Cascade SiT9514x Alarm Registers

Application Note

Contents

Figures

No figures included in this document

Tables

1 Introduction

This document describes the alarm registers for the various clock monitoring functions available for the SiT9514x jitter cleaners, network synchronizers, and clock generators. Additionally, background is provided for the various clock monitor functions and a clear description with examples illustrated for dynamic live status bits and sticky bits.

In addition, SiT9514x family parts have some programmable Flexible IO pins (FLEXIOs) that can be used in a flexible manner to get various status updates on the chip pins. These can be used for dynamic real time clock status monitoring using user-defined options of what is seen on these pins (refer to AN20008).

This application note describes the read back of the alarms for the various fault monitoring arrangements using the chip register map.

2 Clock Loss Monitors

Each of the four inputs (IN0, IN1, IN2, IN3) are monitored for Clock Loss in terms of missing edges to indicate a loss of input signal. The number of edges used to indicate a clock loss (or recovery from a clock loss) is programmable in the SiT9514x GUI interface allowing for flexibility in choosing these thresholds. In addition, there is a programmable "Wait Time" all of which should be interpreted as follows:

Assertion of Clock Loss

We declare a CL if "Trigger Edge" number of consecutive edges are missing. The "Trigger Edge" parameter is programmable in the chip GUI.

De-Assertion of Clock Loss

We declare a ~CL if the clock is back and has less than "Clear Edge" consecutive edges missing. The "Clear Edge" parameter is programmable in the chip GUI.

Wait Time: After the clock is established to have returned, it is ensured that no CL error as defined by the de-assertion threshold occurs for "Val Time" seconds. This valid time is programmable by using the chip GUI with the following options: {2m, 100m, 200m, 1} sec. The use of this valid wait time ensures that sporadic edges in the input clock (such as ones caused by noise on floating nodes or intermittent unstable clock edges) does not de-assert clock loss and it is established over a user determined period of time that the input clock is available and stable.

2.1 Frequency Drift Monitors

Any of the four input clocks or the embedded MEMS clock can be used as the Golden Clock for calculating the frequency drifts of the other input clocks. The Golden Clock can be chosen in the GUI and is used as the "0 ppm" Reference Clock for all monitoring.

Fine Frequency Drift has a step size of ±2 ppm.

Fine Frequency Drift has a range from ±2 to ±510 ppm and an independent threshold is programmable for "Set" (for setting the FD monitor) and for "Clear" (for clearing the FD monitor).

Fine Frequency Drift has an implicit hysteresis with resolution of ±2 ppm since the same range is available for the FD assertion and de-assertion. Use of hysteresis prevents unwanted oscillation of the FD monitor output at the decision threshold and is recommended for robust operation. The value of the FD threshold hysteresis is implicit in the choice of the set and clear thresholds.

The Fine Frequency Drift monitors provide **precise** information for input clock frequency drift. However, since the resolution of the measurement determines time for the measurement, an alternate faster measurement mechanism for drift is needed. This is Coarse Frequency Drift which has coarser measurement but is **fast**. It is available for cases where the drift is very fast in the input frequency and is programmable from options as shown below.

Coarse Frequency Drift has a step size of ±100 ppm.

Coarse Frequency Drift has a range from ±100 to ±1600 ppm and an independent threshold is programmable for "Set" (for setting the FD monitor) and for "Clear" (for clearing the FD monitor).

Coarse Frequency Drift has an implicit hysteresis with resolution of ±100 ppm since the same range is available for the FD assertion and de-assertion. Use of hysteresis prevents unwanted oscillation of the FD monitor output

at the decision threshold and is recommended for robust operation. The value of the FD threshold hysteresis is implicit in the choice of the set and clear thresholds.

Important note regarding the above monitors with respect to clock switch in the PLL:

Normally the CL monitor is used for ascertaining a clock lose for the PLL to switch to a secondary reference or proceed to Holdover. However, the Fine and/or Coarse FD monitors can also be used in addition to the CL monitor to cause a PLL switch. This implements an “OR” logic for the FD Monitors to be used in addition to the CL monitors for triggering a PLL input clock switch or entry to Holdover. This is programmable as an option in the GUI.

2.2 Lock Loss Monitors

Lock loss alarm bits are programmable for each PLL. These bits assert in case of the input PLL reference clock signal phase and the feedback clock signal one of the same PLL are different as per the programmed assertion and de-assertion phase thresholds.

The Set threshold for asserting the LL monitor is programmable from {±0.2, ±0.4, ±2, ±4, ±20, ±40, ±200, ±400, ±2000, ±4000} ppm while the Clear threshold for de-asserting the LL monitor is programmable from {±0.2, ±0.4, ±2, ±200} ppm. A pre-determined level of hysteresis is implicit by choosing appropriately the set and clear thresholds for the LL monitor.

There is a delay in time from the point where the actual frequency value specified in ppm is less than de-asserted threshold in ppm. LL bits never assert during this delay period. The choice of this delay is with a timer that ensures that the delay is in line with the BW of the PLL loop. It is fully programmable from the GUI and is useful to ensure complete settling of the PLL without un-necessary toggling before LL de-assertion.

2.3 XO (MEMS) Clock Loss Monitors

The XO (MEMS) Clock Loss Monitor asserts the XO Clock Loss Alarm bits when the internal MEMS clock is not available.

3 Monitoring through the register map read back: Status and Notify

SiT9514x provides various Status and notify bits that can be accessed from the register map. Below are the details of the procedure to be followed to access the same.

The alarm registers are a set of three types of registers distributed between the various pages as described in the table below and illustrated with some examples in a later section of this application note.

- The Status registers are the current dynamic status of a defect. The live status defects are active high with a '1' indicating the defect is present.
- The Notify registers are the sticky bits for a defect. Sometimes, a very short status pulse occurs, which may not be possible for the external user to capture. Hence a notify register is provided. The notify register is set to 1 whenever there is a rising edge of the corresponding status register. This is a sticky bit and stays at 1 until the user writes a 1 to that specific bit to clear it.
- Each notify bit has a masking bit to enable or disable its operation. The notify sticky bit operates only if the corresponding masking bit is set to 1. If the masking register bit is set to 0, notify will not be asserted even when status toggles. The default value for the mask register is 0xff so all the notify signals are enabled. Once the user writes a 1 to clear the notify, the notify bit can again go high on the next rising edge of the status.

These registers operate on the internal 4 MHz RC clock. When there is a defect (i.e., status) of any bit in register it gets asserted and de-asserted in a live mode. User can read the corresponding register location to see the status at any time.

On the other hand, the default value of the notify and masking register is 0xff – user must write 0xff to both these registers to clear them at the beginning and use all notifies.

3.1 Tabular listing

All details of the alarm registers are in Table 1. To access a page of the register map, the page number has to be written to address 0xff. For instance, to either write to or read from page 1, first the user needs to write to 0xff address a value of 0x01. Following this, any number of write or read operations can be done with page 1.

SL. Nr.	Name of signal	Description	Page Nr.	Register Address	Bit Nr.
1	xo_clkloss_dynamic_status	XO (MEMS) clock dynamic status	00	0x02	2
2	xo_clkloss_dynamic_ntfy	XO (MEMS) clock dynamic Notify (write 0x04 with 0x02 to enable)	00	0x03	2
3	plla_lol_dyn_status	pll lol dynamic status	00	0x06	0
4	pllb_lol_dyn_status	pll lol dynamic status	00	0x06	1
5	pllc_lol_dyn_status	pll lol dynamic status	00	0x06	2

SL. Nr.	Name of signal	Description	Page Nr.	Register Address	Bit Nr.
6	plld_lol_dyn_status	pll lol dynamic status	00	0x06	3
7	plla_ho_frz_status	pll ho freeze dynamic status	00	0x06	4
8	pllb_ho_frz_status	pll ho freeze dynamic status	00	0x06	5
9	pllc_ho_frz_status	pll ho freeze dynamic status	00	0x06	6
10	plld_ho_frz_status	pll ho freeze dynamic status	00	0x06	7
11	plla_lol_dyn_ntfy	pll lol dynamic Notify (write 0x08 with 0x01 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	0
12	pllb_lol_dyn_ntfy	pll lol dynamic Notify (write 0x08 with 0x02 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	1
13	pllc_lol_dyn_ntfy	pll lol dynamic Notify (write 0x08 with 0x04 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	2
14	plld_lol_dyn_ntfy	pll lol dynamic Notify (write 0x08 with 0x08 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	3
15	plla_ho_frz_ntfy	pll ho freeze dynamic Notify (write 0x08 with 0x10 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	4
16	pllb_ho_frz_ntfy	pll ho freeze dynamic Notify (write 0x08 with 0x20 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	5
17	pllc_ho_frz_ntfy	pll ho freeze dynamic Notify (write 0x08 with 0x40 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	6
18	plld_ho_frz_ntfy	pll ho freeze dynamic Notify (write 0x08 with 0x80 to enable selectively, 0xff to enable all notifies in 0x07)	00	0x07	7

SL. Nr.	Name of signal	Description	Page Nr.	Register Address	Bit Nr.
19	in0_clock_loss_dyn_status	Clock Loss Dynamic Status	01	0x02	0
20	In1_clock_loss_dyn_status	Clock Loss Dynamic Status	01	0x02	1
21	In2_clock_loss_dyn_status	Clock Loss Dynamic Status	01	0x02	2
22	In3_clock_loss_dyn_status	Clock Loss Dynamic Status	01	0x02	3
23	in0_clock_loss_fd_dyn_status	Clock Loss + FD Dynamic Status	01	0x02	4
24	In1_clock_loss_fd_dyn_status	Clock Loss + FD Dynamic Status	01	0x02	5
25	In2_clock_loss_fd_dyn_status	Clock Loss + FD Dynamic Status	01	0x02	6
26	In3_clock_loss_fd_dyn_status	Clock Loss + FD Dynamic Status	01	0x02	7
27	in0_clock_loss_ntfy	Clock Loss Dynamic Notify (write 0x04 with 0x01 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	0
28	In1_clock_loss_ntfy	Clock Loss Dynamic Notify (write 0x04 with 0x02 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	1
29	In2_clock_loss_ntfy	Clock Loss Dynamic Notify (write 0x04 with 0x04 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	2
30	In3_clock_loss_ntfy	Clock Loss Dynamic Notify (write 0x04 with 0x08 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	3
31	in0_clock_loss_ntfy	Clock Loss with Fd_dynamic Notify (write 0x04 with 0x10 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	4

SL. Nr.	Name of signal	Description	Page Nr.	Register Address	Bit Nr.
32	In1_clock_loss_ntfy	Clock Loss with Fd_dynamic Notify (write 0x04 with 0x20 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	5
33	In2_clock_loss_ntfy	Clock Loss with Fd_dynamic Notify (write 0x04 with 0x40 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	6
34	In3_clock_loss_ntfy	Clock Loss with Fd_dynamic Notify (write 0x04 with 0x80 to enable selectively, 0xff to enable all notifies in 0x03)	01	0x03	7
35	in0_fd_fine_dyn_status	Frequency drift dynamic status	01	0x06	0
36	In1_fd_fine_dyn_status	Frequency drift dynamic status	01	0x06	1
37	In2_fd_fine_dyn_status	Frequency drift dynamic status	01	0x06	2
38	In3_fd_fine_dyn_status	Frequency drift dynamic status	01	0x06	3
39	in0_fd_coarse_dyn_status	Frequency drift dynamic status	01	0x06	4
40	In1_fd_coarse_dyn_status	Frequency drift dynamic status	01	0x06	5
41	In2_fd_coarse_dyn_status	Frequency drift dynamic status	01	0x06	6
42	In3_fd_coarse_dyn_status	Frequency drift dynamic status	01	0x06	7
43	in0_fd_fine_ntfy	Frequency drift dynamic notify (write 0x08 with 0x01 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	0
44	In1_fd_fine_ntfy	Frequency drift dynamic notify (write 0x08 with 0x02 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	1

SL. Nr.	Name of signal	Description	Page Nr.	Register Address	Bit Nr.
45	In2_fd_fine_ntfy	Frequency drift dynamic notify (write 0x08 with 0x04 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	2
46	In3_fd_fine_ntfy	Frequency drift dynamic notify (write 0x08 with 0x08 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	3
47	in0_fd_coarse_ntfy	Frequency drift dynamic notify (write 0x08 with 0x10 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	4
48	In1_fd_coarse_ntfy	Frequency drift dynamic notify (write 0x08 with 0x20 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	5
49	In2_fd_coarse_ntfy	Frequency drift dynamic notify (write 0x08 with 0x40 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	6
50	In3_fd_coarse_ntfy	Frequency drift dynamic notify (write 0x08 with 0x80 to enable selectively, 0xff to enable all notifies in 0x07)	01	0x07	7
51	fast_lock_dynamic_status_plla	PLL Fast lock dynamic status	0a	0x06	1
52	fast_lock_dynamic_status_pllb	PLL Fast lock dynamic status	0b	0x06	1
53	fast_lock_dynamic_status_pllc	PLL Fast lock dynamic status	0c	0x06	1
54	fast_lock_dynamic_status_plld	PLL Fast lock dynamic status	0d	0x06	1
55	fast_lock_dynamic_ntfy_plla	PLL Fast lock dynamic Notify (write 0x08 with 0x02 to enable selectively, 0xff to enable all notifies in 0x07)	0a	0x07	1

SL. Nr.	Name of signal	Description	Page Nr.	Register Address	Bit Nr.
56	fast_lock_dynamic_ntfy_pllb	PLL Fast lock dynamic Notify (write 0x08 with 0x02 to enable selectively, 0xff to enable all notifies in 0x07)	0b	0x07	1
57	fast_lock_dynamic_ntfy_pllc	PLL Fast lock dynamic Notify (write 0x08 with 0x02 to enable selectively, 0xff to enable all notifies in 0x07)	0c	0x07	1
58	fast_lock_dynamic_ntfy_plld	PLL Fast lock dynamic Notify (write 0x08 with 0x02 to enable selectively, 0xff to enable all notifies in 0x07)	0d	0x07	1
59	ho_valid_dynamic_status_plla	PLL Ho Valid dynamic status	0a	0x06	2
60	ho_valid_dynamic_status_pllb	PLL Ho Valid dynamic status	0b	0x06	2
61	ho_valid_dynamic_status_pllc	PLL Ho Valid dynamic status	0c	0x06	2
62	ho_valid_dynamic_status_plld	PLL Ho Valid dynamic status	0d	0x06	2
63	ho_valid_dynamic_ntfy_plla	PLL Ho Valid dynamic Notify (write 0x08 with 0x04 to enable)	0a	0x07	2
64	ho_valid_dynamic_ntfy_pllb	PLL Ho Valid dynamic Notify (write 0x08 with 0x04 to enable selectively, 0xff to enable all notifies in 0x07)	0b	0x07	2
65	ho_valid_dynamic_ntfy_pllc	PLL Ho Valid dynamic Notify (write 0x08 with 0x04 to enable selectively, 0xff to enable all notifies in 0x07)	0c	0x07	2
66	ho_valid_dynamic_ntfy_plld	PLL Ho Valid dynamic Notify (write 0x08 with 0x04 to enable selectively, 0xff to enable all notifies in 0x07)	0d	0x07	2

3.1.1 Examples for Live Status Read Back

Some examples are presented based on the table above for reading the live status of the defects. In the pseudo code presented below:

- wr_cmd(address, data): refers to a “Write Command” where the corresponding data is written in to the specified register address
- x= rd_cmd(address): refers to a “Read Command” where the corresponding data is read from the specified register address and stored in the variable ‘x’
- y >> x: denotes a bit wise right shift on the number y by x bit locations y << x: denotes a bit wise left shift on the number y by x bit locations
- & is the logical AND operation (bit wise)

Dynamic registers to read the various alarm registers in the RealTime page.

1. Input Clocks CL and FD Related Real Time live status read back:
   wr_cmd(0xff, 0x01) # Program the CLKMON_SYS page number

   # Clock Loss dynamic status

   clock_loss_dyn_status = rd_cmd(0x02) & 0xff

   (clock_loss_dyn_status >> 0) & 0x01 // IN0 Status for CL, Read bit position [0]

   (clock_loss_dyn_status >> 1) & 0x01 // IN1 Status for CL, Read bit position [1]

   (clock_loss_dyn_status >> 2) & 0x01 // IN2 Status for CL, Read bit position [2]

   (clock_loss_dyn_status >> 3) & 0x01 // IN3 Status for CL, Read bit position [3]

   # Frequency Drift dynamic status

   fd_fine_dyn_status = rd_cmd(0x06) & 0x0f

   fine = (fd_fine_dyn_status >> 0) & 0x01 // IN0 Status for Fine FD, Read bit position [0]

   fine = (fd_fine_dyn_status >> 1) & 0x01 // IN1 Status for Fine FD, Read bit position [1]

   fine = (fd_fine_dyn_status >> 2) & 0x01 // IN2 Status for Fine FD, Read bit position [2]

   fine = (fd_fine_dyn_status >> 3) & 0x01 // IN3 Status for Fine FD, Read bit position [3]

   fd_coarse_dyn_status = rd_cmd(0x06) >> 4

   coarse = (fd_coarse_dyn_status >> 0) & 0x01 // IN0 Status for Coarse FD, Read bit position [4]

   coarse = (fd_coarse_dyn_status >> 1) & 0x01 // IN1 Status for Coarse FD, Read bit position [5]

   coarse = (fd_coarse_dyn_status >> 2) & 0x01 // IN2 Status for Coarse FD, Read bit position [6]

   coarse = (fd_coarse_dyn_status >> 3) & 0x01 // IN3 Status for Coarse FD, Read bit position [7]

2. PLL Related Real Time live status read back:

   wr_cmd(0xff, 0x00) # Program the GENERIC_SYS page number, Page 0

   pll_lol_ho_freeze_dyn_status = rd_cmd(0x06) & 0xff

   # PLL Lock Loss dynamic status

(pll_lol_ho_freeze_dyn_status >> 0) & 0x01 // PLLA Status for LL, Read bit position [0]

(pll_lol_ho_freeze_dyn_status >> 1) & 0x01 // PLLB Status for LL, Read bit position [1]
(pll_lol_ho_freeze_dyn_status >> 2) & 0x01 // PLLC Status for LL, Read bit position [2]
(pll_lol_ho_freeze_dyn_status >> 3) & 0x01 // PLLD Status for LL, Read bit position [3]

 # Holdover Status

(pll_lol_ho_freeze_dyn_status >> (0 + 4)) & 0x01 // PLLA Status for HO, Read bit position [4]
(pll_lol_ho_freeze_dyn_status >> (1 + 4)) & 0x01 // PLLA Status for HO, Read bit position [5]
(pll_lol_ho_freeze_dyn_status >> (2 + 4)) & 0x01 // PLLA Status for HO, Read bit position [6]
(pll_lol_ho_freeze_dyn_status >> (3 + 4)) & 0x01 // PLLA Status for HO, Read bit position [7]

3, XO (MEMS) clock loss Related Real Time live status read back:

 # CLOS_X1X2, XO (MEMS) Clock Loss

wr_cmd(0xff, 0x00) # Program the GENERIC_SYS page number, Page 0

clos_x1x2 = rd_cmd(0x02) & 0x04 // XO CL Status, Read bit position [2]

3.2 Examples of Sticky Bit Clearing

As described earlier, the sticky notify bits are cleared by writing a '1' to the corresponding notify bit itself. The notify bit by itself is enabled by writing a '1' to the corresponding mask bit.

In the pseudo code presented below,

rmw_cmd(addr,bit_loc,no_of_bits,data): denotes the read/modify/write operation where no_of_bits number of bits at bit_loc location (denoted as 7:0) is replaced with the data at address location addr.

```
def clr_intb_XO_CL():
# This function is used to clear the sticky notify for XO (MEMS) Clock Loss
# Write the page number
wr_cmd(0xff, 0)
# Information to clr ** Page 0: reg03[2] =1 **
addr = 0x3
bit_loc = 2
no_of_bits = 1
data = 1
rmw_cmd(addr,bit_loc,no_of_bits,data)


def clr_intb_LOL_HO_Freeze():
# This function is used to clear the sticky notify for loss of lock and holdover notify for all PLLs
# Write the page number
wr_cmd(0xff, 0)
# Information to clr ** Page 0: reg07[7:0] = 0xff **
```

```
addr = 0x7
bit_loc = 7
no_of_bits = 8
data = 0xff
rmw_cmd(addr,bit_loc,no_of_bits,data)

def clr_intb_CL():
# This function is used to clear the sticky notify for clear clock loss notify
# Write the page number
wr_cmd(0xff, 1)
# Information to clr ** Page1: reg03[3:0] =0x0f **
addr = 0x3
bit_loc = 3
no_of_bits = 4
data = 0x0f
rmw_cmd(addr,bit_loc,no_of_bits,data)
def clr_intb_drift():
# This function is used to clear the sticky notify for clear drift notify
# Write the page number
wr_cmd(0xff, 1)
# Information to clr ** Page1: reg07[7:0] =0xff **
addr = 0x7
bit_loc = 7
no_of_bits = 8
data = 0xff
rmw_cmd(addr,bit_loc,no_of_bits,data)

def clr_intrb():
'''
This is the main clear function
which calls the 4 clear functions
'''
clr_intb_XO_CL()
```

clr_intb_LOL_HO_Freeze()

clr_intb_CL()

clr_intb_drift()

4 Read back of external voltage supply dips

SiT9514x provides status bits that can be accessed from the register map for dips in the external supply voltages VDDIN and VDD indication. This read back is reliably possible until the supply is high enough for the internal digital blocks to function reliably (typically ~ 1.2V).

VDDIN Status can be read back on Page 0 at register location 0x02[1].

VDD Status can be read back on any enabled PLL page (A/B/C/D) at register location 0x02[2] or Page 3 at register location 0x02[0].

For each of these register locations, a read back of 1 indicates an anomaly such that the external voltage has dipped to lower than its specified value.

5 Pin based access for the defects and the sticky notify bits

SiT9514x provides various Status and notify bits that can be accessed from the register map as explained previously. Additionally, the same status can be read on the various IO pins of the chip. This is shown below to illustrate the function for each pin for getting the alarm status monitors.

Table 2. SiT95141/5/8 Pin Based Alarms

Pin Nr.	Name	Description
12	INTRb	This pin is a "NOR" operation for the various sticky notify bits available on the die. The user can choose from the GUI which notifies are expected to be used on the NOR operation. Volatile writes as defined with respect to Sticky Bit clearing are needed to clear the INTRb. This option is provided as the "Clear" button in the "Real Time" Menu of the GUI. The pin is then asserted low when any of the chosen defects has occurred.
47	LOLb	This is the live status for the Loss of Lock seen on this pin as a "NOR" operation of the Loss of Lock live status of each PLL. This implies that the LOLb pin is high if all PLLs being used are locked and 0 if any one of the PLLs being used is not locked.

Table 3. SiT95147 Pin Based Alarms

Pin Nr.	Name	Description
3	LLAb	This indicates an active low loss of lock status on PLLA (0 when PLL is unlocked and 1 when PLL is locked).
4	LLBb	This indicates an active low loss of lock status on PLLB (0 when PLL is unlocked and 1 when PLL is locked).
5	LLCb	This indicates an active low loss of lock status on PLLC (0 when PLL is unlocked and 1 when PLL is locked).
12	INTRb	This pin is a "NOR" operation for the various sticky notify bits available on the die. The user can choose from the GUI which notifies are expected to be used on the NOR operation. Volatile writes as defined with respect to Sticky Bit clearing are needed to clear the INTRb. This option is provided as the "Clear" button in the "Real Time" Menu of the GUI. The pin is then asserted low when any of the chosen defects has occurred.
25	LOS_XOb	This is a clock loss alarm on XO pathway indicating a loss on either the external XO clock or the internal MEMS oscillator's clock.
47	LLDb	This indicates an active low loss of lock status on PLLD (0 when PLL is unlocked and 1 when PLL is locked).

6 PLL re-lock and re-calibration using register writes

SiT9514x provides an option to re-lock the PLL using register writes. However, a case can arise in the field where the PLL loses lock and a processor detecting that might attempt to re-lock or re-calibrate the same PLL with the same output frequencies.

This is done as per the following procedure:

(wr_cmd(address, data): refers to a “Write Command” where the corresponding data is written in to the specified register address.)

Register Write 1: Go to the appropriate PLL page

wr_cmd(0xff, Page#)

Change page number to the appropriate PLL.
Page # is 0x0A for PLLA, 0x0B for PLLB, 0x0C for PLLC, 0x0D for PLLD.

Register Write 2: Power down the relevant Output Clocks from this PLL.
(This is an optional step to power down all the clocks from the relevant PLL in a glitch free manner.)

wr_cmd(0xc0, 0x01)

Register Write 3 and 4: Perform the large trigger change on the PLL to re-lock / re-calibrate the PLL. (These are 2 necessary register writes needed to perform a re-lock on the PLL).

wr_cmd(0x05, 0x01)
wr_cmd(0x05, 0x00)

Register Write 5: Power up the relevant Output Clocks from this PLL again.
(Register Write 5 is needed only if the optional Register Write 2 was done.)

wr_cmd(0xc0, 0x00)

7 Revision history

Table 1: Document revision history

Revision	Date	Description
1.0	25 Aug 2022	Initial release

SiTime Corporation, 5451 Patrick Henry Drive, Santa Clara, CA 95054, USA | Phone: +1-408-328-4400 | Fax: +1-408-328-4439

© SiTime Corporation, August 2022. The information contained herein is subject to change at any time without notice. SiTime assumes no responsibility or liability for any loss, damage or defect of a Product which is caused in whole or in part by (i) use of any circuitry other than circuitry embodied in a SiTime product, (ii) misuse or abuse including static discharge, neglect or accident, (iii) unauthorized modification or repairs which have been soldered or altered during assembly and are not capable of being tested by SiTime under its normal test conditions, or (iv) improper installation, storage, handling, warehousing or transportation, or (v) being subjected to unusual physical, thermal, or electrical stress.

Disclaimer: SiTime makes no warranty of any kind, express or implied, with regard to this material, and specifically disclaims any and all express or implied warranties, either in fact or by operation of law, statutory or otherwise, including the implied warranties of merchantability and fitness for use or a particular purpose, and any implied warranty arising from course of dealing or usage of trade, as well as any common-law duties relating to accuracy or lack of negligence, with respect to this material, any SiTime product and any product documentation. Products sold by SiTime are not suitable or intended to be used in a life support application or component, to operate nuclear facilities, or in other mission critical applications where human life may be involved or at stake. All sales are made conditioned upon compliance with the critical uses policy set forth below.

CRITICAL USE EXCLUSION POLICY

BUYER AGREES NOT TO USE SITIME'S PRODUCTS FOR ANY APPLICATION OR IN ANY COMPONENTS USED IN LIFE SUPPORT DEVICES OR TO OPERATE NUCLEAR FACILITIES OR FOR USE IN OTHER MISSION-CRITICAL APPLICATIONS OR COMPONENTS WHERE HUMAN LIFE OR PROPERTY MAY BE AT STAKE.

SiTime owns all rights, title and interest to the intellectual property related to SiTime's products, including any software, firmware, copyright, patent, or trademark. The sale of SiTime products does not convey or imply any license under patent or other rights. SiTime retains the copyright and trademark rights in all documents, catalogs and plans supplied pursuant to or ancillary to the sale of products or services by SiTime. Unless otherwise agreed to in writing by SiTime, any reproduction, modification, translation, compilation, or representation of this material shall be strictly prohibited.