 in Continuous Mode

DPLL_CNT_NUM_1
DPLL Number of Sub-Pulses of SUB_INC1 in Continuous Mode(0285C8$_H$) **Reset Value: Table 122**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CNT_NUM_1**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT_NUM_1															
rw															

Field	Bits	Type	Description
CNT_NUM_1	23:0	rw	**Counter for number of SUB_INC1 pulses** Number of pulses in continuous mode for a nominal increment in normal and emergency mode for SUB_INC1, given and updated by CPU only. Count value for continuous mode.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 122 Reset Values of DPLL_CNT_NUM_1

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	––00 0000$_H$	Cleared by state machine

28.20.12.78Memory DPLL_CNT_NUM_2

DPLL Number of Sub-Pulses of SUB_INC2 in Continuous Mode

DPLL_CNT_NUM_2
DPLL Number of Sub-Pulses of SUB_INC2 in Continuous Mode(0285CC$_H$) **Reset Value: Table 123**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CNT_NUM_2**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT_NUM_2															
rw															

Field	Bits	Type	Description
CNT_NUM_2	23:0	rw	**Counter for number of SUB_INC2 pulses** Number of pulses in continuous mode for a nominal increment in normal and emergency mode for SUB_INC2, given and updated by CPU only. Count value for continuous mode.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 123 Reset Values of DPLL_CNT_NUM_2

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	––00 0000$_H$	Cleared by state machine

28.20.12.79Memory DPLL_PVT

DPLL Plausibility Value of Next TRIGGER Slope

DPLL_PVT

DPLL Plausibility Value of Next TRIGGER Slope(0285D0$_H$) **Reset Value: Table 124**

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">0</td><td colspan="8">PVT</td></tr>
<tr><td colspan="8">r</td><td colspan="8">rw</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">PVT</td></tr>
<tr><td colspan="16">rw</td></tr>
</table>

Field	Bits	Type	Description
PVT	23:0	rw	**Plausibility value of next active TRIGGER slope** The meaning of the value depends on the value of the PIT value in the DPLL_CTRL_1 register. For PIT=0: the number of SUB_INC1 pulses to be waited for until a next active *TRIGGER* event is accepted. For PIT=1: PVT is to be multiplied with the **last nominal increment time DT_T_ACT** and divided by 1024 and reduced to a 24 bit value in order to get the time to be waited for until the next active *TRIGGER* event is accepted. The wait time must be exceeded for an active slope. *Note: When an active TRIGGER slope is detected while the wait condition is not fulfilled the interrupt PWI is generated. Please note, that the SGE1 must be set, when PIT=0 in order to provide the necessary SUB_INC1 pulses for checking. After an unexpected missing TRIGGER the plausibility check is suppressed for the following increment. In case of direction change the PVT value is automatically set to zero in order to deactivate the check.*
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 124 Reset Values of DPLL_PVT

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.80Memory DPLL_PSTC

DPLL Actual Calculated Position Stamp of TRIGGER

DPLL_PSTC
DPLL Actual Calculated Position Stamp of TRIGGER(0285E0$_H$) **Reset Value: Table 125**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								PSTC							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSTC															
rw															

Field	Bits	Type	Description
PSTC	23:0	rw	**Calculated position stamp of last TRIGGER input** Value is set by the DPLL and can be updated by the CPU when filter values are to be considered for the exact position (see **DPLL_STATUS** and **DPLL_CTRL** registers for explanation of the status and control bits used). For each active slope of *TRIGGER* in normal mode. When FTD=0: PSTC is set from actual position value, for the first active *TRIGGER* event (no filter delay considered) the CPU must update the value once, taking into account the filter value. When FTD=1: PSTC is incremented at each *TRIGGER* event by SMC=0: (MLT+1)*(SYN_T) +PD; while PD=0 for AMT=0 SMC=1: (MLS1)*(SYN_T) +PD; while PD=0 for AMT=0
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 125 Reset Values of DPLL_PSTC

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.81Memory DPLL_PSSC

DPLL Actual Calculated Position Stamp of STATE

DPLL_PSSC

DPLL Actual Calculated Position Stamp of STATE(0285E4$_H$) **Reset Value: Table 126**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**PSSC**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSSC															
rw															

Field	Bits	Type	Description
PSSC	23:0	rw	**Calculated position stamp for the last STATE input** First value is set by the DPLL and can be updated by the CPU when the filter delay is to be considered. For each active slope of *STATE* in emergency mode. When FSD=0: PSSC is set from actual position value(no filter delay considered), the CPU must update the value once, taking into account the filter value When FSD=1: at each active slope of *STATE* (PD_S_store=0 for AMS=0): SMC=0: add MLS1*(SYN_S) + PD_S_store; SMC=1: add MLS2*(SYN_S) + PD_S_store;
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 126 Reset Values of DPLL_PSSC

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.82Memory DPLL_PSTM

DPLL Measured Position Stamp at Last TRIGGER Input

*Note: The LSB address is determined using the SWON_T value in the OSW register (see **Section 28.20.12.8**).*

DPLL_PSTM

DPLL Measured Position Stamp at Last TRIGGER Input(0285E8$_H$) **Reset Value: Table 127**

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">0</td><td colspan="8">PSTM</td></tr>
<tr><td colspan="8">r</td><td colspan="8">rw</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">PSTM</td></tr>
<tr><td colspan="16">rw</td></tr>
</table>

Field	Bits	Type	Description
PSTM	23:0	rw	**Position stamp of TRIGGER, measured** Measured position stamp of last active *TRIGGER* input. Store the value TBU_TS1 when an active TRIGGER event occurs. The value of PSTM is invalid for (RMO=1 and SMC=0).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 127 Reset Values of DPLL_PSTM

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.83Memory DPLL_PSTM_OLD

DPLL Measured Position Stamp at Last but One TRIGGER Input

Note: The LSB address is determined using the SWON_T value in the OSW register (see ***Section 28.20.12.8****).*

DPLL_PSTM_OLD

DPLL Measured Position Stamp at Last but One TRIGGER Input(0285EC$_H$) **Reset Value: Table 128**

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">0</td><td colspan="8">PSTM_OLD</td></tr>
<tr><td colspan="8">r</td><td colspan="8">rw</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">PSTM_OLD</td></tr>
<tr><td colspan="16">rw</td></tr>
</table>

Field	Bits	Type	Description
PSTM_OLD	23:0	rw	**Last but one position stamp of TRIGGER, measured** Measured position stamp of last but one active *TRIGGER* input. Last PSTM value: see explanation of PSTM.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 128 Reset Values of DPLL_PSTM_OLD

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.84Memory DPLL_PSSM

DPLL Measured Position Stamp at Last STATE Input

Note: The LSB address is determined using the SWON_S value in the OSW register (see ***Section 28.20.12.8****).*

DPLL_PSSM

DPLL Measured Position Stamp at Last STATE Input($0285F0_H$) **Reset Value: Table 129**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**PSSM**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSSM															
rw															

Field	Bits	Type	Description
PSSM	23:0	rw	**Position stamp of STATE, measured** Measured position stamp of last active STATE input. Store the value TBU_TS1 or TBU_TS2, respectively, at the moment when an active STATE event occurs. The value of PSSM is invalid for (RMO=0 and SMC=0).
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 129 Reset Values of DPLL_PSSM

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.85Memory DPLL_PSSM_OLD

DPLL Measured Position Stamp at Last but One STATE Input

Note: The LSB address is determined using the SWON_S value in the OSW register (see ***Section 28.20.12.8****).*

DPLL_PSSM_OLD
DPLL Measured Position Stamp at Last but One STATE Input($0285F4_H$) **Reset Value: Table 130**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**PSSM_OLD**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSSM_OLD															
rw															

Field	Bits	Type	Description
PSSM_OLD	23:0	rw	**Last but one position stamp of STATE, measured** Measured position stamp of last but one active STATE input. Last PSSM value: see explanation of PSSM.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 130 Reset Values of DPLL_PSSM_OLD

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.86Memory DPLL_NMB_T

DPLL Number of Pulses to be Sent in Normal Mode

DPLL_NMB_T
DPLL Number of Pulses to be Sent in Normal Mode(0285F8$_H$) **Reset Value: Table 131**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMB_T															
rw															

Field	Bits	Type	Description
NMB_T	15:0	rw	**Number of pulses for TRIGGER** Calculated number of pulses in normal mode for the current TRIGGER increment. Calculated pulse number.
0	23:16	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 131 Reset Values of DPLL_NMB_T

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	--00 0000$_H$	Cleared by state machine

28.20.12.87Memory DPLL_NMB_S

DPLL Number of Pulses to be Sent in Emergency Mode

DPLL_NMB_S
DPLL Number of Pulses to be Sent in Emergency Mode(0285FC$_H$) **Reset Value: Table 132**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**				**NMB_S**			
r								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMB_S															
rw															

Field	Bits	Type	Description
NMB_S	19:0	rw	**Number of pulses for STATE** Calculated number of pulses in emergency mode for the current STATE increment. Calculated pulse number.
0	23:20	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 132 Reset Values of DPLL_NMB_S

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.88Memory DPLL_RDT_S[i]

DPLL Reciprocal Values of the Nominal STATE i Increment Duration in FULL_SCALE

Note: If DPLL_CTRL_11.STATE_EXT is set, this memory range is not used by the DPLL, but emulated outside the DPLL. The DPLL will access the MCS to DPLL interface 18.15 and will expect data to be correctly stored there. This means in fact, that the handling of the RDT_S values has to be done outside, in the MCS integrated in the same cluster.

DPLL_RDT_Si (i=0-63)
DPLL Reciprocal Values of the Nominal STATE i Increment Duration in FULL_SCALE(028600_H+i*4) Reset Value: Table 133

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**RDT_S**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT_S															
rw															

Field	Bits	Type	Description
RDT_S	23:0	rw	**Reciprocal difference time of STATE** Nominal reciprocal value of the number of time stamp clocks measured in the corresponding increment $*2^{32}$ while only the lower 24 bits are used; no gap considered. The LSB is rounded up when the next truncated bit is 1. There are 2*(SNU+1-SYN_NS) entries for SYSF=0 or 2*(SNU+1)-SYN_NS entries for SYSF=1 respectively.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 133 Reset Values of DPLL_RDT_Si (i=0-63)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX $XXXX_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1BC	$--00\ 0000_H$	Cleared by state machine

28.20.12.89Memory DPLL_TSF_S[i]

DPLL Time Stamp Values of the Nominal STATE i Events in FULL_SCALE

Note: *If DPLL_CTRL_11.STATE_EXT is set, this memory range is not used by the DPLL, but emulated outside the DPLL. The DPLL will access the MCS to DPLL interface 18.15 and will expect data to be correctly stored there. This means in fact, that the handling of the TSF_S values has to be done outside, in the MCS integrated in the same cluster.*

DPLL_TSF_Si (i=0-63)
DPLL Time Stamp Values of the Nominal STATE i Events in FULL_SCALE(028700_H+i*4) Reset Value: Table 134

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TSF_S**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSF_S															
rw															

Field	Bits	Type	Description
TSF_S	23:0	rw	**Time stamp field of STATE** Time stamp value of each active STATE event. There are 2* (SNU+1) entries.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 134 Reset Values of DPLL_TSF_Si (i=0-63)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX_H	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000_H	Cleared by state machine

28.20.12.90Memory DPLL_ADT_S[i]

DPLL Adapt and Profile Values of the STATE i Increments in FULL_SCALE

Note: *If DPLL_CTRL_11.STATE_EXT is set, this memory range is not used by the DPLL, but emulated outside the DPLL. The DPLL will access the MCS to DPLL interface 18.15 and will expect data to be correctly stored there. This means in fact, that the handling of the ADT_S values has to be done outside, in the MCS integrated in the same cluster.*

DPLL_ADT_Si (i=0-63)

DPLL Adapt and Profile Values of the STATE i Increments in FULL_SCALE(028800_H+i*4) Reset Value: Table 135

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">0</td><td colspan="2">0</td><td colspan="6">NS</td></tr>
<tr><td colspan="8">r</td><td colspan="2">rw</td><td colspan="6">rw</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">PD_S</td></tr>
<tr><td colspan="16">rw</td></tr>
</table>

Field	Bits	Type	Description
PD_S	15:0	rw	**Physical deviation of STATE** Adapt values for each *STATE* increment in FULL_SCALE (sint16); This value represents the number of pulses to be added to the correspondent nominal increment. The absolute value of a negative PD_S must not exceed MLS1 or MLS2 respectively. The PD value does mean the number of SUB_INC1 pulses per nominal tooth to be added to NS*((MLS1/2+1) + PD_S);
NS	21:16	rw	**Number of STATEs** Number of nominal STATE parts in the corresponding increment. There are 2*(SNU+1-SYN_NS) entries for SYSF=0 or 2*(SNU+1)-SYN_NS entries for SYSF=1 respectively.
0	23:22	rw	**Not used** Must be written to zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 135 Reset Values of DPLL_ADT_Si (i=0-63)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.91Memory DPLL_DT_S[i]

DPLL Nominal STATE i Increment Duration in FULL_SCALE

Note: *If DPLL_CTRL_11.STATE_EXT is set, this memory range is not used by the DPLL, but emulated outside the DPLL. The DPLL will access the MCS to DPLL interface chapter "MCS to DPLL Register description" and will expect data to be correctly stored there. This means in fact, that the handling of the DT_S values has to be done outside, in the MCS integrated in the same cluster.*

DPLL_DT_Si (i=0-63)
DPLL Nominal STATE i Increment Duration in FULL_SCALE(028900$_H$+i*4) **Reset Value: Table 136**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DT_S							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DT_S															
rw															

Field	Bits	Type	Description
DT_S	23:0	rw	**Difference time of STATE** Nominal increment duration values for each *STATE* increment in FULL_SCALE (considering no gap). There are 2*(SNU+1-SYN_NS) entries for SYSF=0 or 2*(SNU+1)-SYN_NS entries for SYSF=1 respectively.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 136 Reset Values of DPLL_DT_Si (i=0-63)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1BC	--00 0000$_H$	Cleared by state machine

28.20.12.92Register DPLL_TSAC[z]

DPLL Calculated Time Value to start Action z Register

DPLL_TSACz (z=0-31)

DPLL Calculated Time Value to start Action z Register(028E00$_H$+z*4) Application Reset Value: 007F FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								TSAC							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSAC															
rw															

Field	Bits	Type	Description
TSAC	23:0	rw	**Calculated time stamp for ACTION_z** This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.93Register DPLL_PSAC[z]

DPLL ACTION Position/Value Action z Request Register

DPLL_PSACz (z=0-31)

DPLL ACTION Position/Value Action z Request Register(028E80$_H$+z*4) Application Reset Value: 007F FFFF$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								PSAC							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSAC															
rw															

Field	Bits	Type	Description
PSAC	23:0	rw	**Calculated position value for the start of ACTION_z in normal or emergency mode according to equations DPLL-17 or DPLL-20, respectively** This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.94Register DPLL_ACB_[z]

DPLL Control Bits Register z for up to 32 Actions

DPLL_ACB_z (z=0-7)
DPLL Control Bits Register z for up to 32 Actions(028F00$_H$+z*4) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			**ACB_3**					**0**			**ACB_2**				
r			rw					r			rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			**ACB_1**					**0**			**ACB_0**				
r			rw					r			rw				

Field	Bits	Type	Description
ACB_0	4:0	rw	**Action Control Bits of ACTION_z** Reflects ACT_D[i](52:48), i=4*z. When DPLL_CTRL_11.ACBU = '0': ACB_0[4:0] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. When DPLL_CTRL_11.ACBU = '1': ACB_0[4:2] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. ACB_0[1]= '1' is used as input signal to control if "action in past" shall be checked based on position information. ACB_0[1] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far. ACB_0[0] is used as input signal to control if "action in past" shall be checked based on time information. ACB_0[0] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far. This value can only be written via AEI-interface when the DPLL is disabled.

Field	Bits	Type	Description
ACB_1	12:8	rw	**Action Control Bits of ACTION_(i + 1)** Reflects ACT_D[i+1](52:48), i=4*z. When DPLL_CTRL_11.ACBU = '0': ACB_1[4:0] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. When DPLL_CTRL_11.ACBU = '1': ACB_1[4:2] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation.ACB_1[1]= '1' is used as input signal to control if "action in past" shall be checked based on position information. ACB_1[1] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far.ACB_1[0] is used as input signal to control if "action in past" shall be checked based on time information. ACB_1[0] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far.This value can only be written via AEI-interface when the DPLL is disabled.
ACB_2	20:16	rw	**Action Control Bits of ACTION_(i + 2)** Reflects ACT_D[i+2](52:48), i=4*z. When DPLL_CTRL_11.ACBU = '0': ACB_2[4:0] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. When DPLL_CTRL_11.ACBU = '1': ACB_2[4:2] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. ACB_2[1]= '1' is used as input signal to control if "action in past" shall be checked based on position information. ACB_2[1] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far. ACB_2[0] is used as input signal to control if "action in past" shall be checked based on time information. ACB_2[0] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far. This value can only be written via AEI-interface when the DPLL is disabled.

Field	Bits	Type	Description
ACB_3	28:24	rw	**Action Control Bits of ACTION_(i + 3)** Reflects ACT_D[i+3](52:48), i=4*z. When DPLL_CTRL_11.ACBU = '0': ACB_3[4:0] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. When DPLL_CTRL_11.ACBU = '1': ACB_3[4:2] are taken as received by ARU interface and are transmitted unchanged as result of action (PMT) calculation. ACB_3[1]= '1' is used as input signal to control if "action in past" shall be checked based on position information. ACB_3[1] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far. ACB_3[0] is used as input signal to control if "action in past" shall be checked based on time information. ACB_3[0] is written to '1' if action channel has reached "action in past" condition after action has been calculated, written to '0' if action has not reached "past" so far. This value can only be written via AEI-interface when the DPLL is disabled.
0	7:5, 15:13, 23:21, 31:29	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.95Register DPLL_CTRL_11

DPLL Control Register 11

DPLL_CTRL_11
DPLL Control Register 11 **($028F20_H$)** **Application Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WACBU	WSTATE_EXT	WPCMF2_INCCNT_B	WINCF2	WFSYL2	WPCMF2	WERZ2	WSIP2	WADS	WADT	WPCMF1_INCCNT_B	WINCF1	WFSYL1	WPCMF1	WERZ1	WSIP1
rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACBU	STATE_EXT	PCMF2_INCCNT_B	INCF2	FSYL2	PCMF2	ERZ2	SIP2	ADS	ADT	PCMF1_INCCNT_B	INCF1	FSYL1	PCMF1	ERZ1	SIP1
rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SIP1	0	rw	**Simplified increment prediction in normal mode and for the first engine in the case SMC=1** For the first increment after setting SIP1 from 0 to 1, the value of DT_T_ACT is replaced by the value of the DT_T_START register. This results in a CDT_TX value which is equal to DT_T_START. Please notice that this DT_T_Start value must be always > 256. *Note: The value of SIP1 influences only the increment prediction CDT_TX and when NUTE-VTN=1. The calculation of QDT_T itself is not influenced by the SIP1 bit. The value of SIP1 can be only be written when WSIP1=1.* *Note: When SIP1=1 is set, the first pulses of the subincrement generator are not generated with highest frequency for the first increment (DPLL_STATUS.FTD = 0, DPLL_CTRL_1.SGE1=1).* 0_B Increment prediction calculation; the current increment duration is calculated using the relation between increment durations in the past, like explained by the corresponding equations 1_B Increment prediction continuation; in this mode, for the increment prediction value calculation of CDT_TX the value of QDT_T is replaced by 1 for all calculations when NUTE-VTN=1; in the other case, the value of SIP1 is ignored, and the calculation is performed like for SIP1=0
ERZ1	1	rw	**Error is assumed as zero in normal mode and for the first engine for SMC=1** The calculation of EDT_T and MEDT_T is performed independently from the ERZ1 value in all modes without any influence to the MEDT_T value itself. The ERZ1 value influences the use of MEDT_T in normal mode and for SMC=1. The value of ERZ1 can be only written when WERZ1=1. 0_B The MEDT_T and MEDT_S values are considered as provided in the corresponding equations 1_B Instead of using MEDT_T, the value '0' is used in the corresponding equations

Field	Bits	Type	Description
PCMF1	2	rw	**Pulse correction mode fast for INC_CNT1** The fast pulse generation is performed immediately within the current increment. MPVAL1 must be positive integers for the fast pulse correction mode - in the case of negative values, the correction is suppressed, and the FPCE (fast pulse correction error) bit in the DPLL_STATUS register is set, causing the EI (error interrupt) when enabled. The setting of PCMF1 prevents the transfer of control bits PCM1 to the corresponding shadow registers with an active input event, and prevents therefore the distribution of the MPVAL1 values over the current or next increment. The MPVAL1 pulses are sent with the fast clock CMU_CLK0 by the rapid pulse generator RPCUx (see chapter 18.8.3.6 of specification v3.0) triggered in the state 6/26 or 18/38 of the state machines (see **Section 28.20.8.6.1**), respectively. The INC_CNT1 is incremented by MPVAL1, respectively. When taken the MPVAL1 value to RPCUx and INC_CNT1, the PCM1 bit is reset immediately, and after that, also the PCMF1 bit. The value of PCMF1 can be only written when WPCMF1=1. Be careful when using the fast pulse correction during a direction change. Because of sending the correction pulses before, during or after the direction change recognition, the result is typically unpredictable. No automatic correction of the fast correction pulses is provided. The necessary corrections must be performed on responsibility of the user. 0_B No fast update of pulses, provided by MPVAL1 1_B When PCM1 is set while PCMF1=1, the pulses provided by MPVAL1 are sent using the rapid pulse generator RPCUx, without waiting for a new input event
FSYL1	3	rw	**Force Synchronization Loss of LOCK1** The synchronization loss resets SYT/SYS and prevents the use of profiles, respectively. The above described effect for FSYL1=1 is only active when WFSYL1=1 simultaneously. 0_B No force of synchronization loss 1_B Reset LOCK1 and reset SYT in normal mode, and for SMC=1, reset SYS in emergency mode

Field	Bits	Type	Description
INCF1	4	rw	**INC_CNT1 fast correction** The calculation of ADD_IN for the SUB_INC generation is performed without adding the 0.5 value to NMB_T/S in equations DPLL_25 ff. The signal RESET_SIGx of the pulse generator is activated for each new active input slope, in order to reset the register values. *Note: The INCF1 value can be only written when WINCF1=1.* *Note: The INCF1 bit should only be written when DPLL_CTRL_1.DEN = '0' (DPLL disabled) to prevent generation of wrong number of sub increments.* 0_B The calculation of a new INC_CNT1 is performed after an active slope was detected and the plausibility check was performed 1_B The calculation of a new INC_CNT1 is prepared before an active slope is detected; the plausibility check is supported by an additional HW checker in order to get the decision earlier, and after this decision, the pulse generator for SUB_INC1 starts immediately sending out pulses
PCMF1_INCCNT_B	5	rw	**No increment of INC_CNT1 when PCMF1 active (automatic end mode)** The PCMF1_INCCNT_B value can be only written when WPCMF1_INCCNT_B =1. 0_B When fast pulse correction is done by PCM1, PCMF1, the MPVAL1 value is as well added to the INC_CNT1 1_B Do not add MPVAL1 value to the INC_CNT1 register when fast pulse correction is done by PCM1 or PCMF1. This means that just fast pulses are done by decrementing current content of INC_CNT1 register as long as INC_CNT1 is not zero (automatic end mode). The number of pulses (MPVAL1) shall be sufficiently smaller than INC_CNT1 when MPVAL1 is written.
ADT	6	r	**Correction of DT_T_ACTUAL,CDT_TX_nom_corr by PD_T** 0_B No correction of DT_T_ACTUAL, CDT_TX_nom_corr by physical deviation (PD_T) defined in profile of TRIGGER processing unit 1_B Correction of DT_T_ACTUAL, CDT_TX_nom_corr by physical deviation (PD_T) defined in profile of TRIGGER processing unit
ADS	7	r	**Correction of DT_S_ACTUAL, CDT_SX_nom_corr by PD_S** 0_B No correction of DT_S_ACTUAL, CDT_SX_nom_corr by physical deviation (PD_S) defined in profile of STATE processing unit 1_B Correction of DT_S_ACTUAL, CDT_SX_nom_corr by physical deviation (PD_S) defined in profile of STATE processing unit

Field	Bits	Type	Description
SIP2	8	rw	**Simplified increment prediction in emergency mode and for the second engine in the case RMO=1** For the first increment after setting SIP2 from 0 to 1, the value of DT_S_ACT is replaced by the value of the DT_S_START register. This results in a CDT_SX value which is equal to DT_S_START. Please notice that this DT_S_START value must be always > 256. The value of SIP2 influences only the increment prediction and error accumulation when NUSE-VSN=1. The calculation of QDT_S itself is not influenced by the SIP2 bit. The value of SIP2 can be only written when WSIP2=1. 0_B Increment prediction calculation; the current increment duration CDT_SX is calculated using the relation between increments duration in the past, like explained by the corresponding equations 1_B Increment prediction value calculation CDT_SX; the value of QDT_S is replaced by 1 for all calculations when NUSE-VSN=1; in the other case, the value of SIP2 is ignored, and the calculation is performed like for SIP2=0
ERZ2	9	rw	**Error is assumed as zero in emergency mode and for the second engine for SMC=1** The calculation of EDT_S and MEDT_S is performed independently from the ERZ2 value in all modes, without any influence to the MEDT_S value itself. The ERZ2 value influences the use of MEDT_S in emergency mode and for SMC=1 with RMO=1. The value of ERZ2 can be only written when WERZ2=1. 0_B The MEDT_S value is considered as provided in the corresponding equations 1_B Instead of using MEDT_S, the value '0' is used in the corresponding equations

Field	Bits	Type	Description
PCMF2	10	rw	**Pulse correction mode fast for INC_CNT2** The fast pulse generation is performed immediately within the current increment. MPVAL2 must be positive integers for the fast pulse correction mode - in the case of negative values, the correction is suppressed, and the FPCE (fast pulse correction error) bit in the DPLL_STATUS register is set, causing the EI (error interrupt) when enabled. The setting of PCMF2 prevents the transfer of control bits PCM2 to the corresponding shadow registers with an active input event, and therefore prevents the distribution of the MPVAL1 values over the current or next increment. The MPVAL2 pulses are sent with the fast clock CMU_CLK0 by the rapid pulse generator RPCUx (of specification v3.0) triggered in the state 6/26 or 18/38 of the state machines. The INC_CNT2 is incremented by MPVAL2, respectively. When taken the MPVAL2 value to RPCUx and INC_CNT2, the PCM2 bit is reset immediately, and after that also the PCMF2 bit. The value of PCMF2 can be only written when WPCMF2=1. Be careful when using the fast pulse correction during a direction change. Because of sending the correction pulses before, during or after the direction change recognition, the result is typically unpredictable. No automatic correction of the fast correction pulses is provided. The necessary corrections must be performed on responsibility of the user. 0_B No fast update of pulses provided by MPVAL2 1_B When PCM2 is set while PCMF2=1, the pulses provided by MPVAL2 are sent using the rapid pulse generator RPCUx, without waiting for a new input event
FSYL2	11	rw	**Force Synchronization Loss of LOCK2** The synchronization loss resets SYS and prevents the use of profiles respectively. The above described effect for FSYL2=1 is only active when WFSYL2=1 simultaneously. 0_B No force of synchronization loss 1_B Reset LOCK2 and reset SYS in emergency mode and for SMC=1
INCF2	12	rw	**INC_CNT2 fast** The INCF2 value can be only written when WINCF2=1. 0_B The calculation of a new INC_CNT2 is performed after an active slope was detected and the plausibility check was performed 1_B The calculation of a new INC_CNT2 is prepared before an active slope is detected; the plausibility check is supported by an additional HW checker in order to get the decision earlier, and after this decision, the pulse generator for SUB_INC2 starts immediately sending out pulses. The calculation of ADD_IN for the SUB_INC generation is performed without adding the 0.5 value to NMB_S in equations DPLL_25 ff. The signal RESET_SIGx of the pulse generator is activated for each new active input slope in order to reset the register values.

Field	Bits	Type	Description
PCMF2_INCCNT_B	13	rw	**No increment of INC_CNT2 when PCMF2 active (automatic end mode)** The PCMF2_INCCNT_B value can be only written when WPCMF2_INCCNT_B=1. 0_B Add MPVAL2 value is as well added to the INC_CNT2 when fast pulse correction is done by PCM2 or PCMF2 1_B Do not add MPVAL2 value to the INC_CNT2 register when fast pulse correction is done by PCM2 or PCMF2. This means, that just fast pulses are done by decrementing current content of INC_CNT2 register as long as INC_CNT2 is not zero (automatic end mode). The number of pulses (MPVAL2) shall be sufficiently smaller than INC_CNT2 when MPVAL2 is written.
STATE_EXT	14	rw	**Use of STATE engine extension** The STATE_EXT value can be only written when WSTATE_EXT=1 and the DPLL is disabled. See 18.10 for a further explanation. If this bit shall be modified during operation, a software reset of the DPLL module is strongly recommended. A RAM initialisation should also be considered depending on the given application case. 0_B STATE extension is not considered 1_B STATE extension is enabled for up to 128 STATE events
ACBU	15	rw	**ACB use; the ACB values of PMTR are used to decide if an action is in the past** Return ACB values together with actions as zero, when the actions are in the future; Set ACB[1]=1, when calculated position value is in the past and the ACB[1] of PMTR was 1. Set ACB[0]=1, when calculated time value is in the past and the ACB[0] of PMTR was 1. The value of ACBU can be only written when WACBU=1. 0_B ACB values of PMTR are not considered in DPLL; the decision, whether an action is in the past, is made considering the calculated time value 1_B ACB values of PMTR are considered in DPLL as follows: If ACB[1] = 1, consider whether the calculated position value of the corresponding action is in the past if ACB[0] = 1; consider whether the calculated time value of the corresponding action is in the past; ACB[1] and ACB[0] can be set also simultaneously to 1
WSIP1	16	rw	**Write enable for simplified increment prediction 1** Enable writing. 0_B Writing to SIP1 is not enabled 1_B Writing to SIP1 is enabled
WERZ1	17	rw	**Write enable for error zero 1** Enable writing. 0_B Writing to ERZ1 is not enabled 1_B Writing to ERZ1 is enabled

Field	Bits	Type	Description
WPCMF1	18	rw	**Write enable for pulse correction mode fast 1** Enable writing. 0_B Writing to PCM1 is not enabled 1_B Writing to PCM1 is enabled
WFSYL1	19	rw	**Write enable for force synchronization loss 1** Enable writing. 0_B Writing to FSYL1 is not enabled 1_B Writing to FSYL1 is enabled
WINCF1	20	rw	**Write enable for INC_CNT1 fast** Enable writing. 0_B Writing to INCF1 is not enabled 1_B Writing to INCF1 is enabled
WPCMF1_INCCNT_B	21	rw	**Write enable of PCMF1_INCCNT_B** Enable writing. 0_B Writing to PCMF1_INCCNT_B is not enabled 1_B Writing to PCMF1_INCCNT_B is enabled
WADT	22	r	**Write enable of ADT** Enable writing. 0_B Writing to ADT is not enabled 1_B Writing to ADT is enabled
WADS	23	r	**Write enable of ADS** Enable writing. 0_B Writing to ADS is not enabled 1_B Writing to ADS is enabled
WSIP2	24	rw	**Write enable for simplified increment prediction 2** Enable writing. 0_B Writing to SIP2 is not enabled 1_B Writing to SIP2 is enabled
WERZ2	25	rw	**Write enable for error zero 2** Enable writing. 0_B Writing to ERZ2 is not enabled 1_B Writing to ERZ2 is enabled
WPCMF2	26	rw	**Write enable for pulse correction mode fast 2** Enable writing. 0_B Writing to PCMF2 is not enabled 1_B Writing to PCMF2 is enabled
WFSYL2	27	rw	**Write enable for force synchronization loss 2** Enable writing. 0_B Writing to FSYL2 is not enabled 1_B Writing to FSYL2 is enabled
WINCF2	28	rw	**Write enable for INC_CNT2 fast** Enable writing. 0_B Writing to INCF2 is not enabled 1_B Writing to INCF2 is enabled

Field	Bits	Type	Description
WPCMF2_INCCNT_B	29	rw	**Write enable of PCMF2_INCCNT_B** 0_B Writing to PCMF2_INCCNT_B is not enabled 1_B Writing to PCMF2_INCCNT_B is enabled
WSTATE_EXT	30	rw	**Write enable of STATE_EXT** 0_B Writing to STATE_EXT is not enabled 1_B Writing to STATE_EXT is enabled
WACBU	31	rw	**Write enable for ACB use** The ACB values of PMTR are used to decide whether an action is in the past. Enable writing. 0_B Writing to ACBU is not enabled 1_B Writing to ACBU is enabled

28.20.12.96Register DPLL_THVAL2

DPLL Immediate THVAL Value Register

DPLL_THVAL2

DPLL Immediate THVAL Value Register **($028F24_H$)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**THVAL**							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THVAL															
r															

Field	Bits	Type	Description
THVAL	23:0	r	**Measured last pulse time from active to inactive slope of TRIGGER after correction of input slope filter delays** This value is available immediately after the inactive slope of TRIGGER. The measured value considers all input slope filter delays. From the received input the corresponding filter delays are subtracted before the time stamp difference of active and inactive slope is calculated.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.97Register DPLL_TIDEL

DPLL Additional TRIGGER Input Delay Register

DPLL_TIDEL
DPLL Additional TRIGGER Input Delay Register (028F28$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TIDEL**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIDEL															
rw															

Field	Bits	Type	Description
TIDEL	23:0	rw	**TRIGGER input delay** Transmit this value with each active TRIGGER slope into a shadow register. Subtract this shadow register value from each TRIGGER time stamp (active and inactive slope). This feature is always active and cannot be disabled by a control bit.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.98Register DPLL_SIDEL

DPLL Additional STATE Input Delay Register

DPLL_SIDEL
DPLL Additional STATE Input Delay Register (028F2C$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**SIDEL**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIDEL															
rw															

Field	Bits	Type	Description
SIDEL	23:0	rw	**STATE input delay** Transmit this value with each active STATE slope into a shadow register. Subtract this shadow register value from each STATE time stamp (active and inactive slope). This feature is always active and cannot be disabled by a control bit.

Field	Bits	Type	Description
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.99Register DPLL_CTN_MIN

DPLL Minimum CDT_T Nominal Value Register

DPLL_CTN_MIN
DPLL Minimum CDT_T Nominal Value Register (028F6C$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CTN_MIN**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTN_MIN															
rw															

Field	Bits	Type	Description
CTN_MIN	23:0	rw	**CDT_T_NOM min value** Use this register value as CDT_T_NOM value when the calculated value for the nominal increment prediction of TRIGGER is less than the register value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.100Register DPLL_CTN_MAX

DPLL Maximum CDT_T Nominal Value Register

DPLL_CTN_MAX
DPLL Maximum CDT_T Nominal Value Register (028F70$_H$) **Application Reset Value: 00FF FFFF$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CTN_MAX**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTN_MAX															
rw															

Field	Bits	Type	Description
CTN_MAX	23:0	rw	**CDT_T_NOM max value** Use this register value as CDT_T_NOM value when the calculated value for the nominal increment prediction of TRIGGER is greater than the register value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.101Register DPLL_CSN_MIN

DPLL Minimum CDT_S Nominal Value Register

DPLL_CSN_MIN
DPLL Minimum CDT_S Nominal Value Register (028F74$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CSN_MIN**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSN_MIN															
rw															

Field	Bits	Type	Description
CSN_MIN	23:0	rw	**CDT_SX_NOM min value** Use this register value as CDT_SX_NOM value when the calculated value for the nominal increment prediction of STATE is less than the register value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.102Register DPLL_CSN_MAX

DPLL Maximum CDT_S Nominal Value Register

DPLL_CSN_MAX
DPLL Maximum CDT_S Nominal Value Register(028F78$_H$) **Application Reset Value: 00FF FFFF$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CSN_MAX**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSN_MAX															
rw															

Field	Bits	Type	Description
CSN_MAX	23:0	rw	**CDT_SX_NOM max value** Use this register value as CDT_SX_NOM value when the calculated value for the nominal increment prediction of STATE is greater than the register value.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.103Register DPLL_STA

DPLL Status of the State Machine States Register

DPLL_STA

DPLL Status of the State Machine States Register(028F40$_H$) **Application Reset Value: 0000 0000$_H$**

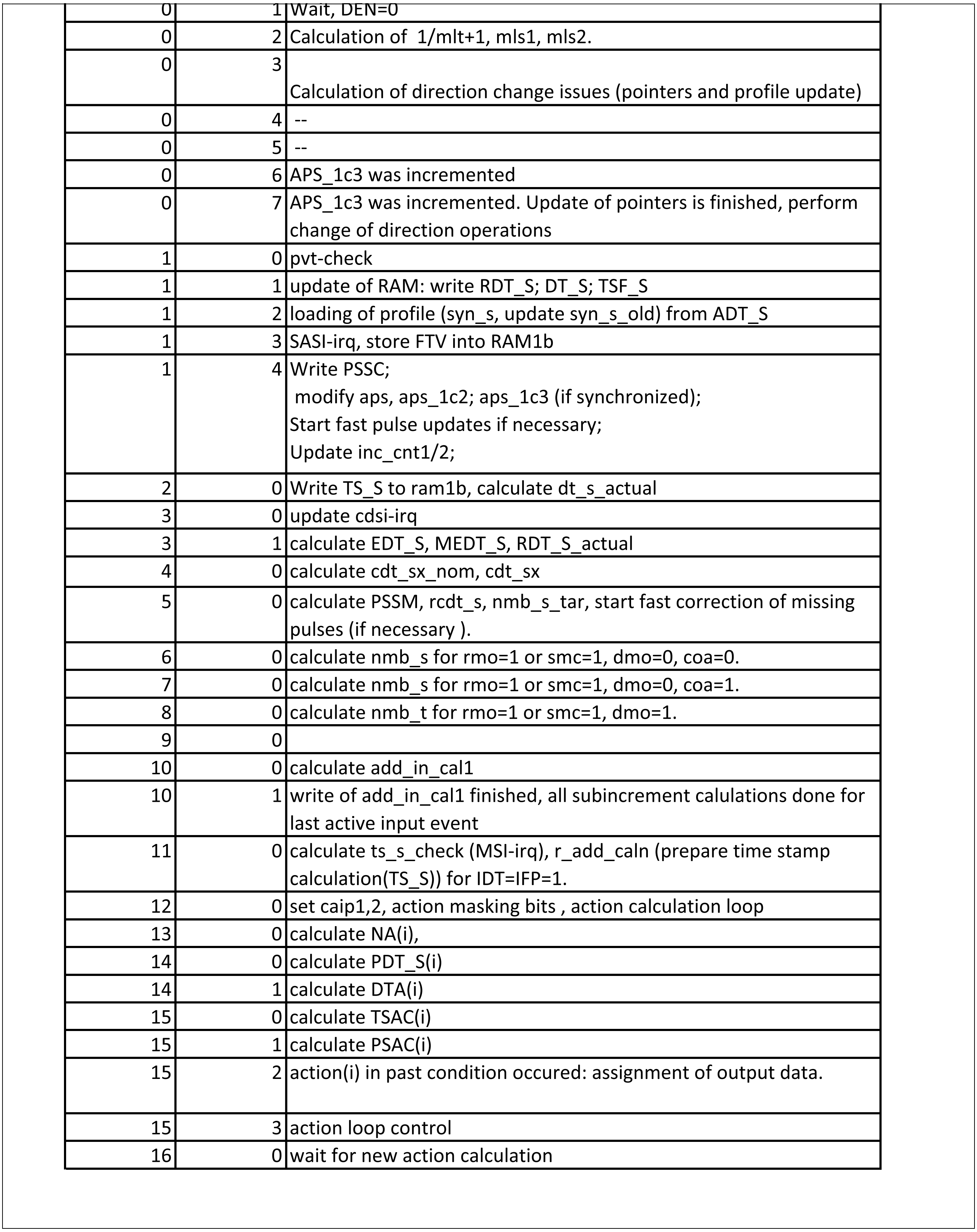
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**CNT_S**			**0**	**STA_S**			
r								r			r	r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STA_S				**CNT_T**			**0**	**STA_T**							
r				r			r	r							

Field	Bits	Type	Description
STA_T	7:0	r	**Status of TRIGGER state machine; state binary coded** This bit field reflects the status of the TRIGGER state machine. The decimal step number 1 to 20 of the state machine is binary coded from 0x01 to 0x14 respectively using the upper 5 bits (8:4). The lower 4 bits (3:0) show substates of the corresponding state machine. When the DPLL is disabled, this field is 0x000. Table STA_T
CNT_T	11:9	r	**Count TRIGGER** This reflects the count of active TRIGGER slopes (mod8). This value shows the number of active TRIGGER slopes (mod8). This value allows distinguishing if the above state machine status is consistent to other status values read before or after it.

Field	Bits	Type	Description
STA_S	19:12	r	**Status of STATE state machine** State binary coded. This bit field reflects the status of the STATE state machine. The decimal step number 21 to 40 of the state machine is binary coded from 0x01 to 0x14 respectively using the upper 5 bits (20:16) after subtraction of 20 to the decimal value. The lower 4 bits (15:12) show substates of the corresponding state machine. When the DPLL is disabled, this field is 0x000. Table_STA_S
CNT_S	23:21	r	**Count STATE** This reflects the count of active STATE slopes (mod8). This value shows the number of active STATE slopes (mod8). This value allows distinguishing if the above state machine status is consistent to other status values read before or after it.
0	8, 20, 31:24	r	**Reserved** Read as zero, shall be written as zero.

Bit description of DPLL_STA

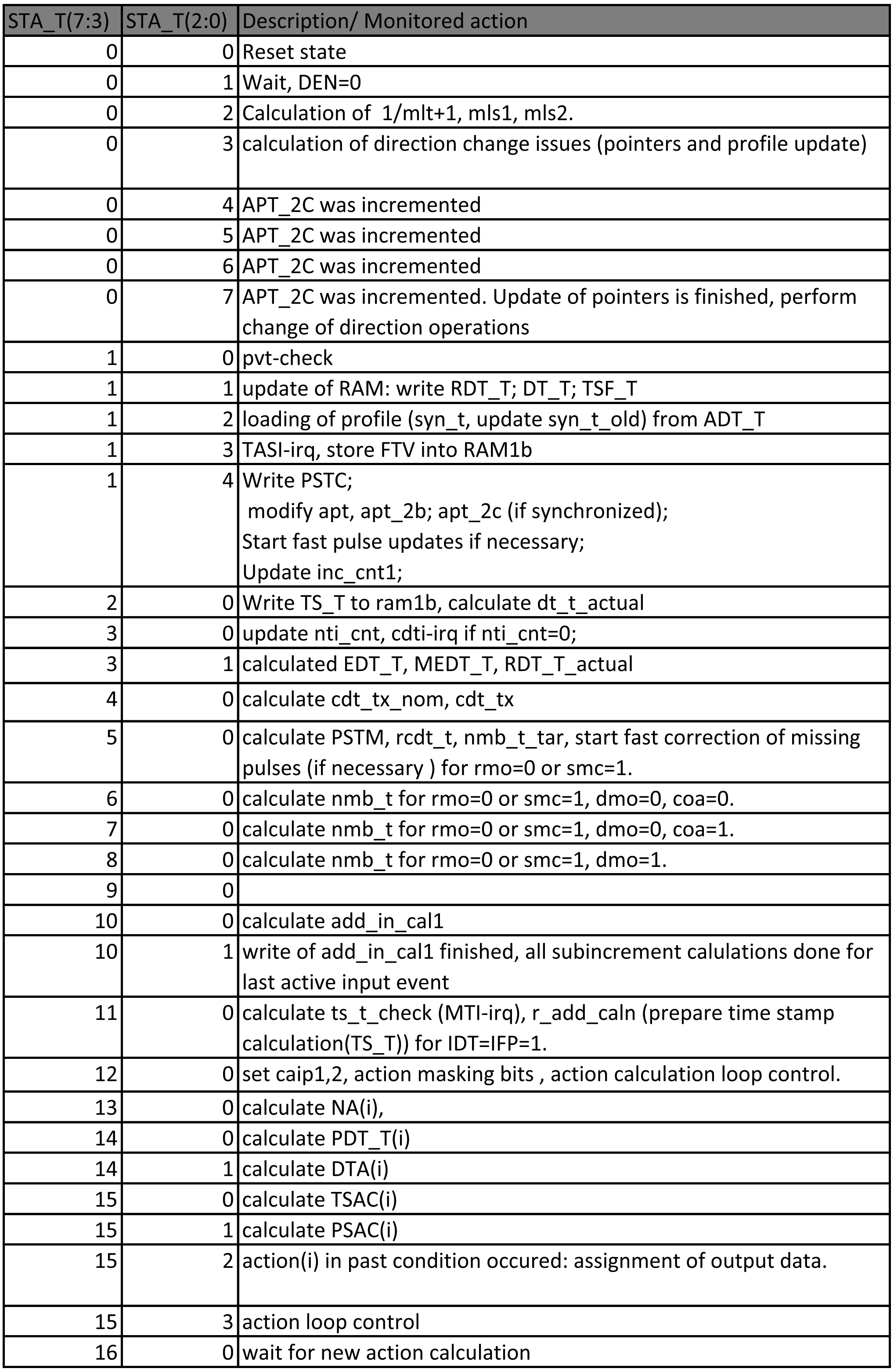


0	1	Wait, DEN=0
0	2	Calculation of 1/mlt+1, mls1, mls2.
0	3	Calculation of direction change issues (pointers and profile update)
0	4	--
0	5	--
0	6	APS_1c3 was incremented
0	7	APS_1c3 was incremented. Update of pointers is finished, perform change of direction operations
1	0	pvt-check
1	1	update of RAM: write RDT_S; DT_S; TSF_S
1	2	loading of profile (syn_s, update syn_s_old) from ADT_S
1	3	SASI-irq, store FTV into RAM1b

1	4	Write PSSC;
		modify aps, aps_1c2; aps_1c3 (if synchronized);
		Start fast pulse updates if necessary;
		Update inc_cnt1/2;

2	0	Write TS_S to ram1b, calculate dt_s_actual
3	0	update cdsi-irq
3	1	calculate EDT_S, MEDT_S, RDT_S_actual
4	0	calculate cdt_sx_nom, cdt_sx
5	0	calculate PSSM, rcdt_s, nmb_s_tar, start fast correction of missing pulses (if necessary).
6	0	calculate nmb_s for rmo=1 or smc=1, dmo=0, coa=0.
7	0	calculate nmb_s for rmo=1 or smc=1, dmo=0, coa=1.
8	0	calculate nmb_t for rmo=1 or smc=1, dmo=1.
9	0	
10	0	calculate add_in_cal1
10	1	write of add_in_cal1 finished, all subincrement calulations done for last active input event
11	0	calculate ts_s_check (MSI-irq), r_add_caln (prepare time stamp calculation(TS_S)) for IDT=IFP=1.
12	0	set caip1,2, action masking bits , action calculation loop
13	0	calculate NA(i),
14	0	calculate PDT_S(i)
14	1	calculate DTA(i)
15	0	calculate TSAC(i)
15	1	calculate PSAC(i)
15	2	action(i) in past condition occured: assignment of output data.
15	3	action loop control
16	0	wait for new action calculation

Figure 130 DPLL_STA.STA_T



STA_T(7:3)	STA_T(2:0)	Description/ Monitored action
0	0	Reset state
0	1	Wait, DEN=0
0	2	Calculation of 1/mlt+1, mls1, mls2.
0	3	calculation of direction change issues (pointers and profile update)
0	4	APT_2C was incremented
0	5	APT_2C was incremented
0	6	APT_2C was incremented
0	7	APT_2C was incremented. Update of pointers is finished, perform change of direction operations
1	0	pvt-check
1	1	update of RAM: write RDT_T; DT_T; TSF_T
1	2	loading of profile (syn_t, update syn_t_old) from ADT_T
1	3	TASI-irq, store FTV into RAM1b

1	4	Write PSTC;
		modify apt, apt_2b; apt_2c (if synchronized);
		Start fast pulse updates if necessary;
		Update inc_cnt1;

2	0	Write TS_T to ram1b, calculate dt_t_actual
3	0	update nti_cnt, cdti-irq if nti_cnt=0;
3	1	calculated EDT_T, MEDT_T, RDT_T_actual
4	0	calculate cdt_tx_nom, cdt_tx
5	0	calculate PSTM, rcdt_t, nmb_t_tar, start fast correction of missing pulses (if necessary) for rmo=0 or smc=1.
6	0	calculate nmb_t for rmo=0 or smc=1, dmo=0, coa=0.
7	0	calculate nmb_t for rmo=0 or smc=1, dmo=0, coa=1.
8	0	calculate nmb_t for rmo=0 or smc=1, dmo=1.
9	0	
10	0	calculate add_in_cal1
10	1	write of add_in_cal1 finished, all subincrement calulations done for last active input event
11	0	calculate ts_t_check (MTI-irq), r_add_caln (prepare time stamp calculation(TS_T)) for IDT=IFP=1.
12	0	set caip1,2, action masking bits , action calculation loop control.
13	0	calculate NA(i),
14	0	calculate PDT_T(i)
14	1	calculate DTA(i)
15	0	calculate TSAC(i)
15	1	calculate PSAC(i)
15	2	action(i) in past condition occured: assignment of output data.
15	3	action loop control
16	0	wait for new action calculation

Figure 131 DPLL_STA.STA_S

28.20.12.104Register DPLL_INCF1_OFFSET

DPLL Start Value of the ADD_IN_ADDER1 Register

DPLL_INCF1_OFFSET
DPLL Start Value of the ADD_IN_ADDER1 Register(028F44$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DPLL_INCF1_OFFSET**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPLL_INCF1_OFFSET															
rw															

Field	Bits	Type	Description
DPLL_INCF1_OFFSET	23:0	rw	**Start value of the ADD_IN_ADDER1** In the case of set DPLL_CTRL_11-INCF1 the ADD_IN_ADDER1 starts always after an active new input event (TRIGGER in normal mode or STATE in emergency mode respectively) with this offset value. In the case of choosing DPLL_INCF1_OFFSET= 0xFFFFFF the generation of the first SUB_INC1 pulse is performed with the next TS_CLK. In the case of DPLL_INCF1_OFFSET= 0x000000 the first pulse is delayed by a full SUB_INC1 period and in the case of DPLL_INCF1_OFFSET= 0x7FFFFF the first pulse is delayed by a half SUB_INC1 period. Any other value is possible.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.105Register DPLL_INCF2_OFFSET

DPLL Start Value of the ADD_IN_ADDER2 Register

DPLL_INCF2_OFFSET
DPLL Start Value of the ADD_IN_ADDER2 Register(028F48$_H$) **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DPLL_INCF2_OFFSET**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPLL_INCF2_OFFSET															
rw															

Field	Bits	Type	Description
DPLL_INCF2_ OFFSET	23:0	rw	**Start value of the ADD_IN_ADDER2** In the case of set DPLL_CTRL_11-INCF2 the ADD_IN_ADDER2 starts always after an active new input event (STATE) with this offset value. In the case of choosing DPLL_INCF2_OFFSET= 0xFFFFFF the generation of the first SUB_INC2 pulse is performed with the next TS_CLK. In the case of DPLL_INCF2_OFFSET= 0x000000 the first pulse is delayed by a full SUB_INC2 period and in the case of DPLL_INCF2_OFFSET= 0x7FFFFF the first pulse is delayed by a half SUB_INC2 period. Any other value is possible.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.106Register DPLL_DT_T_START

DPLL Start Value of DPLL_DT_T_ACT for the First Increment after SIP1 is Set to 1

DPLL_DT_T_START
DPLL Start Value of DPLL_DT_T_ACT for the First Increment after SIP1 is Set to 1(028F4C$_H$) Application Reset Value: 0000 0101$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DPLL_DT_T_START**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPLL_DT_T_START															
rw															

Field	Bits	Type	Description
DPLL_DT_T_S TART	23:0	rw	**Start value of DPLL_DT_T_ACT for the first increment after SIP1 is set to 1** For the first increment after setting SIP1 from 0 to 1, the value of DPLL_DT_T_START is taken instead of the calculated DPLL_DT_T_ACT for the current increment duration. This value should be always > 256 in order to avoid an overflow during the calculation of DPLL_RDT_T_ACT.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.107Register DPLL_DT_S_START

DPLL Start Value of DPLL_DT_S_ACT for the First Increment after SIP2 is Set to 1

DPLL_DT_S_START
DPLL Start Value of DPLL_DT_S_ACT for the First Increment after SIP2 is Set to 1(028F50$_H$) Application Reset Value: 0000 0101$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DPLL_DT_S_START**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPLL_DT_S_START															
rw															

Field	Bits	Type	Description
DPLL_DT_S_START	23:0	rw	**Start value of DPLL_DT_S_ACT for the first increment after SIP2 is set to 1** For the first increment after setting SIP2 from 0 to 1, the value of DPLL_DT_S_START is taken instead of the calculated DPLL_DT_S_ACT for the current increment duration. This value should be always > 256 in order to avoid an overflow during the calculation of DPLL_RDT_S_ACT.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.108Register DPLL_STA_MASK

DPLL Trigger Masks for Signals DPLL_STA_T and DPLL_STA_S

DPLL_STA_MASK
DPLL Trigger Masks for Signals DPLL_STA_T and DPLL_STA_S(028F54$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STA_NOTIFY_S								**STA_NOTIFY_T**							
rw								rw							

Field	Bits	Type	Description
STA_NOTIFY_T	7:0	rw	**Notify value for STA_T of register DPLL_STA** The STA_NOTIFY_T is representing a trigger mask of DPLL_STA.STA_T. When DPLL_STA.STA_T reaches the value of STA_NOTIFY_T the flag DPLL_STA_FLAG.STA_FLAG_T is set to '1' when DPLL_STA.STA_T is leaving the state STA_NOTIFY_T.The signal is visible to MCS0 sub module as part of the special function register.
STA_NOTIFY_S	15:8	rw	**Notify value for STA_S of register DPLL_STA** The STA_NOTIFY_S is representing a trigger mask of DPLL_STA.STA_S. When DPLL_STA.STA_S reaches the value of STA_NOTIFY_S the flag DPLL_STA_FLAG.STA_FLAG_S is set to '1' when DPLL_STA.STA_S is leaving the state STA_NOTIFY_S.
0	31:16	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.109Register DPLL_STA_FLAG

DPLL STA Flag Register

DPLL_STA_FLAG
DPLL STA Flag Register **(028F58$_H$)** **Application Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					**INC_CNT2_FLAG**	**INC_CNT1_FLAG**	**STA_FLAG_S**	**0**							**STA_FLAG_T**
r					rw	rw	rw	r							rw

Field	Bits	Type	Description
STA_FLAG_T	0	rw	**Flag according to DPLL_MASK.STA_NOTIFY_T** The STA_FLAG_T is set to '1' indicating that the signal DPLL_STA.STA_T has left the state defined by the trigger mask of DPLL_STA_MASK.STA_NOTIFY_T. The Flag is reset when this bit of the register is written to '1'. The signal is visible to MCS0 sub module as part of the special function register.

Field	Bits	Type	Description
STA_FLAG_S	8	rw	**Flag according to DPLL_STA_MASK.STA_NOTIFY_S** The STA_FLAG_S is set to '1' indicating that the signal DPLL_STA.STA_S has left the state defined by the trigger mask of DPLL_STA_MASK.STA_NOTIFY_S. The Flag is reset when this bit of the register is written to '1'. The signal is visible to MCS0 sub module as part of the special function register.
INC_CNT1_FLAG	9	rw	**Flag according to DPLL_INC_CNT1_MASK.INC_CNT1_NOTIFY** The INC_CNT1_FLAG is set to '1' indicating that the signal DPLL_INC:CNT1.INC_CNT1 has left the state defined by the trigger mask of DPLL_INC_CNT1_MASK.INC_CNT1_NOTIFY. The Flag is reset when this bit of the register is written to '1'. The signal is visible to MCS0 sub module as part of the special function register.
INC_CNT2_FLAG	10	rw	**Flag according to DPLL_INC_CNT2_MASK.INC_CNT2_NOTIFY** The INC_CNT2_FLAG is set to '1' indicating that the signal DPLL_INC_CNT2.INC_CNT2 has left the state defined by the trigger mask of DPLL_INC_CNT2_MASK.INC_CNT2_NOTIFY. The Flag is reset when this bit of the register is written to '1'. The signal is visible to MCS0 sub module as part of the special function register.
0	7:1, 31:11	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.110Register DPLL_INC_CNT1_MASK

DPLL INC_CNT1 Trigger Mask

DPLL_INC_CNT1_MASK

DPLL INC_CNT1 Trigger Mask **($028F5C_H$)** **Application Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**INC_CNT1_NOTIFY**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC_CNT1_NOTIFY															
rw															

Field	Bits	Type	Description
INC_CNT1_NOTIFY	23:0	rw	**Notify value for INC_CNT1 of register DPLL_INC_CNT1** The INC_CNT1_NOTIFY is representing a trigger mask of DPLL_INC_CNT1.INC_CNT1. When DPLL_INC_CNT1.INC_CNT1 reaches the value of INC_CNT1_NOTIFY the flag DPLL_STA_FLAG.INC_CNT1_FLAG is set to '1' when DPLL_INC_CNT1.INC_CNT1 is leaving the state INC_CNT1_NOTIFY.

Field	Bits	Type	Description
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.111Register DPLL_INC_CNT2_MASK

DPLL INC_CNT2 Trigger Mask

DPLL_INC_CNT2_MASK
DPLL INC_CNT2 Trigger Mask (028F60$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**INC_CNT2_NOTIFY**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC_CNT2_NOTIFY															
rw															

Field	Bits	Type	Description
INC_CNT2_NOTIFY	23:0	rw	**Notify value for INC_CNT2 of register DPLL_INC_CNT2** The INC_CNT2_NOTIFY is representing a trigger mask of DPLL_INC_CNT2.INC_CNT2. When DPLL_INC_CNT2.INC_CNT2 reaches the value of INC_CNT2_NOTIFY the flag DPLL_STA_FLAG.INC_CNT2_FLAG is set to '1' when DPLL_INC_CNT2.INC_CNT2 is leaving the state INC_CNT2_NOTIFY.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.112Register DPLL_NUSC_EXT1

DPLL Extension Register Number 1 for DPLL_NUSC 4

DPLL_NUSC_EXT1
DPLL Extension Register Number 1 for DPLL_NUSC 4(028F64$_H$) Application Reset Value: 0001 0001$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	**WSYN**	**0**							**SYN_S_OLD**						
r	rw	r							rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									**SYN_S**						
r									rw						

Field	Bits	Type	Description
SYN_S	6:0	rw	**Number of real and virtual events to be considered for the current increment** This value reflects the NS value of the last valid increment, stored in ADT_S[i]; to be updated after all calculations in step 37 of Table **Section 28.20.8.6.6**. This value can only be written when the WSYN bit in this register is set.
SYN_S_OLD	22:16	rw	**Number of real and virtual events to be considered for the last increment** This value reflects the NS value of the last but one valid increment, stored in ADT_S[i]; is updated automatically when writing SYN_S. This value is updated by the SYN_S value when the WSYN bit in this register is set.
WSYN	30	rw	**Write control bit for SYN_S and SYN_S_OLD** Read as zero. 0_B The SYN_S value is not writeable 1_B The SYN_S value is writeable
0	15:7, 29:23, 31	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.113Register DPLL_NUSC_EXT2

DPLL Extension Register Number 2 for DPLL_NUSC 4

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is set. If DPLL_CTRL_11.STATE_EXT is not set, any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_NUSC_EXT2

DPLL Extension Register Number 2 for DPLL_NUSC 4(028F68$_H$) **Application Reset Value: 0000 0001$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WVSN	**0**	**WNUS**	**0**						**VSN**						
rw	r	rw	r						rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSS	**0**								**NUSE**						
rw	r								rw						

Field	Bits	Type	Description
NUSE	6:0	rw	**Number of recent STATE events used for SUB_INCx calculations modulo 2*(SNUmax+1)** No gap is considered in that case for this value, but in the VSN value (see below): This register is set by the CPU but reset automatically to “1” by a change of direction or loss of LOCK. Each other value can be set by the CPU, maybe Full_SCALE, HALF_SCALE or parts of them. The relation values QDT_Sx are calculated using NUSE values in the past with its maximum value of 2*SNU+1. This value can only be written when the WNUS bit is set.
FSS	15	rw	**This value is to be set, when NUSE is set to FULL_SCALE** This value is set by the CPU, but reset automatically to '0' by a change of direction or loss of LOCK. This value can only be written when the WNUS bit is set. 0_B The NUSE value is less then FULL_SCALE 1_B The NUSE value is equal to FULL_SCALE
VSN	22:16	rw	**Number of virtual state increments in the current NUSE region** This value reflects the number of virtual increments in the current NUSE region; for NUSE=1 this value is zero, when the CPU sets NUSE to a value > 1 or zero(2^7 modulo 2^7) , it must also set VSN to the correspondent value;the VSN value is subtracted from the NUSE value in order to get the corresponding APS value for the past; the VSN value is not used for the APS_1C2 pointer. VSN is to be updated by the CPU when a new gap is to be considered for NUSE or a gap is leaving the NUSE region; for this purpose the SASI interrupt can be used; no further update of VSN is necessary when NUSE is set to FULL_SCALE. This value can only be written when the WVSN bit is set.
WNUS	29	rw	**Write control bit for NUSE** Read as zero. 0_B The NUSE value is not writeable 1_B The NUSE value is writeable
WVSN	31	rw	**Write control bit for VSN** Read as zero. 0_B The VSN value is not writeable 1_B The VSN value is writeable
0	14:7, 28:23, 30	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.114Register DPLL_APS_EXT

DPLL Extension Register for DPLL_APS

Note: *This register is only used when DPLL_CTRL_11.STATE_EXT is set. If DPLL_CTRL_11.STATE_EXT is not set, any read/write access to this register will return AEI_STATUS = 0b10.*

DPLL_APS_EXT

DPLL Extension Register for DPLL_APS (028F38$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											**APS_1C2**				
r											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APS_1C2		**WAPS_1C2**	**0**				**APS**							**WAPS**	**0**
rw		rw	r				rw							rw	r

Field	Bits	Type	Description
WAPS	1	rw	**Write bit for address pointer APS** Read as zero. 0_B The APS is not writeable 1_B The APS is writeable
APS	8:2	rw	**Actual RAM pointer address value for DT_S[i] and RDT_S[i]** Actual RAM pointer and synchronization position/value of *STATE* events in FULL_SCALE for up to 128 *STATE* events but limited to 2*(SNU+1-SYN_NS) in normal and emergency mode for SYSF=0 or to 2*(SNU+1)-SYN_NS for SYSF=1 respectively; See **Section 28.20.10**. APS is incremented (decremented) by one for each active *STATE* event and DIR2=0 DIR2=1). The APS offset value is added in the above shown bit position with the subsection offset of the RAM region. The APS pointer value is directed to the RAM position, in which the data values are to be written, which correspond to the last increment. The APS value is not to be changed, when the direction (shown by DIR2) changes, because it points always to a storage place after the considered increment. Changing of DIR2 takes place always after an active *STATE* event and the resulting increment/decrement. This value can only be written when the WAPS bit is set.
WAPS_1C2	13	rw	**Write bit for address pointer APS_1C2** Read as zero. 0_B The APS_1C2 is not writeable 1_B The APS_1C2 is writeable

Field	Bits	Type	Description
APS_1C2	20:14	rw	**Actual RAM pointer address value for TSF_S[i]** Initial value: zero (0x00). Actual RAM pointer and synchronization position/value of *STATE* events in FULL_SCALE for up to 64 *STATE* events but limited to 2*(SNU+1) in normal and emergency mode; this pointer is used for the RAM region 1c2. For SYS=1: APS_1C2 is incremented (decremented) by SYN_S_OLD for each active *STATE* event and DIR2=0 (DIR2=1). For SYS=0: APT_1c2 is incremented or decremented by 1 respectively. The APS_1C2 offset value is added in the above shown bit position with the subsection offset of the RAM region. In addition when the APS_1C3 value is written by the CPU - in order to synchronize the DPLL- with the next active *STATE* event the APS_1C2_EXT value is added/subtracted (while APS_1C2_STATUS is one; see DPLL_APT_SYNC register at **Section 28.20.12.25**). This value can only be written when the WAPS_1C2 bit is set
0	0, 12:9, 31:21	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.115Register DPLL_APS_1C3_EXT

DPLL Extension Register for DPLL_APS_1C3

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is set. If DPLL_CTRL_11.STATE_EXT is not set, any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_APS_1C3_EXT
DPLL Extension Register for DPLL_APS_1C3 (028F3C$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							**APS_1C3**							**0**	
r							rw							r	

Field	Bits	Type	Description
APS_1C3	8:2	rw	**Actual RAM pointer address value for ADT_S[i]** Initial value: zero (0x00). Actual RAM pointer and synchronization position/value of *STATE* events in FULL_SCALE for up to 128 *STATE* events but limited to 2*(SNU+1-SYN_NS) in normal and emergency mode for SYSF=0 or to 2*(SNU+1)-SYN_NS for SYSF=1 respectively; this pointer is used for the RAM region 1c3. See **Section 28.20.10**. The RAM pointer is set by the CPU accordingly, when the synchronization condition was detected. The APS_1C3 pointer value is directed to the RAM position of the profile element in RAM region 1c2, which corresponds to the current increment. When changing the direction DIR1 or DIR2 respectively, this is always known before an active *STATE* event is processed. This is because of the pattern recognition in SPE (for PMSM) or because of the direction change recognition by TRIGGER. This direction change results in an automatic increment (forwards) or decrement (backwards) when the input event occurs in addition with a 2 times correction. The APS_1C3_x offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region.
0	1:0, 31:9	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.116Register DPLL_APS_SYNC_EXT

DPLL Extension Register for DPLL_APS_SYNC

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is set. If DPLL_CTRL_11.STATE_EXT is not set, any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_APS_SYNC_EXT

DPLL Extension Register for DPLL_APS_SYNC (028F30$_H$) Application Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									**APS_1C2_OLD**						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APS_1C2_STATUS	**0**								**APS_1C2_EXT**						
rw	r								rw						

Field	Bits	Type	Description
APS_1C2_EXT	6:0	rw	**Address pointer 1c2 extension** This offset value determines, by which value the APS_1C2 is changed at the synchronization time; set by CPU before the synchronization is performed. This offset value is the number of virtual increments to be inserted in the TSF for an imminent intended synchronization; the CPU sets its value dependent on the gaps until the synchronization time taking into account the considered NUSE value to be set and including the next future increment (when SYN_S_OLD is still 1). When the synchronization takes place, this value is to be added to the APS_1C2 address pointer (for forward direction, DIR2=0) and the APT_1c2_status bit is cleared after it. For backward direction subtract APS_1C2_EXT accordingly. When the synchronization is intended and the NUSE value is to be set to FULL_SCALE after it, the APS_1C2_EXT value must be set to SYN_NS (for SYSF=1) or 2*SYN_NS (for SYSF=0) in order to be able to fill all gaps in the extended TSF_S with the corresponding values by the CPU. When still not all values for FULL_SCALE are available, the APS_1C2_EXT value considers only a share according to the NUSE value to be set after the synchronization.
APS_1C2_STATUS	15	rw	**Address pointer 1c2 status** Set by CPU before the synchronization is performed. The value is cleared automatically when the APS_1C2_OLD value is written. 0_B APS_1C2_EXT is not to be considered 1_B APS_1C2_EXT has to be considered for time stamp field extension
APS_1C2_OLD	22:16	rw	**Address pointer STATE for RAM region 1c2 at synchronization time** This value is set by the current APS_1C2 value when the synchronization takes place for the first active STATE event after writing APS_1C3 but before adding the offset value APS_1C2_EXT (that means: when APS_1C2_STATUS=1). Address pointer APS_1C2 value at the moment of synchronization, before the offset value is added, that means the pointer with this value points to the last value before the additional inserted gap.
0	14:7, 31:23	r	**Reserved** Read as zero, shall be written as zero.

28.20.12.117Register DPLL_CTRL_EXT

DPLL Extension Register for DPLL_CTRL

Note: This register is only used when DPLL_CTRL_11.STATE_EXT is set. If DPLL_CTRL_11.STATE_EXT is not set, any read/write access to this register will return AEI_STATUS = 0b10.

DPLL_CTRL_EXT

DPLL Extension Register for DPLL_CTRL (028F34$_H$) Application Reset Value: 0000 0017$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										**SYN_NS**					
r										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										**SNU**					
r										rw					

Field	Bits	Type	Description
SNU	5:0	rw	**STATE number** This bit can only be written when the DPLL is disabled. The number of nominal *STATE* events is the decimal value plus 1. This value can only be written when (RMO=0 and SMC=0) or DEN=0. Set SSL=00 before changing this value and set RMO=1 only after FULL_SCALE with SSL>0.
SYN_NS	21:16	rw	**Synchronization number of STATE** Summarized number of virtual increments in HALF_SCALE. Sum of all systematic missing *STATE* events in HALF_SCALE (for SYSF=0) or FULL SCALE (for SYSF=1) ; the SYN_NS missing *STATES* can be divided up to an arbitrary number of blocks. The pattern of events and missing events in FULL_SCALE is shown in RAM region 1c3 as value NS in addition to the adapted values. The number of stored increments in FULL_SCALE must be equal to 2*(SNU+1-SYN_NS) for SYSF=0 or 2*(SNU+1)-SYN_NS for SYSF=1 . This pattern is written by the CPU beginning from a fixed reference point (maybe beginning of the FULL_SCALE region). The relation to the actual increment is established by setting of the profile RAM pointer APS_1C3 in an appropriate relation to the RAM pointer APS of the actual increment by the CPU. This value can only be written when the DPLL is disabled. This value can only be written when (RMO=0 and SMC=0) or DEN=0. Set SSL=00 before changing this value and set RMO=1 only after FULL_SCALE with SSL>0.
0	15:6, 31:22	r	**Reserved** Read as zero, shall be written as zero.

28.20.13 DPLL RAM Region 1a value description

28.20.13.1Memory DPLL_PSA[i]

DPLL ACTION_i Position/Value Request

DPLL_PSAi (i=0-31)
DPLL ACTION_i Position/Value Request (028200$_H$+i*4) Reset Value: Table 137

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**PSA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSA															
rw															

Field	Bits	Type	Description
PSA	23:0	rw	**Position information of a desired action i** This value can only be written when the DPLL is disabled. The PSA values for actions 24...31 are not available for all devices but depends on specific product configuration.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 137 Reset Values of DPLL_PSAi (i=0-31)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1A	0000 0000$_H$	Cleared by state machine

28.20.13.2Memory DPLL_DLA[i]

DPLL ACTION_i Time to React before PSAi

DPLL_DLAi (i=0-31)
DPLL ACTION_i Time to React before PSAi (028280$_H$+i*4) Reset Value: Table 138

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DLA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLA															
rw															

Field	Bits	Type	Description
DLA	23:0	rw	**Time to react before the corresponding position value of a desired action i is reached** In the case of LOW_RES=1 (see **Table 70**), this delay value must be also given as low resolution value. This value can only be written when the DPLL is disabled.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 138 Reset Values of DPLL_DLAi (i=0-31)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT_1A	0000 0000$_H$	Cleared by state machine

28.20.13.3Memory DPLL_NA[i]

DPLL Calculated Number of TRIGGER/STATE Increments to ACTION_i

DPLL_NAi (i=0-31)
DPLL Calculated Number of TRIGGER/STATE Increments to ACTION_i(028300$_H$+i*4) Reset Value: Table 139

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="8">0</td><td colspan="4">0</td><td colspan="4">DW</td></tr>
<tr><td colspan="8">r</td><td colspan="4">rw</td><td colspan="4">rw</td></tr>
</table>

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="6">DW</td><td colspan="10">DB</td></tr>
<tr><td colspan="6">rw</td><td colspan="10">rw</td></tr>
</table>

Field	Bits	Type	Description
DB	9:0	rw	**Number of events to Action_i (fractional part)** The NA values for actions 24...31 are only available for device 4 or 5. This value can only be written when the DPLL is disabled.
DW	19:10	rw	**Number of events to Action_i (integer part)** Use the maximum value for NA_DW=0x3FF in the case of a calculated value which exceeds the represent able value. This value can only be written when the DPLL is disabled.
0	23:20	rw	**Reserved** Read as zero, shall be written as zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 139 Reset Values of DPLL_NAi (i=0-31)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1A	0000 0000$_H$	Cleared by state machine

28.20.13.4Memory DPLL_DTA[i]

DPLL Calculated Relative TIME to ACTION_i

DPLL_DTAi (i=0-31)

DPLL Calculated Relative TIME to ACTION_i (028380$_H$+i*4) **Reset Value: Table 140**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DTA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTA															
rw															

Field	Bits	Type	Description
DTA	23:0	rw	**Calculated relative time to ACTION_i** This value can only be written when the DPLL is disabled. The DTA value is a positive integer value. When calculations using equations DPLL-12 or DPLL-14 result in a negative value, it is replaced by zero.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

Table 140 Reset Values of DPLL_DTAi (i=0-31)

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _1A	0000 0000$_H$	Cleared by state machine

28.20.14 DPLL RAM Region 2 value description

Bits 31 to 24 of RAM region 2 are not implemented and therefore always read as zero (reserved). Other bits which are declared as reserved are not protected against writing. Unused address regions are not protected against writing when implemented.

28.20.14.1Memory DPLL_RDT_T[i]

Region 2a. Reciprocal value of the corresponding successive increment i, for each true nominal increment.

Reciprocal Values of the Nominal TRIGGER Increments Duration in FULL_SCALE

Note: The starting index for Memory DPLL_RDT_T[i] in RAM2 is defined by the parameter AOSV_2A in DPLL_AOSV2 Register

DPLL_RDT_Ti
Region 2a. Reciprocal value of the corresponding successive increment i, for each true nominal increment.
($02C000_H$) **Reset Value: Table 141**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**RDT_T**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT_T															
rw															

Field	Bits	Type	Description
RDT_T	23:0	rw	**Reciprocal difference time of TRIGGER; 2* (TNU+1- SYN_NT) stored values nominal reciprocal value of the number of time stamp clocks measured in the corresponding increment (which is divided by the number of nominal increments); multiplied by *232 while only the lower 24 bits are used; the LSB is rounded up, when the next truncated bit is 1.** RDT_T: Reciprocal difference time of TRIGGER; 2* (TNU+1- SYN_NT) stored values nominal reciprocal value of the number of time stamp clocks measured in the corresponding increment (which is divided by the number of nominal increments); multiplied by *232 while only the lower 24 bits are used; the LSB is rounded up, when the next truncated bit is 1. **Note:** There are 2* (TNU+1- SYN_NT) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
0	31:24	r	

Table 141 Reset Values of DPLL_RDT_Ti

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _2	--00 0000$_H$	Cleared by state machine

28.20.14.2Memory DPLL_TSF_T[i]

Region 2b. Time Stamp Field for TRIGGER event i, for each true nominal increment plus each virtual increment.

Time Stamp Values of the Nominal TRIGGER Increments in FULL_SCALE

Note: The starting index for Memory DPLL_TSF_T[i] in RAM2 is defined by the parameter AOSV_2C in DPLL_AOSV2 Register

DPLL_TSF_Ti
Region 2b. Time Stamp Field for TRIGGER event i, for each true nominal increment plus each virtual increment. **(02C000$_H$)** **Reset Value: Table 142**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**TSF_T**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSF_T															
rw															

Field	Bits	Type	Description
TSF_T	23:0	rw	**Time stamp field of active TRIGGER slopes** TSF_T: Time stamp field of active TRIGGER slopes **Note:** There are 2* (TNU+1) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
0	31:24	r	

Table 142 Reset Values of DPLL_TSF_Ti

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX$_H$	Undefined after Power On
DPLL_RAM_INI.INIT _2	--00 0000$_H$	Cleared by state machine

28.20.14.3Memory DPLL_ADT_T[i]

Region 2c. Adapt values for the current TRIGGER increment i, for each true nominal increment.

Adapt and Profile Values of the TRIGGER Increments in FULL_SCALE

Note: The starting index for Memory DPLL_ADT_T[i] in RAM2 is defined by the parameter AOSV_2C in DPLL_AOSV2 Register

DPLL_ADT_Ti
Region 2c. Adapt values for the current TRIGGER increment i, for each true nominal increment.(02C000$_H$)
Reset Value: Table 143

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**0**					**NT**		
r								rw					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TINT			**PD**												
rw			rh												

Field	Bits	Type	Description
PD	12:0	rh	**Physical deviation; Adapt values for each nominal TRIGGER increment in FULL_SCALE (sint13);** PD: Physical deviation; Adapt values for each nominal TRIGGER increment in FULL_SCALE (sint13); The PD value does mean the number of SUB_INC1 pulses to be added to NT*((MLT+1) + PD); the absolute value of a negative PD must not exceed (MLT+1) or MLS1 respectively;systematic missing *TRIGGER* events must be considered for the value of PD;
TINT	15:13	rw	**TRIGGER Interrupt information** Depending on the value, up to 7 different interrupts can be generated. In the current version, the 5 interrupts TE0_IRQ ... TE4_IRQ are supported by TINT=001$_B$, 010$_B$, 011$_B$, 100$_B$ and 101$_B$, respectively. For the values 000$_B$, 110$_B$ and 111$_B$, no interrupt is generated and no other reaction is performed. The corresponding interrupt is activated when the TINT value is read by the DPLL together with the other values (PD, NT) according to the profile.
NT	18:16	rw	**Number of TRIGGERs; number of nominal TRIGGER parts in the corresponding increment.** NT: Number of TRIGGERs; number of nominal TRIGGER parts in the corresponding increment. **Note:** There are 2* (TNU+1- SYN_NT) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
0	23:19	rw	**Not used - NOT_USED** Not used Note: must be written to zero.
0	31:24	r	

Table 143 Reset Values of DPLL_ADT_Ti

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX_H	Undefined after Power On
DPLL_RAM_INI.INIT_2	--00 0000_H	Cleared by state machine

28.20.14.4Memory DPLL_DT_T[i]

Region 2d. Uncorrected last increment value of TRIGGER i, for each true nominal increment.

Nominal TRIGGER Increments Duration in FULL_SCALE

Note: The starting index for Memory DPLL_DT_T[i] in RAM2 is defined by the parameter AOSV_2D in DPLL_AOSV2 Register

DPLL_DT_Ti
Region 2d. Uncorrected last increment value of TRIGGER i, for each true nominal increment.($02C000_H$)
Reset Value: Table 144

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DT_T**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DT_T															
rw															

Field	Bits	Type	Description
DT_T	23:0	rw	**Difference time of TRIGGER; increment duration values for each TRIGGER increment in FULL_SCALE divided by the number of nominal increments (nominal value).** DT_T: Difference time of TRIGGER; increment duration values for each TRIGGER increment in FULL_SCALE divided by the number of nominal increments (nominal value). **Note:** There are 2* (TNU+1- SYN_NT) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
0	31:24	r	

Table 144 Reset Values of DPLL_DT_Ti

Reset Type	Reset Value	Note
PowerOn Reset	XXXX XXXX_H	Undefined after Power On
DPLL_RAM_INI.INIT_2	--00 0000_H	Cleared by state machine

28.20.15 MCS to DPLL Register description

28.20.15.1Register MCS2DPLL_DEB0

MCS to DPLL Data Exchange Buffer 0

READ access from MCS: Duration of the last increment DT_S_ACT (**Update of RAM in Normal and Emergency Mode**). This value is updated by the DPLL during the update of ram (STA_S = 0b0000_1001) and is ready to be read when STA_S is modified to 0b0000_1010.WRITE access from MCS: The DPLL expects DT_S[p-1] (**Equation DPLL-8 to calculate the error of last prediction**) or DT_S[p+1](**Equation DPLL-8a to calculate the error of the last prediction**) during the increment prediction (STA_S = 0b0001_0000).

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB0
MCS to DPLL Data Exchange Buffer 0 (007800$_H$) **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 0. Actual content depends on whether it is a Read or Write operation from MCS.** DATA: Data exchange buffer 0. Actual content depends on whether it is a Read or Write operation from MCS.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.2Register MCS2DPLL_DEB1

MCS to DPLL Data Exchange Buffer 1

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects RDT_S[p-1] (**Equation DPLL-7a1 to calculate QDT_S_ACT**) or RDT_S[p+1](**Equation DPLL-7a2 to calculate QDT_S_ACT backwards**) during the increment prediction (STA_S = 0b0001_0000) and RDT_S[t-1] (**Action calculations for STATE forwards**) or RDT_S[t+1](**Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: In both cases, the data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

MCS2DPLL_DEB1

MCS to DPLL Data Exchange Buffer 1 **(007804$_H$)** **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 1.** DATA: Data exchange buffer 1.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.3Register MCS2DPLL_DEB2

MCS to DPLL Data Exchange Buffer 2

READ access from MCS: TS_Sx (**Equation DPLL-6a4 to update the time stamp values for STATE**). Use to compute/update the time stamp values for STATE during the update of ram (STA_S = 0b0000_1001)WRITE access from MCS: The DPLL expects RDT_S[p-q-1] (**Equation DPLL-8 to calculate the error of last prediction**) or RDT_S[p+q+1](**Equation DPLL-8a to calculate the error of the last prediction**) during the increment prediction (STA_S = 0b0001_0000).

Note: The data read from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB2

MCS to DPLL Data Exchange Buffer 2 **(007808$_H$)** **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 2. Actual content depends on whether it is a Read or Write operation from MCS.** DATA: Data exchange buffer 2. Actual content depends on whether it is a Read or Write operation from MCS.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.4Register MCS2DPLL_DEB3

MCS to DPLL Data Exchange Buffer 3

READ access from MCS: DT_Sx (**Equation DPLL-6a4 to update the time stamp values for STATE**). Use to compute/update the time stamp values for STATE during the update of ram (STA_S = 0b0000_1001)WRITE access from MCS: The DPLL expects DT_S[p-q] (**Equation DPLL-8 to calculate the error of last prediction**) or DT_S[p+q](**Equation DPLL-8a to calculate the error of the last prediction**) during the increment prediction (STA_S = 0b0001_0000).

Note: The data read from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

MCS2DPLL_DEB3

MCS to DPLL Data Exchange Buffer 3 **(00780C$_H$)** **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 3. Actual content depends on whether it is a Read or Write operation from MCS.** DATA: Data exchange buffer 3. Actual content depends on whether it is a Read or Write operation from MCS.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.5Register MCS2DPLL_DEB4

MCS to DPLL Data Exchange Buffer 4

READ access from MCS: SYN_S_OLD (**Equation DPLL-6a4 to update the time stamp values for STATE**). Use to compute/update the time stamp values for STATE during the update of ram (STA_S = 0b0000_1001)WRITE access from MCS: The DPLL expects RDT_S[p-q] (**Equations DPLL-10 to calculate the current increment (nominal value)**) or RDT_S[p+q](**Equations DPLL-10 to calculate the current increment value**) during the increment prediction (STA_S = 0b0001_0000) and RDT_S[t-q] (**Action calculations for STATE forwards**) or RDT_S[t+q](**Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: The data read from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB4

MCS to DPLL Data Exchange Buffer 4 (007810$_H$) PowerOn Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 4. Actual content depends on whether it is a Read or Write operation from MCS.** DATA: Data exchange buffer 4. Actual content depends on whether it is a Read or Write operation from MCS.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.6Register MCS2DPLL_DEB5

MCS to DPLL Data Exchange Buffer 5

READ access from MCS: M_DW (m in **Action calculations for STATE forwards** and **Action calculations for STATE backwards**). Use to provide the proper Time Stamp Field value during the action calculation (STA_S = 0b0111_0000)WRITE access from MCS: The DPLL expects DT_S[p-q+1] (**Equations DPLL-10 to calculate the current increment (nominal value)**) or DT_S[p+q-1](**Equations DPLL-10 to calculate the current increment value**) during the increment prediction (STA_S = 0b0001_0000) and DT_S[t-q+1] (**Action calculations for STATE forwards**) or DT_S[t+q-1](**Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

MCS2DPLL_DEB5
MCS to DPLL Data Exchange Buffer 5 (007814$_H$) PowerOn Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 5. Actual content depends on whether it is a Read or Write operation from MCS.** DATA: Data exchange buffer 5. Actual content depends on whether it is a Read or Write operation from MCS.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.7Register MCS2DPLL_DEB6

MCS to DPLL Data Exchange Buffer 6

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects ADT_S[APS_1C2] during the update of RAM (STA_S = 0b0000_1001) and change of direction (STA_S = 0b0000_0100 and STA_S = 0b0000_0110)

Note: In both cases, the current ADT_S[APS_1C2] value should be stored in the register before unlocking the state machine the second time, i.e.: between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

Note: The format of ADT_S should match the defined in **ADDR**

MCS2DPLL_DEB6
MCS to DPLL Data Exchange Buffer 6 (007818$_H$) PowerOn Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 6.** DATA: Data exchange buffer 6.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.8Register MCS2DPLL_DEB7

MCS to DPLL Data Exchange Buffer 7

READ access from MCS: Duration of the reciprocal of the last increment RDT_S_ACT (**Update of RAM in Normal and Emergency Mode**). This value is written by the DPLL during the update of ram (STA_S = 0b0000_1001) and is ready to be read when STA_S is modified to 0b0000_1010.WRITE access from MCS: The DPLL expects the reciprocal of the last increment RDT_S[APS] (**Update of RAM in Normal and Emergency Mode**) before it is overwritten with RDT_S_ACT during the update of ram (STA_S = 0b0000_1001). For the action calculation, this value is needed as well as RDT_S[t] in **Action calculations for STATE forwards** and **Action calculations for STATE backwards** .

Note: During an update of ram, perform the write before unlocking the state machine (STA_S = 0b0000_1001), i.e.: after the first write to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO) but before the second write. During the action calculation, the data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

MCS2DPLL_DEB7

MCS to DPLL Data Exchange Buffer 7 (00781C$_H$) PowerOn Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 7. Actual content depends on whether it is a Read or Write operation from MCS.** DATA: Data exchange buffer 7. Actual content depends on whether it is a Read or Write operation from MCS.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.9Register MCS2DPLL_DEB8

MCS to DPLL Data Exchange Buffer 8

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects TSF_S[p] (**Action calculations for STATE forwards** and **Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB8

MCS to DPLL Data Exchange Buffer 8 (007820$_H$) PowerOn Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 8.** DATA: Data exchange buffer 8.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.10Register MCS2DPLL_DEB9

MCS to DPLL Data Exchange Buffer 9

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects TSF_S[p-n] (**Action calculations for STATE forwards**) or TSF_S[p+n](**Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB9

MCS to DPLL Data Exchange Buffer 9 (007824$_H$) PowerOn Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 9.** DATA: Data exchange buffer 9.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.11Register MCS2DPLL_DEB10

MCS to DPLL Data Exchange Buffer 10

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects TSF_S[p+m-n] (**Action calculations for STATE forwards**) or TSF_S[p-m+n](**Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB10
MCS to DPLL Data Exchange Buffer 10 (007828$_H$) **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 10.** DATA: Data exchange buffer 10.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.12Register MCS2DPLL_DEB11

MCS to DPLL Data Exchange Buffer 11

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects TSF_S[p+m] (**Action calculations for STATE forwards**) or TSF_S[p-m](**Action calculations for STATE backwards**) during the action calculation (STA_S = 0b0111_0000)

Note: The data write from MCS should be performed between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO)

MCS2DPLL_DEB11
MCS to DPLL Data Exchange Buffer 11 (00782C$_H$) **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 11.** DATA: Data exchange buffer 11.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.13Register MCS2DPLL_DEB12

MCS to DPLL Data Exchange Buffer 12

READ access from MCS: Reads as 0.WRITE access from MCS: The DPLL expects the future adapt information ADT_S[APS_1C2+1] (when in forwards) or ADT_S[APS_1C2-1] (when in backwards) (**Equation DPLL-8 to calculate the error of last prediction**) during the update of RAM (STA_S = 0b0000_1001) and change of direction (STA_S = 0b0000_0100 and STA_S = 0b0000_0110)

Note: In both cases, the current ADT_S[APS_1C2+1] or ADT_S[APS_1C2-1] value should be stored in the register before unlocking the state machine the second time, i.e.: between the two writes to MCS2DPLL_DEB15 (MCS2DPLL_STATUS_INFO).

Note: The format of ADT_S should match the defined in **ADDR**

MCS2DPLL_DEB12
MCS to DPLL Data Exchange Buffer 12 **(007830$_H$)** **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 12.** DATA: Data exchange buffer 12.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.14Register MCS2DPLL_DEB13

MCS to DPLL Data Exchange Buffer 13

READ access from MCS: Reads as 0.WRITE access from MCS: Ignored during DPLL processing.

MCS2DPLL_DEB13
MCS to DPLL Data Exchange Buffer 13 (007834$_H$) **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rh															

Field	Bits	Type	Description
DATA	23:0	rh	**Data exchange buffer 13.** DATA: Data exchange buffer 13.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.15Register MCS2DPLL_DEB14

MCS to DPLL Data Exchange Buffer 14

READ access from MCS: Reads as 0.WRITE access from MCS: Ignored during DPLL processing.

MCS2DPLL_DEB14
MCS to DPLL Data Exchange Buffer 14 (007838$_H$) **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rh							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rh															

Field	Bits	Type	Description
DATA	23:0	rh	**Data exchange buffer 14.** DATA: Data exchange buffer 14.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.

28.20.15.16Register MCS2DPLL_DEB15

MCS to DPLL Data Exchange Buffer 15

READ access from MCS: Reads as 0.WRITE access from MCS: Unlocks the DPLL STATE state machine. See **General functionality** .

MCS2DPLL_DEB15
MCS to DPLL Data Exchange Buffer 15 **(00783C$_H$)** **PowerOn Reset Value: 0000 0000$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								**DATA**							
r								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															
rw															

Field	Bits	Type	Description
DATA	23:0	rw	**Data exchange buffer 15.** DATA: Data exchange buffer 15.
0	31:24	r	**Reserved** Read as zero, shall be written as zero.