AN5266
Application note

SPC58xNx/SPC58xEx/SPC58xGX FCCU fault sources and reaction

Introduction

This application note describes the FCCU input fault sources. Furthermore, for each of them, it describes how to verify the integrity of the error reaction path and the recommended methods to inject a fault.

The device mentioned in this document is the SPC58xNx/SPC58xE/SPC58xG (40 nm - ASIL D featured). Most of the concepts, however, are also valid for the other devices belonging to the 40 nm and 55 nm families of SPC5 32-bit Automotive MCUs.

Before reading this document, the reader should have a clear understanding about the usage of FCCU. Refer to "Fault Collection and Control Unit (FCCU)" chapter in the SPC58xEx/SPC58xNx/xGx Microcontroller Reference Manual for further details on this module. Refer also to the SPC58xEx/SPC58xNx/xGx errata sheet.

Contents

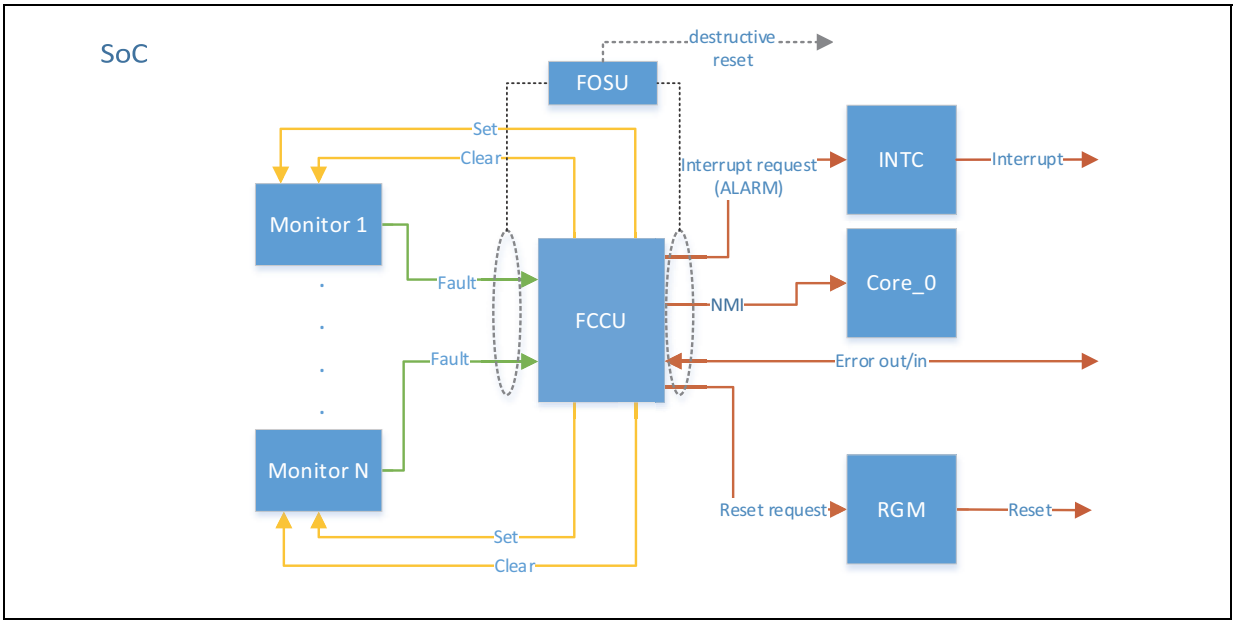
List of tables

List of figures

1 Overview

The FCCU is a key element of the functional safety concept of the SPC58 and SPC57 families of SPC5 32-bit Automotive MCUs. It is responsible for collecting and reacting to failure notifications coming from different modules indicated as monitors. Examples of monitors are CMU, MEMU, XBIC and so forth.

Figure 1. FCCU Monitor to reaction on path



Note: Some monitors might miss the Set and Clear signals.

FCCU Inner diagram1 shows how the FCCU is connected to the other blocks of the SoC. The reader shall consider this figure – and all other figures in this document – as a block diagram that not exactly reflects the physical implementation in the silicon.

In case of a fault, the FCCU can move the device into the safe state[a] without any CPU intervention. Since the FCCU and the whole error reaction path are prone to latent failures, the safety concept requires the execution of a software test to verify the integrity of the error reaction path. The user shall run this software test for each used and testable channel of the FCCU at least once per Trip time[b].

This document goes through the list of the faults reported by the FCCU. For each of them it describes how to test the reaction path in order to fulfill the requirement of the safety manual. Note that the user can't test the error reaction path for few monitors.

Table 1 lists and describes all FCCU input fault sources for SPC58xEx/SPC58xNx/xGx.

a. It's worth noticing that our safety concept doesn't consider the Safe Mode of the Mode Entry as a safety states. The safety manual lists the considered safe states.

b. The safety analysis assumes a Trip time of 10 hours.

Table 1. List of faults

Channel #	Source	Description
0	PMC DIG	Temperature out of range from TSENS detector
1	PMC DIG	Supply voltage out of range from LVDs
2	PMC DIG	Supply voltage out of range from HVDs
3	PMC DIG	Digital PMC initialization error during DCF data load.
4	PMC DIG	Voltage detector BIST error
5	SSCM/FLASH	Transfer error (during the SSCM and STCU DCF loading) OR FLASH memory initialization error
6	STCU	BIST result - wrong signature (STCU Unrecoverable Fault)
7	STCU	BIST result - wrong signature (STCU Recoverable Fault)
8	STCU	Unwanted activation of the online self-test
9	GLUE LOGIC	JTAG or NPC not in reset or activation of dangerous debug functionality. Spurious activation of SSCM
10	CORE 0	Lockstep out of sync due to a fault in the Core 0 or associated DSMC
11	—	Not implemented
12	CORE 2	Lockstep out of sync due to a fault in the Core 2 or associated DSMC
13	—	Not implemented
14	INTC	Lockstep out of sync due to a fault in the INTC
15	DMA 0	Lockstep out of sync due to a fault in the DMA 0
16	DMA 1	Lockstep out of sync due to a fault in the DMA 1
17	SWT	Software watchdog timer 3 reset request
18	SWT	Software watchdog timer 2 reset request
19	SWT	Software watchdog timer 1 reset request
20	SWT	Software watchdog timer 0 reset request
21	MEMU	A new error is recorded in the MEMU SysRAM correctable error table
22	MEMU	A new error is recorded in the MEMU SysRAM uncorrectable error table
23	MEMU	A new error is recorded in the MEMU SysRAM overflow error table

Table 1. List of faults (continued)

Channel #	Source	Description
24	MEMU	A new error is recorded in the MEMU Peripheral RAM correctable error table
25	MEMU	A new error is recorded in the MEMU Peripheral RAM uncorrectable error table
26	MEMU	A new error is recorded in the MEMU Peripheral RAM overflow error table
27	MEMU	A new error is recorded in the MEMU FLASH correctable error table
28	MEMU	A new error is recorded in the MEMU FLASH uncorrectable error table
29	MEMU	A new error is recorded in the MEMU FLASH overflow error table
30	IMA	IMA is active
31	SMPU	SMPU XBAR 0 Monitor incorrectly refuses an access
32	SMPU	SMPU XBAR 1 Monitor incorrectly refuses an access
33	SMPU	SMPU XBAR 0 Monitor correctly refuses an access
34	SMPU	SMPU XBAR 1 Monitor correctly refuses an access
35	CORE 0	I-MEM address feedback error
36	CORE 0	D-MEM address feedback error
37	CORE 0	I-CACHE address feedback error
38	CORE 0	D-CACHE address feedback error
39	CORE 1	I-MEM address feedback error
40	CORE 1	D-MEM address feedback error
41	CORE 1	I-CACHE address feedback error
42	CORE 1	D-CACHE address feedback error
43	CORE 2	I-MEM address feedback error
44	CORE 2	D-MEM address feedback error
45	CORE 2	I-CACHE address feedback error
46	CORE 2	D-CACHE address feedback error
47	DMA 0	DMA 0 TCD RAM address feedback error
48	DMA 1	DMA 1 TCD RAM address feedback error
49	PLL DIG	PLL0 Loss of lock error
50	PLL DIG	PLL1 Loss of lock error
51	CMU	Loss of XOSC clock (OLR signal from CMU0)
52	CMU	System clock frequency out of range (FHH or FLL signal from CMU0)
53	CMU	Platform clock frequency out of range (FHH or FLL signal from a CMU belonging to CMU_Platform group)
54	CMU	Other clocks frequency out of range (FHH or FLL signal from a CMU belonging to CMU_other group)
55	IAHBG	Error in the frequency signaled by the gasket between the XBAR 1 and the XBAR 0

Table 1. List of faults (continued)

Channel #	Source	Description
56	XBAR 1	Crossbar integrity error signaled by XBIC
57	XBAR 0	Crossbar integrity error signaled by XBIC
58	FLASH 0	EDC after ECC error from FLASH array
59	PFLASHC 0	EDC after ECC error from FLASH controller
60	FLASH 0	FLASH Encoding Error
61	PFLASHC 0	FLASH Controller Address feedback checker error
62	PFLASHC 0	Transaction monitor mismatch in pseudoreplicated calibration hardware
63	FLASH 0 / 1	FLASH read reference error
64	FLASH 1	EDC after ECC error from FLASH array
65	PFLASHC 1	EDC after ECC error from FLASH controller
66	FLASH 1	FLASH Encoding Error
67	PFLASHC 1	FLASH Controller Address feedback checker error
68	PFLASHC 1	Transaction monitor mismatch in pseudoreplicated calibration hardware (from PFLASHC)
69	PRAM (Overlay)	Transaction monitor mismatch in pseudoreplicated calibration hardware (from on-chip calibration RAM)
70	PRAM 0	Indication of an addressing error in system RAM or a write error in the RAM controller resulting in corrupted RAM access.
71	PRAM 0	EDC after ECC for RAM (read-modify-write ECC)
72	Concentrator 0	Indication of an hardware fault affecting the control signals of the gasket
73	Concentrator 1	Indication of an hardware fault affecting the control signals of the gasket
74	PRAM 2	Indication of an addressing error in system RAM or a write error in the RAM controller resulting in corrupted RAM access.
75	PRAM 2	EDC after ECC for RAM (read-modify-write ECC)
76	CORE 1(SPC58xN) PRAM 3 (SPC58xE, SPC58xG)	Indication of an hardware fault affecting the control signals of the DSMC associated to the core 1(SPC58xN) Indication of an addressing error in system RAM or a write error in the RAM controller resulting in corrupted RAM access. (SPC58xE, SPC58xG)
77	PRAM 3 (SPC58xE, SPC58xG)	Not implemented in SPC58xN EDC after ECC error for RAM (read-modify-write ECC) (SPC58xE, SPC58xG)
78	DFT1	Design for testability - Test circuitry Group 1 activation.
79	DFT2	Design for testability - Test circuitry Group 2 activation.
80	DFT3	Design for testability - Test circuitry Group 3 activation.
81	DFT4	Design for testability - Test circuitry Group 4 activation.
82	CORE 0	Machine check exception indication
83	CORE 1	Machine check exception indication
84	CORE 2	Machine check exception indication

Table 1. List of faults (continued)

Channel #	Source	Description
85	CORE 0	Indication of disablement of lockstep mode
86	AIPS 0	Error detected by the PBRIDGE monitor (AIC) during a read/write transaction
87	AIPS 0	Error detection (EDC) during a transaction trough the AIPS
88	AIPS 1	Error detected by the PBRIDGE monitor (AIC) during a read/write transaction
89	AIPS 1	Error detection (EDC) during a transaction trough the AIPS
90	AIPS 2	Error detected by the PBRIDGE monitor (AIC) during a read/write transaction
91	AIPS 2	Error detection (EDC) during a transaction trough the AIPS
92	CONCENTRATOR 0	Internal Crossbar integrity error signaled by its internal XBIC
93	CONCENTRATOR 0	EDC after ECC error in Concentrator 0
94	MC_RGM	Safe mode entry request from RGM
95	COMPENSATION CELLS	Indication of undesired deactivation of pad compensation
96	FCCU EIN	Error input pin (from the external world)
97	—	Reserved - Fake fault error injection should be avoided
98	—	Reserved - Fake fault error injection should be avoided
99	CONCENTRATOR 0	Error in the frequency gasket in the concentrator 0
100	SWT	First Timeout event of SWT3 which is also mapped to INTC
101	SWT	First Timeout event of SWT2 which is also mapped to INTC
102	SWT	First Timeout event of SWT1 which is also mapped to INTC
103	SWT	First Timeout event of SWT0 which is also mapped to INTC
104	CONCENTRATOR 1	EDC after ECC error alarm for – SIPI1 Debug
105	CONCENTRATOR 1	EDC after ECC error alarm for – FlexRay 0
106	CONCENTRATOR 1	EDC after ECC error alarm for – Ethernet 0
107	CONCENTRATOR 2	Not implemented in SPC58xN EDC after ECC error alarm for – Ethernet 1(SPC58xE, SPC58xG)
108	CONCENTRATOR 1	Error in the frequency gasket in the concentrator 1
109	CONCENTRATOR 2	Not implemented in SPC58xN Internal Crossbar integrity error signaled by its internal XBIC (SPC58xE, SPC58xG)
110	CONCENTRATOR 1	Internal Crossbar integrity error signaled by its internal XBIC
111	CORE 2	Indication of disablement of lockstep mode
112	INTC	Indication of disablement of lockstep mode
113	DMA 0	Indication of disablement of lockstep mode

Table 1. List of faults (continued)

Channel #	Source	Description
114	DMA 1	Indication of disablement of lockstep mode
115	FLASH 0 / 1	FLASH internal unrecoverable error

Before the safety application starts, the user shall configure and set a proper reaction for each FCCU input fault source. See “FCCU register reset values” paragraph in SPC58xEx/SPC58xNx/xGx Microcontroller Reference Manual for the device default configuration.

Possible reactions to a failure are:

- Internal reactions[c]:
  - No reset reaction
  - IRQ
  - Long or Short functional reset
  - NMI to Core 0
- External reaction:
  - Error out (EOUT) signaling

Once it is enabled[d], the FCCU controls EOUT pins that signal the status of the MCU without any intervention of the core.

A dedicated module, i.e. the FOSU, monitors the integrity of the FCCU. The FOSU triggers a destructive reset if its internal counter reaches a timeout before the FCCU takes a reaction to an incoming – and enabled[e] – fault. The FOSU does not require any configuration done by the user. A functional reset has not impact on the FCCU.

c. The user can configure separately the internal reaction for each FCCU input.

d. The EOUT pins are enabled by the FCCU_SET_AFTER_RESET flag of the FCCU_CFG register.

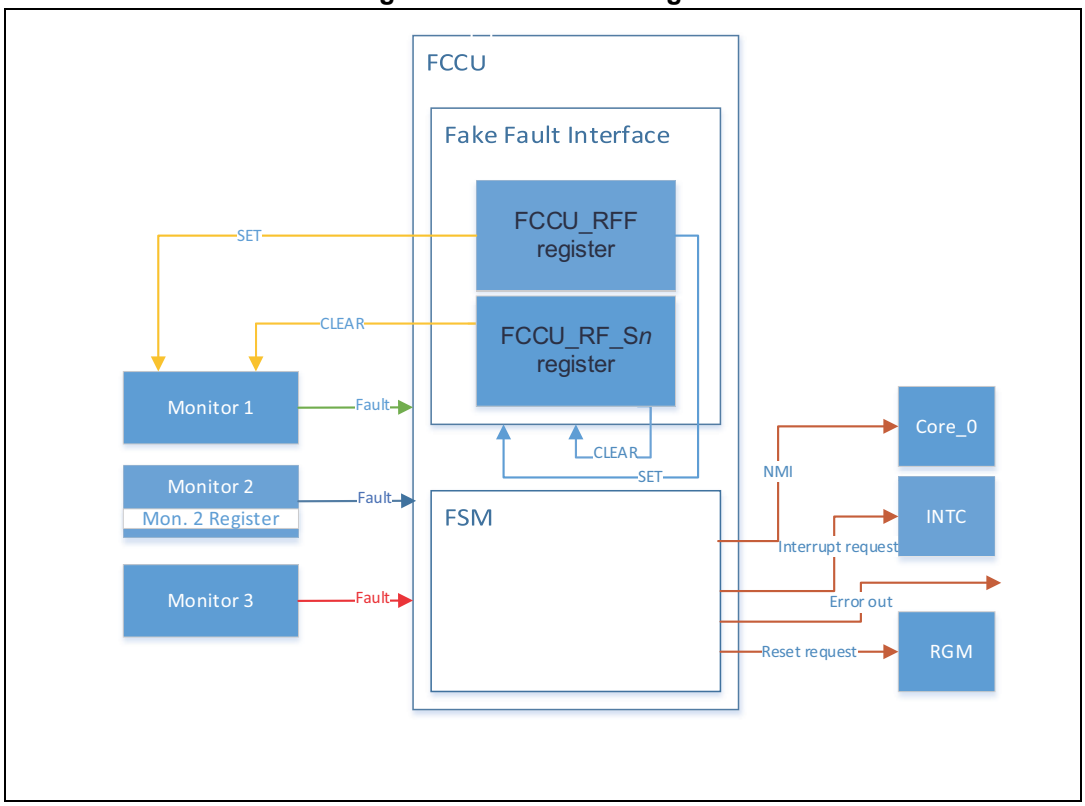
e. The FOSU ignores the incoming faults that aren’t enabled through the corresponding bit in the FCCU_RF_En register.

2 FCCU Fault injection, clearing and fake fault interface

The application can use the fault injection to diagnose physical defects affecting the connections between the hardware monitors and the FCCU. The procedure to inject a fault depends on the specific monitor. We can distinguish among three different sets of fault inputs:

1. faults that are injectable by using the FCCU fake fault interface (“Mon 1” in FCCU Inner diagram) (so called Long loop injection)
2. faults that are injectable by using a SW procedure to stimulate the fault (“Mon 2” in *Figure 2*) (so called Short loop injection)
3. faults that are not injectable (“Mon 3” in *Figure 2*).

Figure 2. FCCU Inner diagram



For the first group, the FCCU can trigger a fault event directly in the monitor (even if no real failure occurs) through the fake fault interface.

To generate this fault event, an additional - and optional[f]- signal is available (SET signal in FCCU Inner diagram, yellow arrow). The user can inject a fake fault by asserting the bit corresponding to the channel under test into the FCCU_RFF register. To clear a fake fault directly in the monitor, an additional - and optional[f]- signal is available (CLEAR signal in FCCU Inner diagram, yellow arrow). The de-assertion of the FCCU_RF_Sn status bit

f. Not all monitors embed this interface. When available, fake fault injection method is suggested in the following sections.

indicates that the software has properly cleared the fake fault in the monitor. Some monitors miss the SET and CLEAR signals. With such a method, the user can debug the reaction of the FCCU, but they can't verify the integrity of the connection between these monitors and the FCCU. For the second group, the application can trigger a fault event writing into a specific register of the monitor or using a SW procedure (e.g. injection of ECC errors, setting wrong thresholds, willfully bad operations). To clear a fault that belongs to this group, the application must clear the fault indication in the monitor(g) and afterward de-assert the corresponding FCCU_RF_Sn status bit.

It is not possible to test the connection for the faults that belong to the third group. The safety analysis considers and assesses this limitation.

The reaction path test procedure must also include the check of FCCU's reaction (EOUT, Interrupt, NMI, Reset request) at least for one FCCU input fault.

Depending on the monitor, the fault indication is a pulse, i.e. fault edge triggered, or a constant value, fault level triggered.

The fault management shall consider that the user can configure a fault as:

- HW recoverable fault, i.e. the fault status within the FCCU remains asserted until the monitor keeps the fault indication asserted. As soon as the monitor clears the fault indication, it also clears the fault status within the FCCU.
- SW recoverable fault, i.e. the fault status within the FCCU remains asserted until the software clears it even if the monitor de-asserts the fault indication.

The generic recommendation is to configure all faults as SW recoverable. In such a way, the FCCU clears the respective status flag only after an explicit request from the software. In case of HW recoverable, the status flag automatically clears and the application may not react properly to the incoming fault.

g. This is not necessary for monitors that have a pulse fault indication

3 Faults description

The following sections describe all the faults collected by FCCU for SPC58xEx/SPC58xNx/xGx device and how – if possible - to inject them for checking the integrity of the relevant reaction path.
The following convention is adopted in following figures:

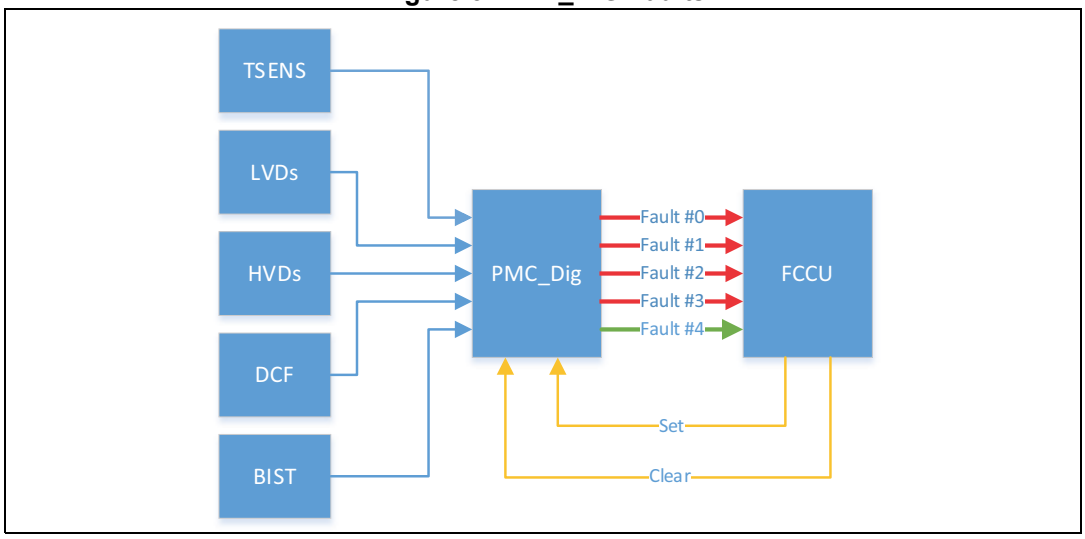
1. a GREEN arrow marks the faults as injectable by the FCCU fake fault interface
2. a BLUE arrow marks the faults as injectable by using a SW procedure to stimulate the fault
3. a RED arrow marks the faults that the user can't inject

Refer to device SPC58EHx/SPC58NHx Microcontroller Reference Manual for further details on the mentioned modules.

3.1 PMC_DIG Faults

PMC_DIG is the source of five different FCCU input faults.

Figure 3. PMC_DIG Faults



3.1.1 Temperature out of range from TSENS (Fault #0)

The temperature sensor generates an output voltage that is proportional to the internal junction temperature of the device. When this temperature exceeds the defined thresholds[h], the PMC_DIG forwards this fault to the FCCU.

The SSCM loads the trim values of the temperature sensor from the FLASH by the DCF record interface during the boot phase.

The user can't inject this fault.

h. There are three thresholds: TS0, TS1 and TS2. User can configure them in the PMC_DIG FEE_TD register.

3.1.2 Voltage out of range from LVDs (Fault #1)

The PMC_DIG forwards this fault to the FCCU when an LVD detects a voltage that drops below the defined threshold[i]. The MCU embeds several LVDs (see SPC58xEx/SPC58xNx/xGx Microcontroller Reference Manual for further details on LVDs) and their output signals are OR'ed before arriving to the FCCU.

The SSCM loads their trim values from the FLASH by the DCF record interface during the boot phase.

The user can't inject this fault.

3.1.3 Voltage out of range from HVDs (Fault #2)

The PMC_DIG forwards this fault to the FCCU when an HVD detects a voltage that raises above the defined threshold[j]. The MCU embeds several HVDs (see SPC58xEx/SPC58xNx/xGx Microcontroller Reference Manual for further details on HVDs) and their output signals are OR'ed before arriving to the FCCU.

The SSCM loads the trim values from the FLASH by the DCF record interface during the boot phase.

The user can't inject this fault.

3.1.4 PMC_Dig DCF Safety Error (Fault #3)

DCF records are used to configure certain registers in the device during system boot, e.g. trimming values being used for analog modules (e.g. PMC, Temperature Sensor and ADC Bandgap). If an error occurs while the SSCM loads the DCF records into the PMC, the PMC_DIG forwards this fault to the FCCU.

The user can't inject this fault.

3.1.5 Voltage detector BIST (Fault #4)

The BIST is a testing procedure initiated by software. The voltage detector BIST verifies the integrity of all the supply voltage monitors. In case the BIST fails, the PMC_DIG forwards this fault to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.2 SCM/FLASH Fault

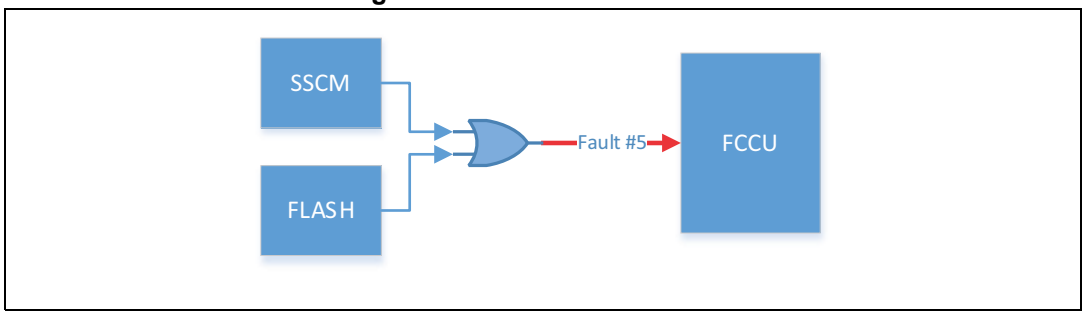
The SSCM reads the system configuration from the FLASH[k] and pushes it to the various clients inside the microcontroller.

i. Corresponding bit in the PMC_Dig FEE_LVx must be enabled.

j. Corresponding bit in the PMC_Dig FEE_HVx must be enabled.

k. System configuration is mainly saved as DCF records that are located either in the Test or UTest sectors of the flash.

Figure 4. SSCM / FLASH Faults



3.2.1 STCU Configuration error OR FLASH memory initialization error (Fault #5)

A fault can occur while the SSCM loads the STCU configuration. In this case, the SSCM forwards this fault to the FCCU. Moreover, an unexpected condition, e.g. ECC double-bit detections on the reset reads, can occur within the FLASH module during its initial configuration. In this case, the FLASH forwards this fault to the FCCU.

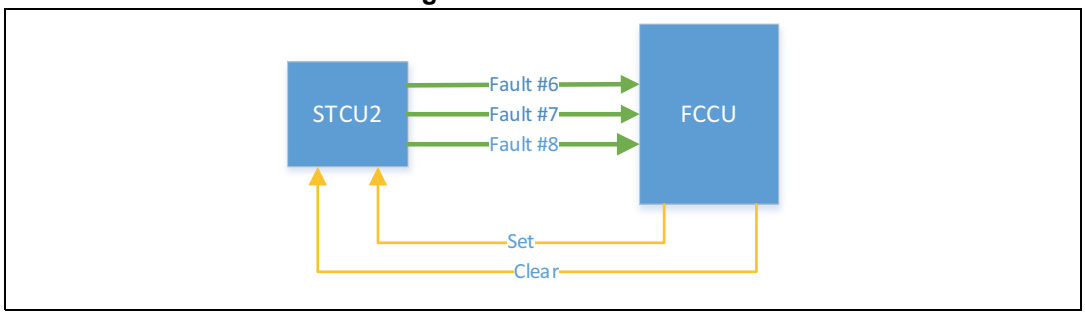
The FCCU Fault #5 is the resulting OR between the previous two signals.

The user can't inject this fault.

3.3 STCU Faults

The STCU is a programmable hardware module that controls the execution of the Self-Test during both the Off-line and/or On-line procedure. The STCU is the source of three different FCCU input faults.

Figure 5. STCU2 Faults



3.3.1 BIST result - wrong signature (STCU Unrecoverable Fault) (Fault #6)

If the BIST detects a fault that is configured as unrecoverable[I], the STCU forwards this fault to the FCCU. Although BIST targets permanent faults, a transient one may trigger a fault event.

I. The user shall configure the STCU to trigger either a recoverable or an unrecoverable fault if the BIST fails. This configuration is done by programming the proper DCF records.

Therefore, re-running the self-test may be the appropriate action to this fault. The STCU can trigger this fault only during BIST execution.

The user can inject this fault by the FCCU fake fault interface.

3.3.2 BIST result - wrong signature (STCU Recoverable Fault) (Fault #7)

If the BIST detects a fault that is configured as recoverable(j), the STCU forwards this fault to the FCCU. Although BIST targets permanent faults, a transient one may trigger a fault event.

Therefore, re-running the self-test may be the appropriate action to this fault. This fault can be triggered only during BIST execution.

The user can inject this fault by the FCU fake fault interface.

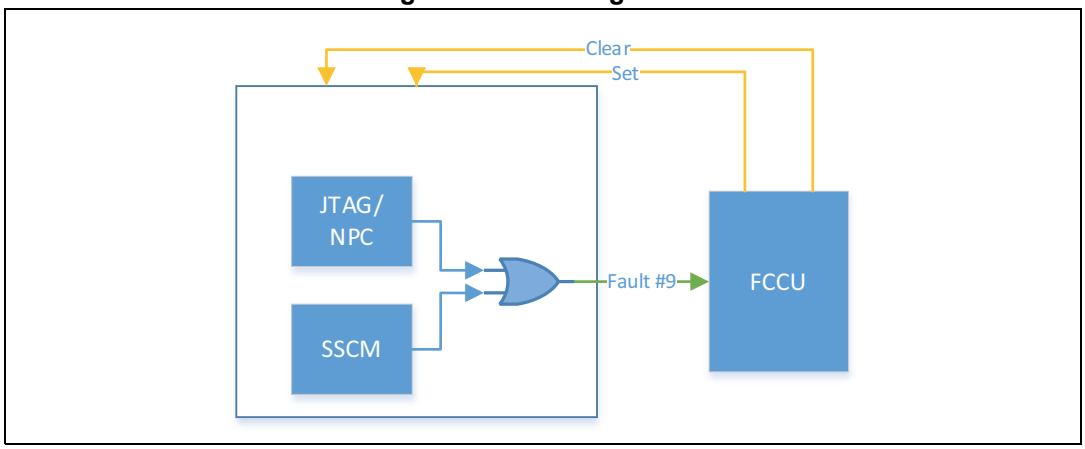
3.3.3 BIST control signals go to wrong condition during User application. (Fault #8)

If the online self-test activates when the RUNSW field of the register STCU_RUNSW is set to ‘0’, The STCU detects this unexpected condition and forwards it to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.4 GLUE Logic faults

Figure 6. GLUE Logic fault

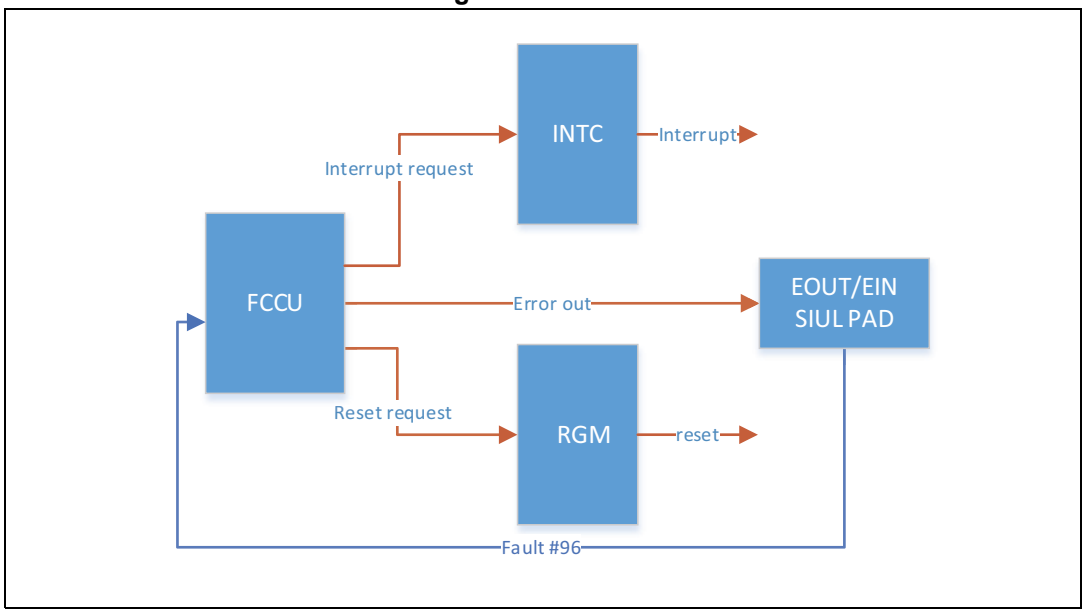


3.4.1 Activation of debug functionality. Spurious activation of SSCM (Fault #9)

The JTAG/NPC module monitors the unwanted activation of the JTAG/DEBUG Mode. If this event occurs, a dedicated logic forwards this fault to the FCCU. The hardware also monitors the unwanted activation of the SSCM and, if this event occurs, a dedicated logic forwards this fault to the FCCU. The FCCU Fault #9 is the resulting OR between the previous two signals.

The user can inject this fault by the FCCU fake fault interface.

Figure 7. EIN fault



3.4.2 External activation of EIN (Fault #96)

If an external device pulls down the EOUT/EIN pin, the MCU receives a notification of an external faulty condition. A dedicated logic forwards this fault to the FCCU.

The EOUT signal drives the EIN signal through an internal loopback. This connection results in a loop where if the FCCU is in the FAULT state, it drives the EOUT which in turn drives the EIN.

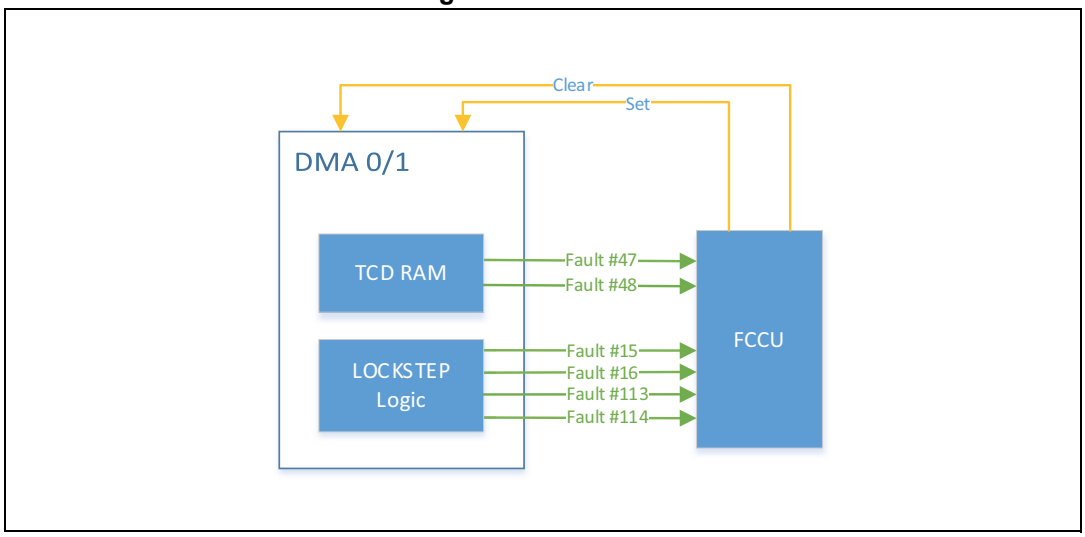
The user can inject this fault by a SW procedure that asserts the EOUT / EIN loopback, driving the EOUT trough the FCCU_SET_CLEAR field of FCCU_CFG register.

To clear the fault, the SW procedure must break the EOUT/EIN loopback by setting the GPIO mode of the EOUT pin for a time longer than the EOUT time to expire (T_min set using the FCCU_XTMR register).

3.5 DMA faults

The eDMA controller is a module capable of performing complex data transfers with minimal intervention of a host processor

Figure 8. DMA faults



3.5.1 DMA 0 Lockstep out of sync (Fault #15)

In case the operation of the DMA 0 and its replica are out of sync due to a fault, the RCCU signals the fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.5.2 DMA 1 Lockstep out of sync (Fault #16)

In case the operation of the DMA 1 and its replica are out of sync due to a fault, the RCCU signals the fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.5.3 DMA 0 TCD Ram feedback checker (Fault #47)

If latched control signals don't match with the ones originally sent to DMA 0 TCD RAM, the feedback checker forwards this fault to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.5.4 DMA 1 TCD Ram feedback checker (Fault #48)

If latched control signals don't match with the ones originally sent to DMA 1 TCD RAM, the feedback checker forwards this fault to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.5.5 DMA 0 Lockstep mode disablement (Fault #113)

In case of disablement of the LSM of DMA 0 happens, the event is forwarded to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.5.6 DMA 1 Lockstep mode disablement (Fault #114)

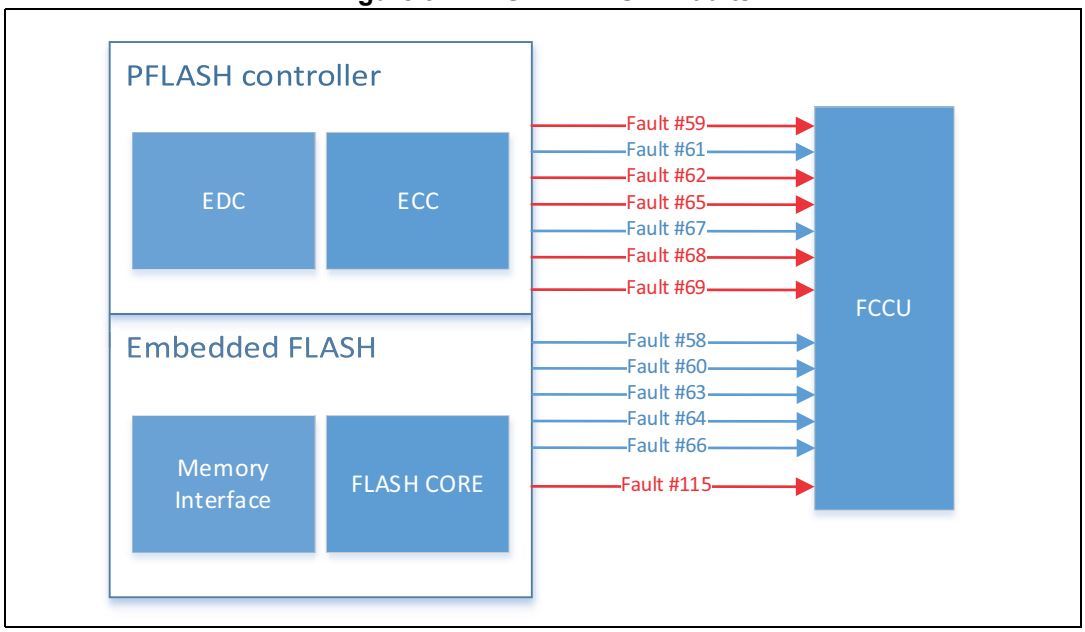
In case of disablement of the LSM of DMA 1 happens, the event is forwarded to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.6 FLASH/PFLASHC faults

The PFLASH controller is the interface between the FLASH memory and the crossbar switch. It provides FLASH configuration and control functions.

Figure 9. FLASH/PFLASHC faults



3.6.1 FLASH reset error (Fault #115)

FLASH forwards this fault to the FCCU in case one of the following unrecoverable errors occurs:

ECC errors on flash internal reads during configuration loading (startup);

ECC errors on flash internal reads during firmware[m] copy (startup);

Double ECC errors on KRAM[n] during internal self-check routine (always running).

The user can't inject this fault.

3.6.2 Current error in the FLASH memory array (Fault #63)

The FLASH monitors its internal current and voltage references. In case one of these values is out of the allowed range, the FLASH forwards this event to FCCU.

The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block.

m. This is the PFLASH controller firmware and doesn't refer to the user firmware. It's not accessible to the user.

n. This is a RAM internal to the PFLASHC not visible to the user.

3.6.3 EDC after ECC for FLASH Array 0/1 (Fault #58, Fault #64)

In case a hardware fault occurs in the ECC logic of the FLASH memory resulting in a corrupted ECC correction, the PFLASHC forwards this fault to FCCU. The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block.

3.6.4 EDC after ECC for FLASH Controller 0/1 (Fault #59, Fault #65)

In case a hardware fault occurs in the ECC logic of the FLASH controller resulting in a corrupted ECC correction, the PFLASHC forwards this fault to FCCU.

The user can't inject these faults

3.6.5 Encoding Error FLASH 0/1 (Fault #60, Fault #66)

In case a hardware fault occurs resulting in a corrupted FLASH memory access, the PFLASHC forwards this fault to FCCU.

The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block.

3.6.6 Address Feedback Error FLASH Controller 0/1 (Fault #61, Fault #67)

In case a hardware fault occurs resulting in a mismatch between the address from the crossbar and the feedback address from the FLASH, the PFLASHC forwards this fault to FCCU.

The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block[o].

3.6.7 Transaction monitor mismatch in calibration hardware FLASH Controller 0/1 (Fault #62, Fault #68)

The Flash memory controller supports calibration development by providing a remapping function to route Flash accesses to overlay RAM (on-chip overlay RAM, extended overlay RAM in special ED devices, a portion of SRAM). A transaction monitor verifies the integrity of the transactions between the Flash memory controller and the overlay RAM. In case of a mismatch due to a fault the error is forwarded to the FCCU.

The user can't inject these faults.

3.6.8 On-chip overlay RAM Address Feedback (Fault #69).

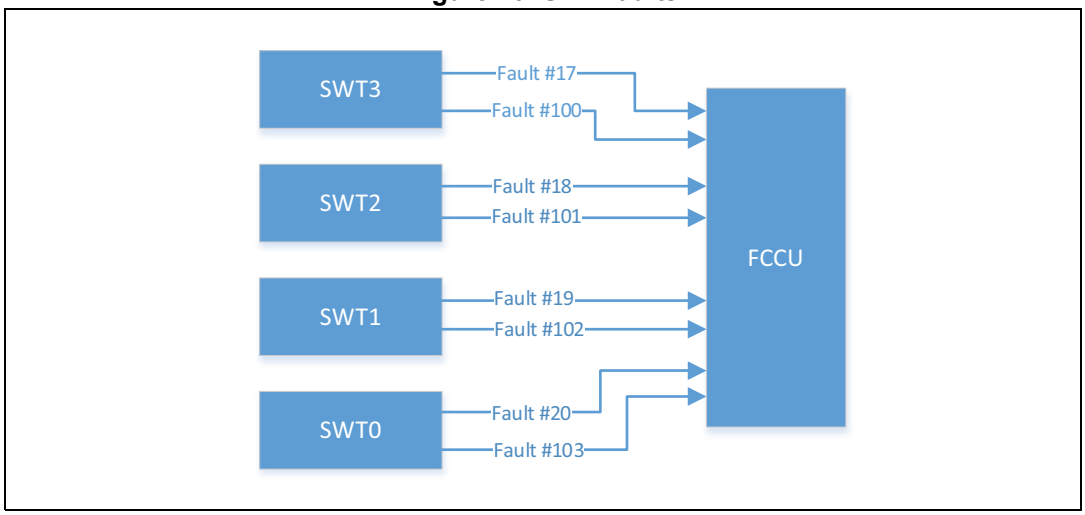
In case the flash controller detected a mismatch compared to the on-chip overlay RAM address feedback mechanism, it forwards this fault to the FCCU. The user can't inject this fault.

o. If an Address encoding error is injected (faults #60 or #66), then address feedback to PFC is incorrect, therefore also PFC raise this faults.

3.7 SWT faults

The SWT is a module that can prevent system lockup in situations such as software getting trapped in an endless loop or a bus transaction which fails to terminate. When enabled, the SWT requires periodic execution of a watchdog servicing operation. The servicing operation resets the timer to a specified time-out period. If the software doesn't service the SWT before the timer expires, the SWT generates an interrupt or a reset according to its configuration.

Figure 10. SWT faults



3.7.1 SWT3 reset request (Fault #17)

If the SWT3 reaches a timeout(p) the SWT3 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT3 and doesn't service it. Note that the user can clear this fault request only by triggering a reset.

3.7.2 SWT2 reset request (Fault #18)

If the SWT2 reaches a timeout(p) the SWT2 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT2 and doesn't service it. Note that the user can clear this fault request only by triggering a reset.

3.7.3 SWT1 reset request (Fault #19)

If the SWT1 reaches a timeout(p) the SWT1 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT1 and doesn't service it. Note that the user can clear this fault request only by triggering a reset.

p. If ITR field of the SWT_CR register is set to zero, the request is sent on the first time-out. If ITR field is set to one, the request is sent on the second consecutive time-out.

3.7.4 SWT0 reset request (Fault #20)

If the SWT0 reaches a timeout[p] the SWT0 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT0 and doesn't service it. Note that the user can clear this fault request only by triggering a reset.

3.7.5 SWT3 first timeout request (Fault #100)

If the SWT3 reaches a timeout[p] [q] the SWT3 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT3 and doesn't service it. User can clear the fault flag by servicing the watchdog and consequent clear.

3.7.6 SWT2 first timeout request (Fault #101)

If the SWT3 reaches a timeout[p] [q] the SWT2 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT2 and doesn't service it. User can clear the fault flag by servicing the watchdog and consequent clear.

3.7.7 SWT1 first timeout request (Fault #102)

If the SWT3 reaches a timeout[p] [q] the SWT1 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT1 and doesn't service it. User can clear the fault flag by servicing the watchdog and consequent clear.

3.7.8 SWT0 first timeout request (Fault #103)

If the SWT3 reaches a timeout[p] [q] the SWT0 forwards this fault to FCCU.

The user can inject this fault by a SW procedure that enables the SWT0 and doesn't service it. User can clear the fault flag by servicing the watchdog and consequent clear.

3.8 MEMU faults

The MEMU is responsible for collecting and reporting error events captured by ECC. When an ECC error occurs, the MEMU receives an error signal and – as consequence - it records the event. The MEMU filters and then forwards notifications in an aggregated manner to the FCCU.

The MEMU does not receive any error signal on ECC events occurring while accessing to the EEPROM sections of the FLASH memory.

Note that to clear an FCCU fault related to the MEMU, the user must clear before the corresponding bit in the MEMU ERR_FLAG register and then resetting the corresponding bit in the FCCU FCCU_RF_Sn register[r].

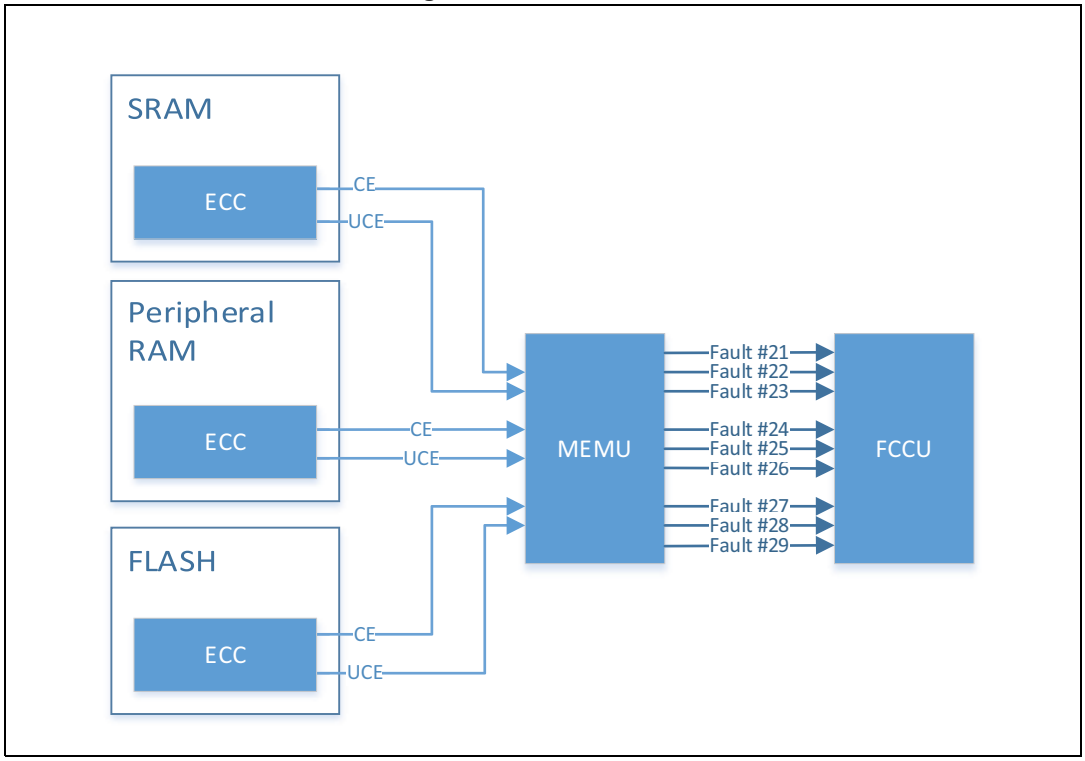
The device provides two mechanisms for accessing any chip memory for the purpose of reading or modifying data, or both, including ECC check bits. This capability is useful for test

q. SWT_CR register must be configured to allow first timeout interrupt only (when SWT_CR.ITR is set).

r. The MEMU remains on during functional reset.

activities, including injecting FCCU faults MEMU related. These mechanisms are the IMA module and the CPU End2End ECC test feature.

Figure 11. MEMU faults



3.8.1 MEMU RAM correctable error (Fault #21)

In case of correctable errors detected by a master of the system, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.2 MEMU RAM uncorrectable error (Fault #22)

In case of uncorrectable errors detected by a master of the system, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.3 MEMU RAM overflow error (Fault #23)

In case of overflow of system RAM error reporting table, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.4 MEMU Peripheral RAM correctable error (Fault #24)

In case of correctable errors in peripheral RAM, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.5 MEMU Peripheral RAM uncorrectable error (Fault #25)

In case of uncorrectable errors in peripheral RAM, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.6 MEMU Peripheral RAM overflow (Fault #26)

In case of overflow of peripheral RAM error reporting table, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.7 MEMU FLASH correctable error (Fault #27)

In case of correctable errors in FLASH, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.8 MEMU FLASH uncorrectable error (Fault #28)

In case of uncorrectable errors in FLASH, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.8.9 MEMU FLASH overflow error (Fault #29)

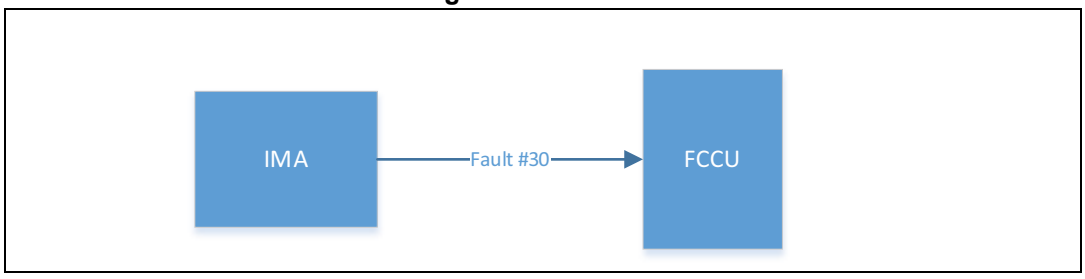
In case of overflow of peripheral FLASH error reporting table, the MEMU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.9 IMA fault

Indirect Memory Access (IMA) refers to the activity of accessing any chip memory for the purpose of reading and/or modifying data and ECC check bits. This capability is useful for test activities, e.g., verifying the integrity of the ECC logic.

Figure 12. IMA fault



3.9.1 IMA SoC Active (Fault #30)

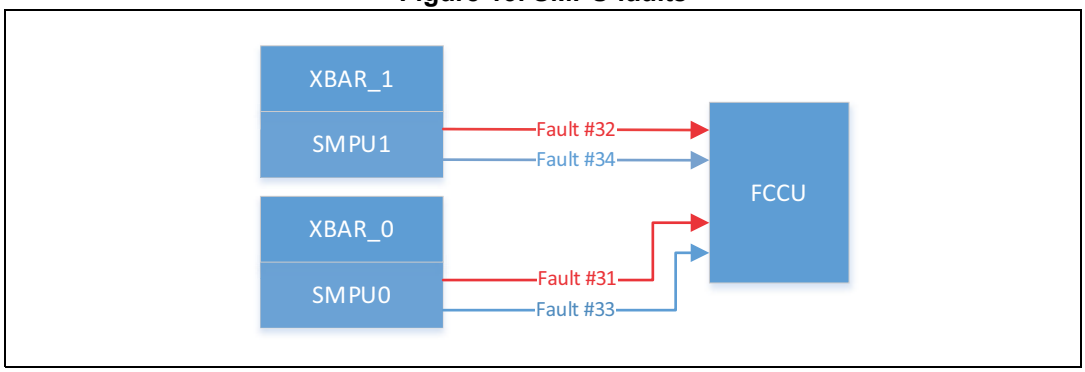
Since unwanted activation of the IMA can interfere with the safety function the FCCU monitors unwanted activation of the IMA. Before any intentional access to the memories by the IMA, the user shall disable the relevant FCCU error input.

The user can inject this fault by a SW procedure using IMA to access to a memory address without disabling the relevant FCCU error input.

3.10 SMPU faults

The SMPU provides hardware access control for system bus resources. The SMPU concurrently monitors and evaluates system bus transactions using pre-programmed region descriptors that define memory spaces and their access rights. Memory accesses that have sufficient access control rights are allowed to complete, while accesses that are not mapped to any region descriptor or have insufficient rights terminates with an access error response.

Figure 13. SMPU faults



3.10.1 SMPU XBAR 0 correctly refuses an access (Fault #33)

In case of an access to a memory location not mapped to any region descriptor or with insufficient rights, it terminates with an access error response. The SMPU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that accesses a memory address not allowed by SMPU configuration.

3.10.2 SMPU XBAR 0 incorrectly refuses an access (Fault #31)

The SMPU include a hardware monitors that detects the following unexpected behavior:

the SPMU allows an access to a memory location not mapped to any descriptor or with insufficient rights, and

the SPMU doesn't allow a memory access with sufficient rights.

If this hardware monitor detects a fault, it forwards this event to the FCCU.

The user can't inject this fault.

3.10.3 SMPU XBAR 1 correctly refuses an access (Fault #34)

In case of an access to a memory location not mapped to any region descriptor or with insufficient rights, it terminates with an access error response. The SMPU forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that accesses a memory address not allowed by SMPU configuration.

3.10.4 SMPU XBAR 1 incorrectly refuses an access (Fault #32)

The SMPU include a hardware monitor that detects the following unexpected behaviors:

- the SPMU allows an access to a memory location not mapped to any descriptor or with insufficient rights

  and

- the SPMU doesn't allow a memory access with sufficient rights.

If this hardware monitor detects a fault, it forwards this event to the FCCU.

The user can't inject this fault.

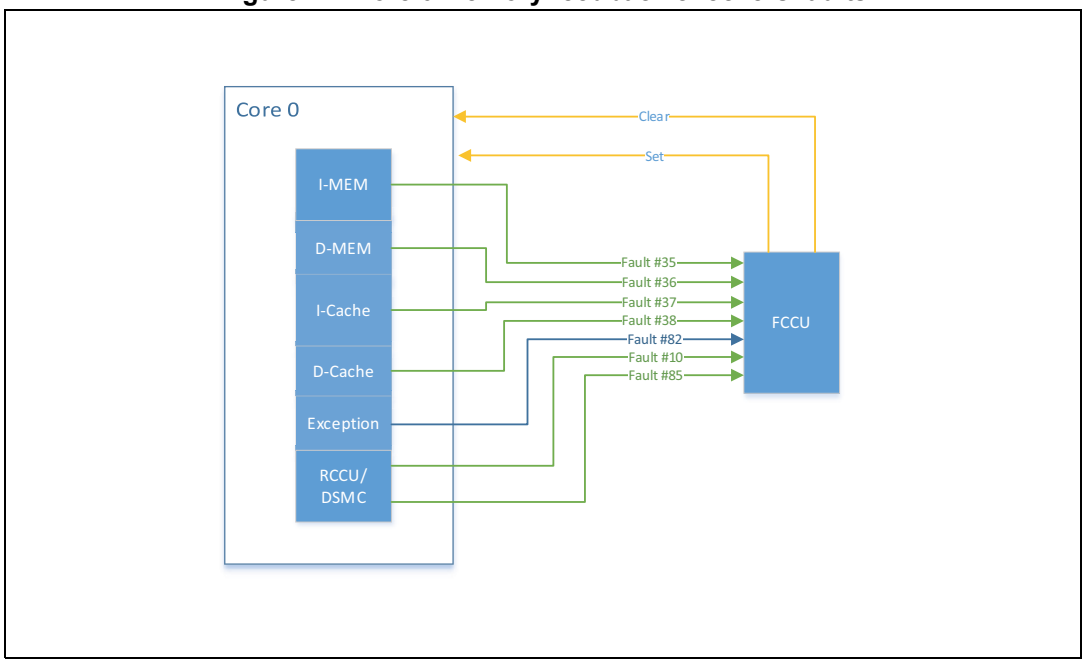
3.11 CORE 0 faults

Memory feedback checkers inside the core 0 are able to detect failures affecting the control logic of the local memory.

The control signals, which the memory controller sends to the memory array for any operation, are latched and sent back from the memory array to the memory controller. The memory controller compares these received control signals with the transmitted control signals. If they do not match, it forwards an error event to the FCCU. There is also RCCU logic which takes care of lockstep operation if they are enabled by the user. If the logic detects a synchronization issue, it forwards the error event to the FCCU.

Besides, the FCCU monitor the activation of the replicated core (i.e., LSM).

Figure 14. Core 0 Memory feedback checkers faults



3.11.1 Lockstep out of sync in the Core or associated DSMC (Fault #10)

In case the operation of the safety core and its replica are out of sync due to a fault, the RCCU signals the fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.11.2 I-MEM Feedback checker error (Fault #35)

In case control signals latched back to the memory controller don't match the ones originally sent to I-MEM, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.11.3 D-MEM feedback checker error (Fault #36)

In case control signals latched back to the memory controller don't match the ones originally sent to D-MEM, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.11.4 I-CACHE feedback checker error (Fault #37)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.11.5 D-CACHE feedback checker error (Fault #38)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.11.6 Core 0 Machine check exception indication (Fault #82)

When the core runs into the machine check condition, the hardware signals this fault to the FCCU. The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block containing uncorrectable errors.

3.11.7 Indication of disablement of lockstep mode (Fault #85)

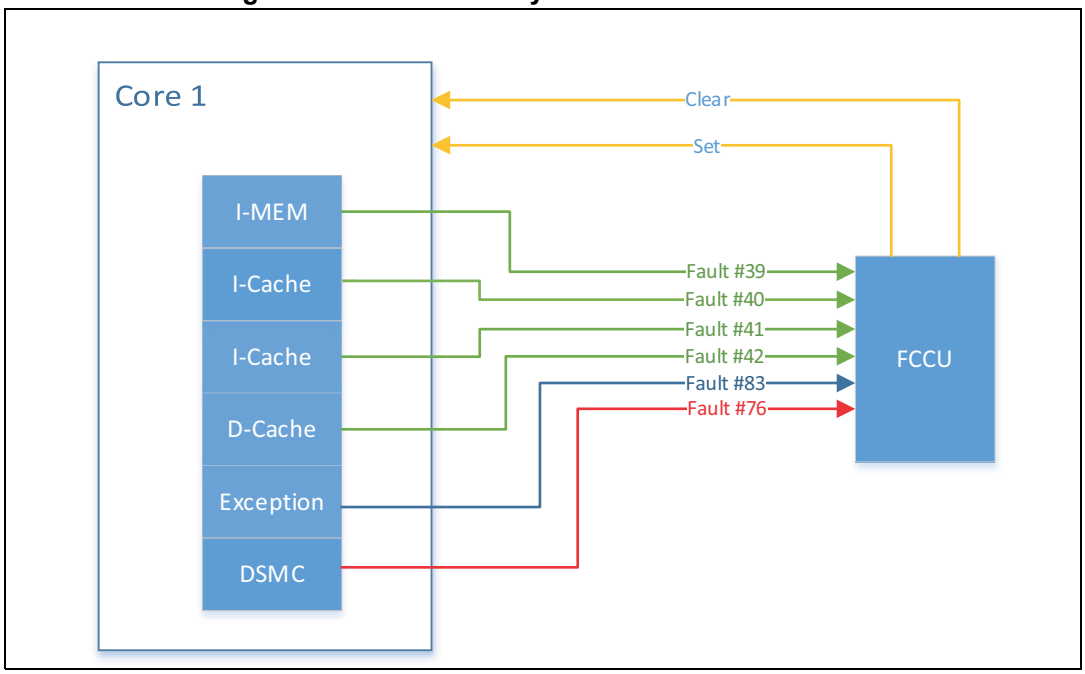
In case of disablement of the LSM on the safety core, the event is forwarded to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12 CORE 1 faults

Memory feedback checkers inside the core 1 are able detect failures affecting the control logic of the local memory.

The control signals, which the memory controller sends to the memory array for any operation, are latched and sent back from the memory array to the memory controller. The memory controller compares these received control signals with the transmitted control signals. If they do not match, it forwards an error event to the FCCU.

Figure 15. Core 1 Memory feedback checkers faults



3.12.1 I-MEM Feedback checker error (Fault #39)

In case control signals latched back to the memory controller don't match the ones originally sent to I-MEM, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.2 D-MEM feedback checker error (Fault #40)

In case control signals latched back to the memory controller don't match the ones originally sent to D-MEM, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.3 I-CACHE feedback checker error (Fault #41)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.4 D-CACHE feedback checker error (Fault #42)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.5 Core 1 Machine check exception indication (Fault #83)

When the core runs into the machine check condition, the hardware forwards this fault to the FCCU. The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block containing uncorrectable errors.

3.12.6 (SPC58xN only) Core 1 DSMC control signal error (Fault #76)

DSMC generates atomic read-modify-write bus transactions to the attached slave memory controller. In a case of error in DMSC, the hardware forwards this fault to the FCCU.

The user can't inject this fault.

3.13 CORE 2 faults

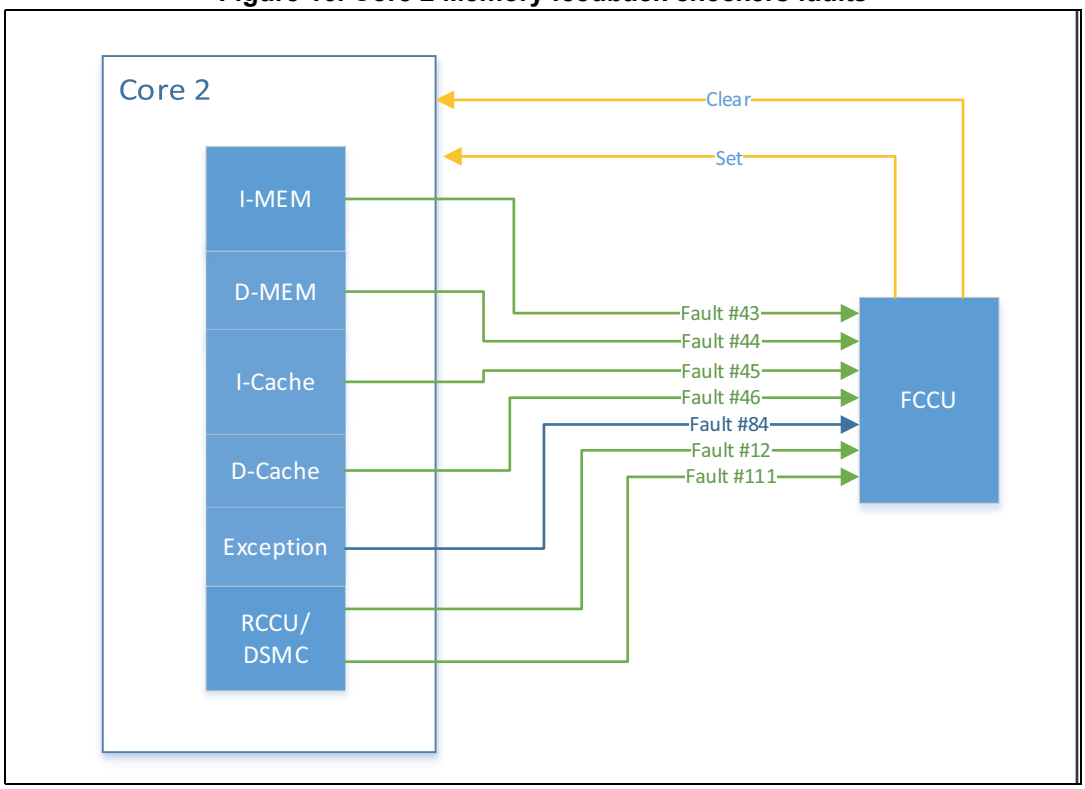
Memory feedback checkers inside the core 2 are able detect failures affecting the control logic of the memory.

The control signals, which the memory controller sends to the memory array for any operation, are latched and sent back from the memory array to the memory controller. The memory controller compares these received control signals with the transmitted control signals. If they do not match, it forwards an error event to the FCCU.

There is also RCCU logic which takes care of lockstep operation if they enabled by user. If the logic detects a synchronization issue, it forwards the error event to the FCCU.

Besides, the FCCU monitor the activation of the replicated core (i.e., LSM).

Figure 16. Core 2 Memory feedback checkers faults



3.13.1 Lockstep out of sync in the Core or associated DSMC (Fault #12)

In case the operation of the safety core and its replica are out of sync due to a fault, the RCCU signals the fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.2 I-MEM feedback checker error (Fault #43)

In case control signals latched back to the memory controller don't match the ones originally sent to I-MEM, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.3 D-MEM feedback checker error (Fault #44)

In case control signals latched back to the memory controller don't match the ones originally sent to D-MEM, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.4 I-CACHE feedback checker error (Fault #45)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.5 D-CACHE feedback checker error (Fault #46)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.6 Core 2 Machine check exception indication (Fault #84)

When the core runs into the machine check condition, the hardware forwards this fault to the FCCU. The user can inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block containing uncorrectable errors.

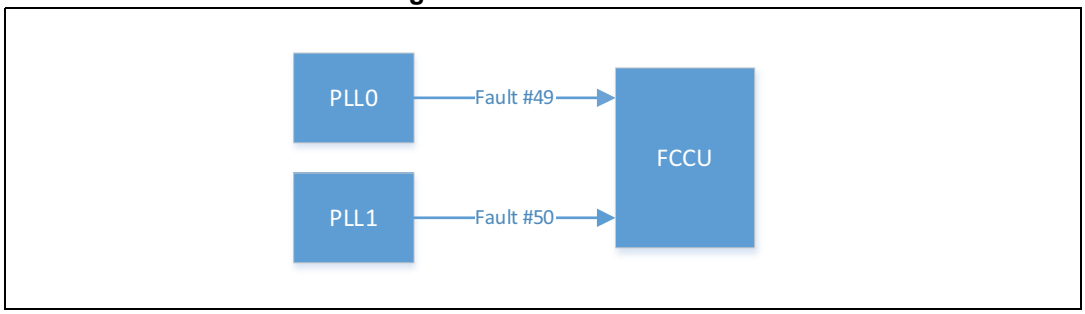
3.13.7 Indication of disablement of lockstep mode (Fault #111)

In case of disablement of the LSM on the safety core, the event is forwarded to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.14 PLLDIG faults

The SPC58xEx/SPC58xNx/xGx embeds a dual PLL system which provides separate system and peripheral clocks.

Figure 17. PLL DIG faults



3.14.1 PLL0/1 Loss of lock (Fault #49 Fault #50)

A built-in mechanism can detect a loss of lock for PLL. The PLLDIG forwards this fault to the FCCU. The user can inject this fault by a SW procedure that changes on-the-fly the PLL configuration generating a loss of lock.

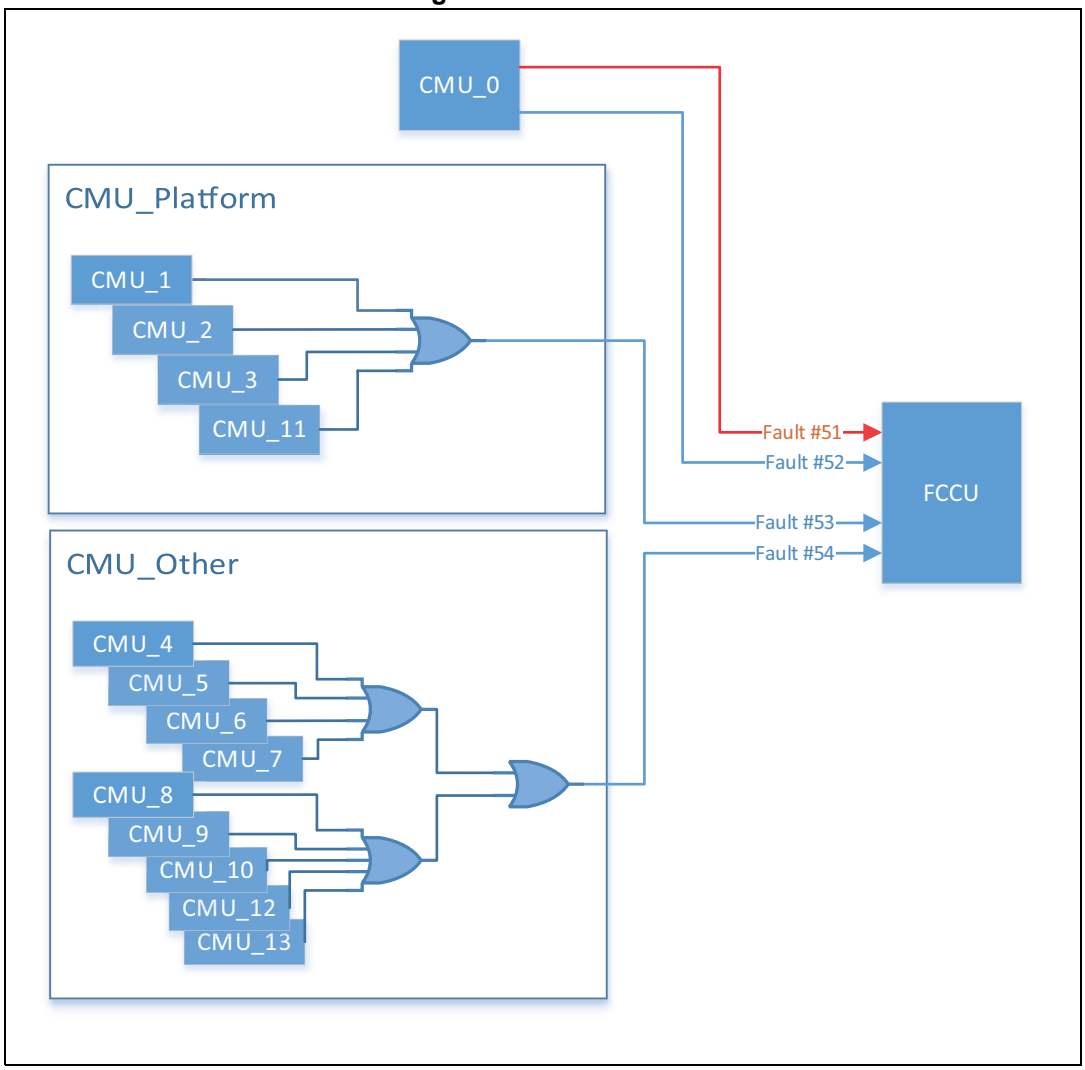
3.15 CMU faults

Different CMU modules supervise the integrity of the clock sources of the device. If the monitored clock frequency is less than the reference frequency, or it violates an upper or lower frequency boundary, the CMU detects and forwards[s] this event to the FCCU. Refer to device SPC58xEx/SPC58xNx/xGx Microcontroller Reference Manual for further details on CMU configuration and which clock is monitored[t].

s. The user must enable the CMU that is disabled by default.

t. Refer to the section “Clock input sources” of the reference manual.

Figure 18. CMU faults



3.15.1 XOSC less than IRC, XTAL Pin Floating, EXTAL Pin Floating (Fault #51)

The CMU_0 monitors the XOSC frequency. If the XOSC frequency is less than a configurable value or it is not stable (e.g., the XOSC is not connected), the CMU_0 can detect this event and forwards it to the FCCU.

The user can't inject this fault[u].

3.15.2 System clock frequency out of range (Fault #52)

The CMU_0 monitors the system clock frequency (PHI output of PLL_0) using the IRCOSC and XOSC frequencies as monitor references. If the PLL_0 frequency is above or below the monitoring thresholds, the CMU_0 can detect this event and forward it to the FCCU. The

u. The CMU0 checks if the Frequency of the XOSC is less than the frequency of the IRC divided by a configurable factor within a range from 1 to 8. If the frequency of the XOSC is higher than the frequency of the IRC it is not possible inject the fault by a SW procedure

user can inject this fault by a SW procedure that sets misconfigured values for the monitoring thresholds.

3.15.3 Platform clocks frequency out of range (Fault #53)

The CMU_1 monitors the clock frequency used by COREs, HSM and XBAR. The CMU_2 monitors the clock frequency used by DMA, SIPI, GTM and INTC. The CMU_3 monitors the clock frequency used by all peripheral bridges and peripherals not connected to any other system clock divider. The CMU_11 monitors the clock frequency used by SWT, STM and the SEM4. If one of these frequencies is above or below the monitoring thresholds, the relevant CMU can detect this event and forwards it to the FCCU[v]. The user can inject this fault by a SW procedure that sets misconfigured values for the monitoring thresholds.

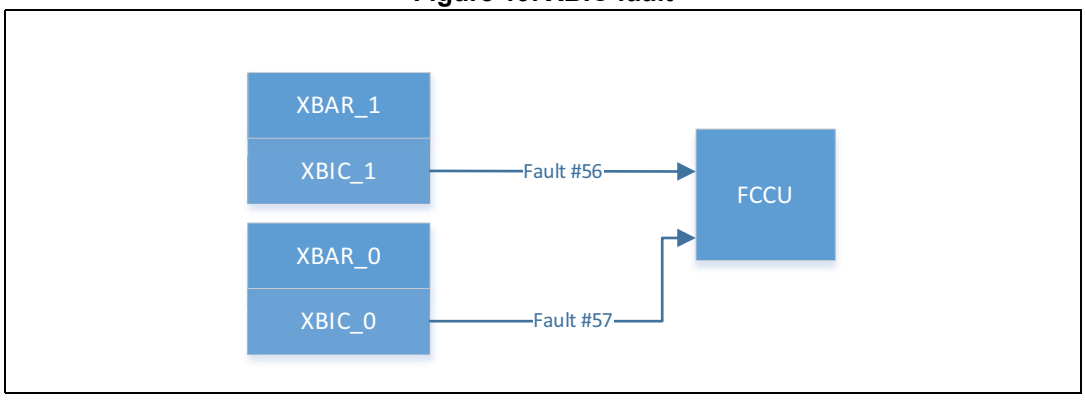
3.15.4 Other clocks frequency out of range (Fault #54)

The CMU_4 monitors the clock frequency used by the GTM module, the CMU_5 monitors the clock frequency used by the SDADC module, the CMU_6 monitors the clock frequency used by the SARADC, the CMU_7 monitors the clock frequency sent to the SENT module, the CMU_8, CMU_9, CMU_10 monitor the clocks frequencies used by the PSI5 module, the CMU_12 (only for SPC58xE/G) monitors (the clock frequency used by EMIOS and the CMU_13 monitors the clock frequency used by the CAN module. If one of these frequencies is above or below the monitoring thresholds, the relevant CMU can detect this event and forwards it to the FCCU[v]. The user can inject this fault by a SW procedure that sets misconfigured values for the monitoring thresholds.

3.16 XBIC fault

The XBIC verifies the integrity of each crossbar transfer. The crossbar transfer attributes information for all master and slave ports is routed to the XBIC that verify the coherence between input and output signals of the crossbar.

Figure 19. XBIC fault



v. The FCCU receives the OR'ed signal from these CMU modules.

3.16.1 XBAR 0/1 XBIC fault (Fault#57, Fault #56)

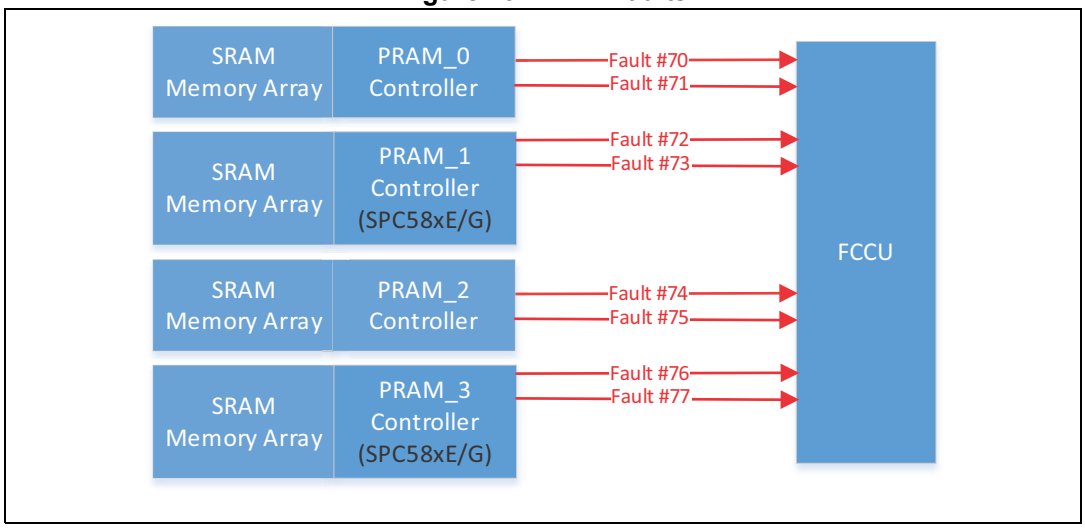
In case of corrupted transaction through the XBAR, the XBIC detects this event and forwards it to the FCCU.

The user can inject this fault by a SW procedure that uses the XBIC error injection feature through the XBIC_EIR register.

3.17 PRAM faults

The PRAM controller is the interface between the system bus and the system RAM. It converts the protocols between the system bus and the RAM array interface. The device embeds either two (SPC58xNx) or four (SPC58xEx, SPC58xGx) controllers. Refer to device SPC58xEx/SPC58xNx/xGx Microcontroller Reference Manual for further details on PRAM controllers.

Figure 20. PRAM faults



3.17.1 System RAM Address Feedback or RAM Late-Write Buffer mismatch PRAM0 (Fault #70)

In case of addressing error[w] the PRAM controller is able to detect this event and it forwards this fault to the FCCU. The user can't inject this fault.

3.17.2 EDC after ECC for RAM (read-modify-write ECC) PRAM0 (Fault #71)

The EDC after ECC detects a random failure affecting the ECC logic of the PRAM controller and forwards this event to the FCCU. The user can't inject this fault.

w. E.g. a write error in the RAM controller resulting in corrupted RAM access.

3.17.3 System RAM Address Feedback or RAM Late-Write Buffer mismatch PRAM2 (Fault #74)

In case of addressing error(w) the PRAM controller is able to detect this event and it forwards this fault to the FCCU. The user can't inject this fault.

3.17.4 EDC after ECC for RAM (read-modify-write ECC) PRAM2 (Fault #75)

The EDC after ECC detects a random failure affecting the ECC logic of the PRAM controller and forwards this event to the FCCU. The user can't inject this fault.

3.17.5 (SPC58xE/G only) System RAM Address Feedback or RAM Late-Write Buffer mismatch PRAM1 (Fault #72)

In case of addressing error(w) the PRAM controller is able to detect this event and it forwards this fault to the FCCU. The user can't inject this fault.

3.17.6 (SPC58xE/G only) EDC after ECC for RAM (read-modify-write ECC) PRAM1 (Fault #73)

The EDC after ECC detects a random failure affecting the ECC logic of the PRAM controller and forwards this event to the FCCU. The user can't inject this fault.

3.17.7 (SPC58xE/G only) System RAM Address Feedback or RAM Late-Write Buffer mismatch PRAM3 (Fault #76)

In case of addressing error(w) the PRAM controller is able to detect this event and it forwards this fault to the FCCU. The user can't inject this fault.

3.17.8 (SPC58xE/G only) EDC after ECC for RAM (read-modify-write ECC) PRAM3 (Fault #77)

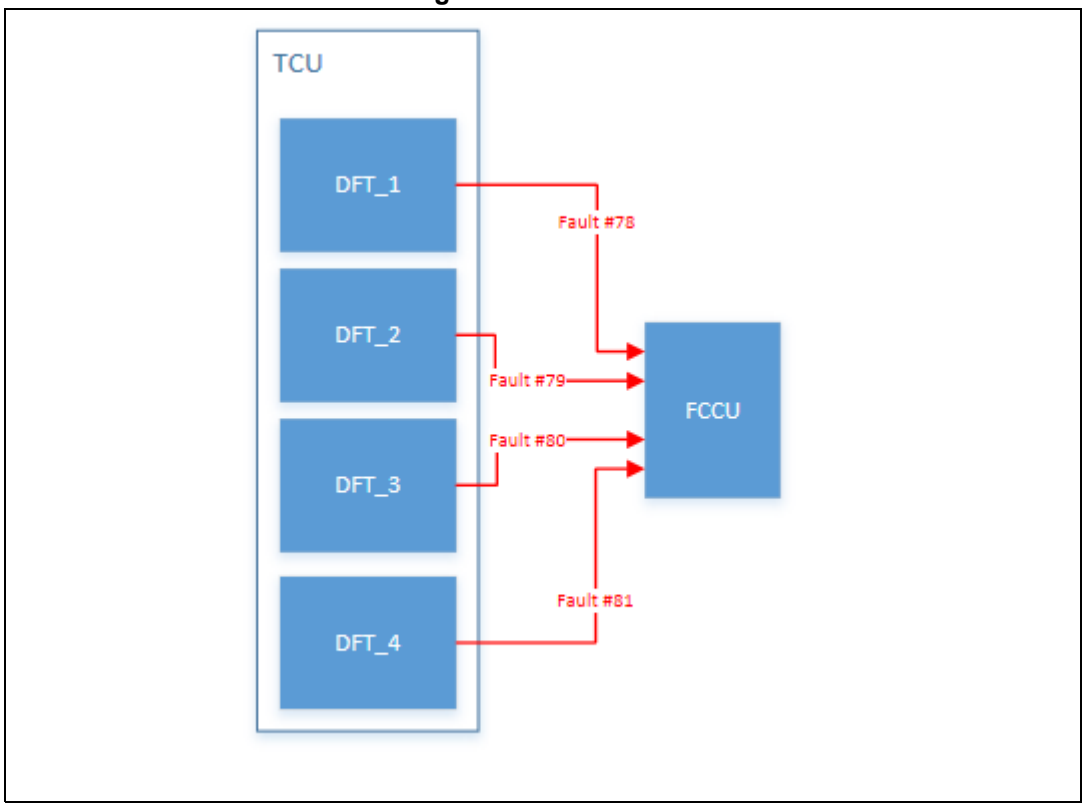
The EDC after ECC detects a random failure affecting the ECC logic of the PRAM controller and forwards this event to the FCCU. The user can't inject this fault.

3.18 TCU faults

The device monitors the signals that are part of the TCU. These signals can move the device into TEST mode. Test mode, however, must not be enabled while the safety application runs. For this reason, if a random event asserts one of these signals the event is detected and forwarded to the FCCU.

A different FCCU channel monitors each Diagnostic Function Test domain.

Figure 21. TCU faults



3.18.1 Test circuitry Group 1 activation (Fault #78)

In case of unwanted activation of the test circuitry in the related diagnostic function test domain, the event is detected and forwarded to the FCCU. The user can't inject this fault.

3.18.2 Test circuitry Group 2 activation (Fault #79)

In case of unwanted activation of the test circuitry in the related diagnostic function test domain, the event is detected and forwarded to the FCCU. The user can't inject this fault.

3.18.3 Test circuitry Group 3 activation (Fault #80)

In case of unwanted activation of the test circuitry in the related diagnostic function test domain, the event is detected and forwarded to the FCCU. The user can't inject this fault.

3.18.4 Test circuitry Group 4 activation (Fault #81)

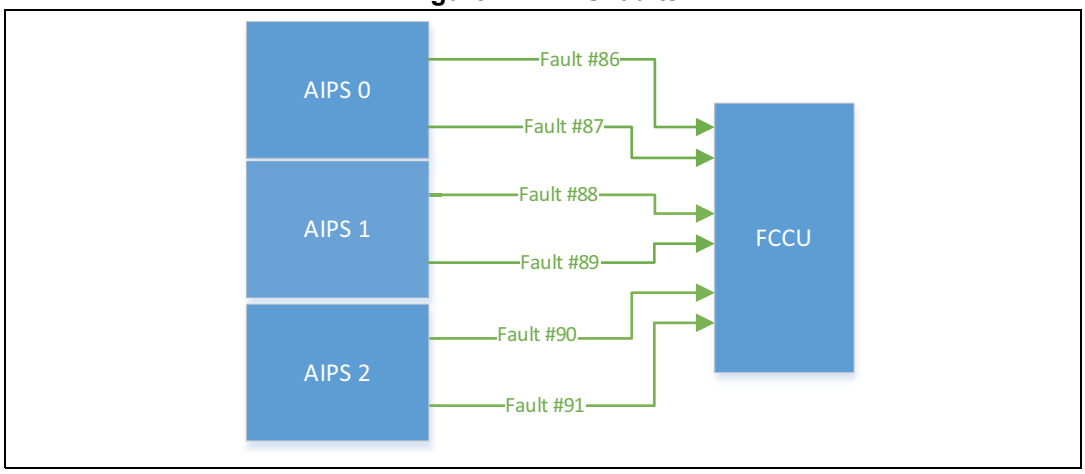
In case of unwanted activation of the test circuitry in the related diagnostic function test domain, the event is detected and forwarded to the FCCU. The user can't inject this fault.

3.19 AIPS faults

The peripheral bridge connects the crossbar switch interface to the register interface of the peripherals embedded within the device. The device incorporates three peripheral bridges: AIPS_h0, AIPS_1 and AIPS_2.

The Gasket is an interface between the crossbar and the AIPS. It adapts the frequency of domains with fast and slow clocks. In addition, the hardware embeds a *gasket monitor* that detects failures affecting the gasket. The AIPS are also protected from random failure by a lockstep replica that monitors the control signals (i.e., AIC).

Figure 22. AIPS faults



3.19.1 AIC or gasket monitor error AIPS_0 (Fault #86)

In case of hardware fault results in a wrong frequency conversion between the XBAR and the AIPS or corrupted transaction detected by the AIC, the hardware forwards the event to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.19.2 EDC after ECC error on a transaction through the AIPS_0 (Fault #87)

A random failure affecting the ECC correction logic can cause a corrupted ECC correction. The EDC after ECC can detects this event and forward it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.19.3 AIC or gasket monitor error AIPS_1 (Fault #88)

In case of hardware fault results in a wrong frequency conversion between the XBAR and the AIPS or corrupted transaction detected by the AIC, the hardware forwards the event to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.19.4 EDC after ECC error on a transaction through the AIPS_1 (Fault #89)

A random failure affecting the ECC correction logic can cause a corrupted ECC correction. The EDC after ECC can detects this event and forward it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.19.5 AIC or gasket monitor error AIPS_2 (Fault #90)

In case of hardware fault results in a wrong frequency conversion between the XBAR and the AIPS or corrupted transaction detected by the AIC, the hardware forwards the event to the FCCU. The user can inject this fault by the FCCU fake fault interface.

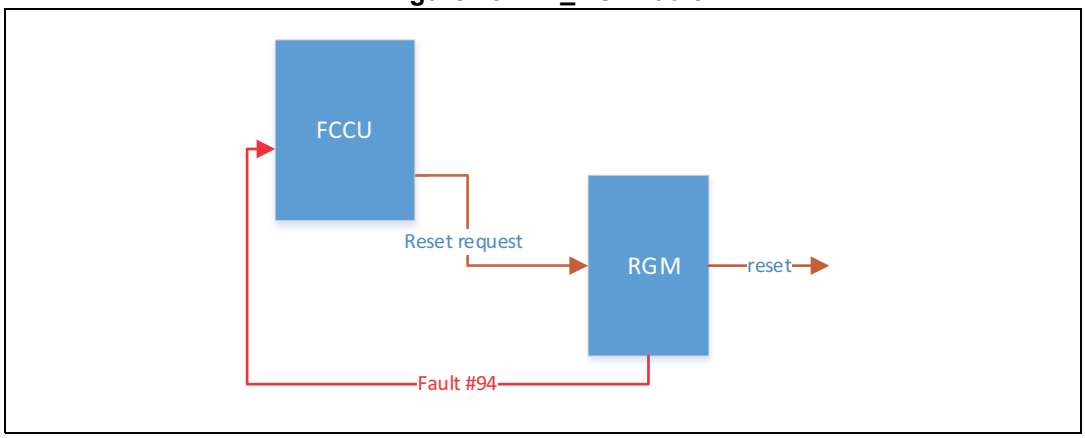
3.19.6 EDC after ECC error on a transaction through the AIPS_2 (Fault #91)

A random failure affecting the ECC correction logic can cause a corrupted ECC correction. The EDC after ECC can detects this event and forward it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.20 MC_RGM fault

The MC_RGM centralizes the different reset sources and manages the reset sequence of the chip.

Figure 23. MC_RGM fault



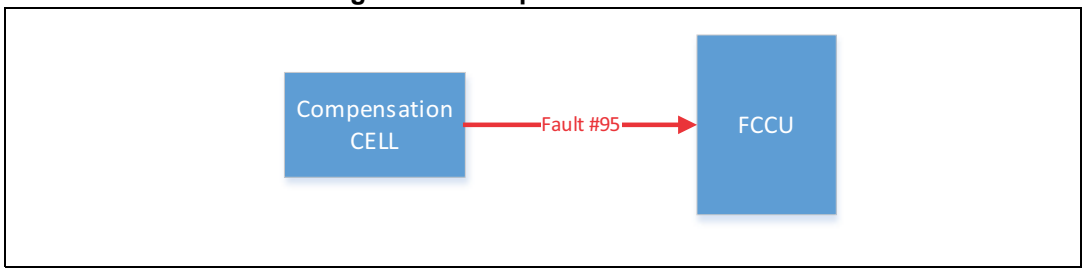
3.20.1 Safe Mode Entry Indication (Fault #94)

The MC_RGM can request a transition to SAFE mode to the MC_ME. In a case of an unwanted safe mode request due to a random event, the hardware detected and forwards this event to the FCCU. The user can't inject this fault.

3.21 Compensation cells fault

Compensation cells generate 8-bit compensation code for IO buffers. Compensation reduces the spread of some circuit parameters(x) in the IO buffers over temperature, pressure and voltage.

Figure 24. Compensation cell fault



x. Some of these parameters are the slew rate of the output signal and the output impedance.

3.21.1 Pad Compensation Disabled (Fault #95)

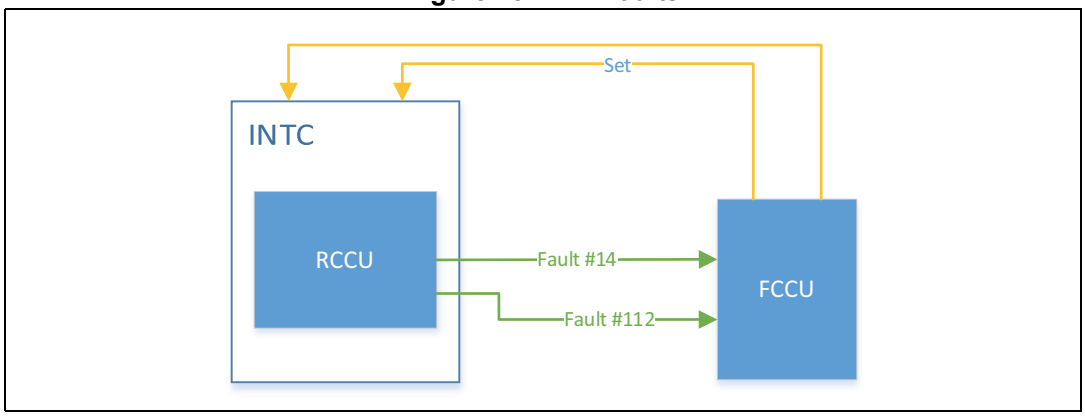
When the compensation cell is in normal mode, it generates the compensation codes. If it exits the normal mode, the hardware detects this event and forwards it to the FCCU. The user can't inject this fault.

3.22 Interrupt Controller (INTC) faults

The INTC provides a mechanism to schedule and service the interrupt requests external to the cores.

The INTC is duplicated and the operation are executed in lockstep mode. The replicated operations are compared and any operational deviation between the supervised signals notifies the FCCU of the discrepancy.

Figure 25. INTC faults



3.22.1 INTC Lockstep out of sync due to a fault (Fault #14)

In case a deviation in the operation of the INTC and its replica, the RCCU signals the fault to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

3.22.2 Indication of disablement of INTC lockstep mode (Fault #112)

In case of disablement of the LSM in the INTC, the event is forwarded to the FCCU.

The user can inject this fault by the FCCU fake fault interface.

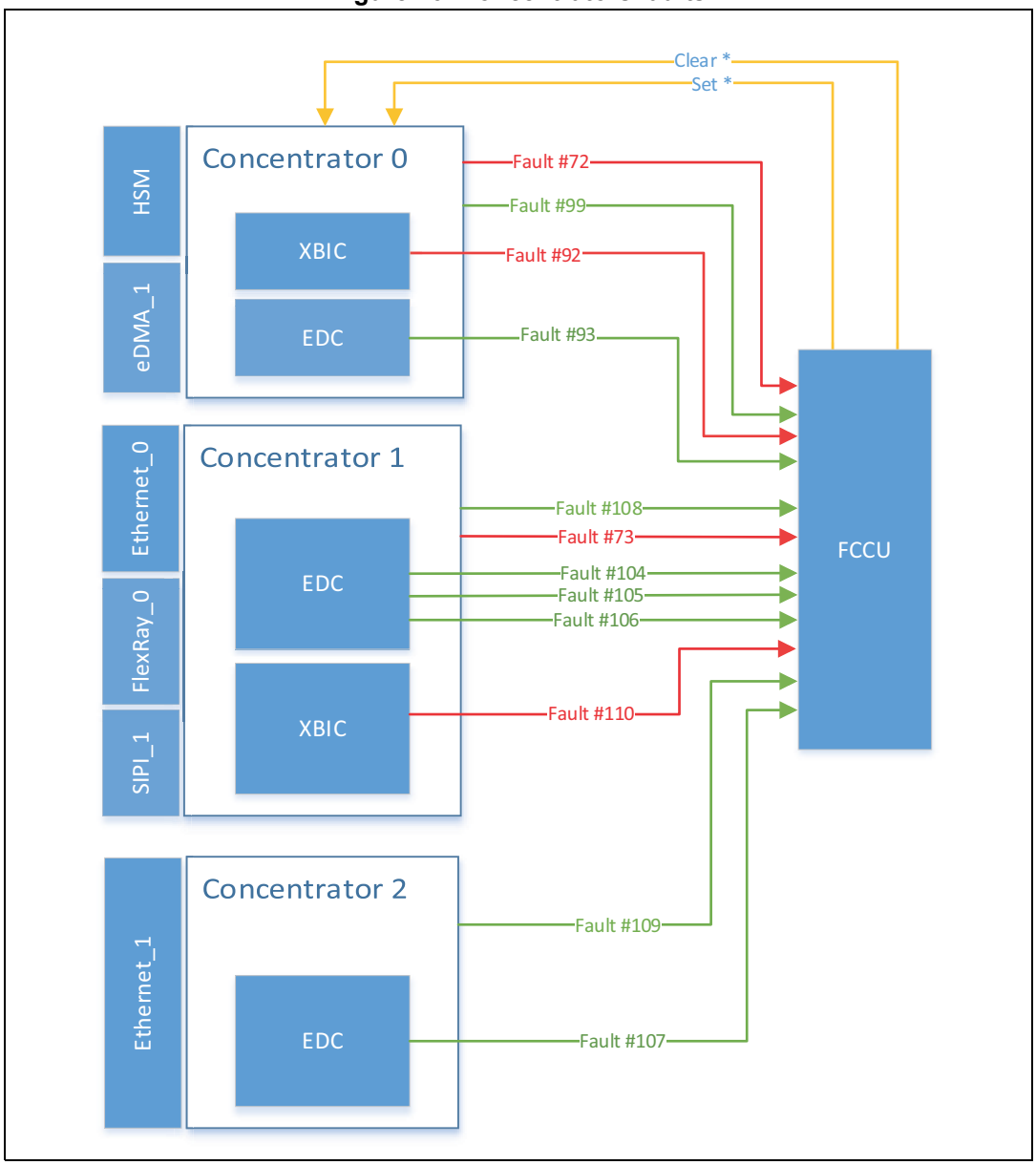
3.23 Concentrator faults

There are integrated two different concentrators on SPC58xNx and three of them on SPC58xEx/xGx.

They forward and handle data and control signals between crossbar and some master devices. The concentrator 0 is linked to the HSM and eDMA_1, concentrator 1 handles Ethernet_0, Flexray_0 and SIPI_1. Concentrator 2 handles Ethernet_1 and is implemented only on SPC58xEx/xGx.

Each concentrator embeds internal frequency gaskets which interface domains with different clock frequencies. In addition, the hardware embeds a *gasket monitor* that detects failures affecting the gasket and ECC logic for the e2eECC that is monitored by a EDC after ECC mechanism.

Figure 26. Concentrators faults



3.23.1 Concentrator 0/1 XBIC errors (Faults #92, #110)

Concentrators 0 and 1 embeds an internal XBAR along with an XBIC. The XBIC checks the integrity of signals that are routed through this XBAR. In case of error detected by or on internal XBIC of the given concentrator it forwards a fault to the FCCU.

The user can't inject these faults.

3.23.2 Concentrator 0 EDC errors Faults #93

The Concentrator 0 embeds the EDC after ECC logic to implement the e2eECC. In a case of a hardware fault affecting the ECC logic and resulting in a corrupted ECC correction, the EDC after ECC logic detects this event and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.23.3 Concentrator 0 gasket monitor errors (SPC58xN only) Fault #72

In a case of error in AHB control signals and the response signals that transit through concentrator port, the hardware forwards this fault to the FCCU.

The user can’t inject the fault

3.23.4 Concentrator 0 gasket frequency errors Fault #99

In case of hardware fault results in a wrong frequency conversion between the XBAR and the concentrator, the hardware forwards this fault to the FCCU.

The user can inject the fault by the FCCU fake fault interface.

3.23.5 Concentrator 1 monitor errors (SPC58xN only) Fault #73

In a case of error in AHB control signals and the response signals that transit through concentrator port, the hardware forwards this fault to the FCCU.

The user can’t inject the fault

3.23.6 Concentrator 1 EDC errors Faults #104

The Concentrator 1 embeds the EDC after ECC logic for each master connected to the concentrator to implement the e2eECC. In a case of a hardware fault affecting the ECC logic dedicated to the SIPI1 resulting in a corrupted ECC correction, the EDC after ECC logic detects this event and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.23.7 Concentrator 1 EDC errors Faults #105

The Concentrator 1 embeds the EDC after ECC logic for each master connected to the concentrator to implement the e2eECC. In a case of a hardware fault affecting the ECC logic dedicated to the Flexray 0 resulting in a corrupted ECC correction, the EDC after ECC logic detects this event and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.23.8 Concentrator 1 EDC errors Faults #106

The Concentrator 1 embeds the EDC after ECC logic for each master connected to the concentrator to implement the e2eECC. In a case of a hardware fault affecting the ECC logic dedicated to the Ethernet 0 resulting in a corrupted ECC correction, the EDC after ECC logic detects this event and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.23.9 Concentrator 1 gasket frequency errors Fault #108

In case of hardware fault results in a wrong frequency conversion between the XBAR and the concentrator, the hardware forwards this fault to the FCCU.

The user can inject the fault by the FCCU fake fault interface.

3.23.10 Concentrator 2 EDC errors (SPC58xE/G only) Faults #107

The Concentrator 1 embeds the EDC after ECC logic for each master connected to the concentrator to implement the e2eECC. In a case of a hardware fault affecting the ECC logic dedicated to the Ethernet 1 resulting in a corrupted ECC correction, the EDC after ECC logic detects this event and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.23.11 Concentrator 2 gasket frequency errors (SPC58xE/G only) Fault #109

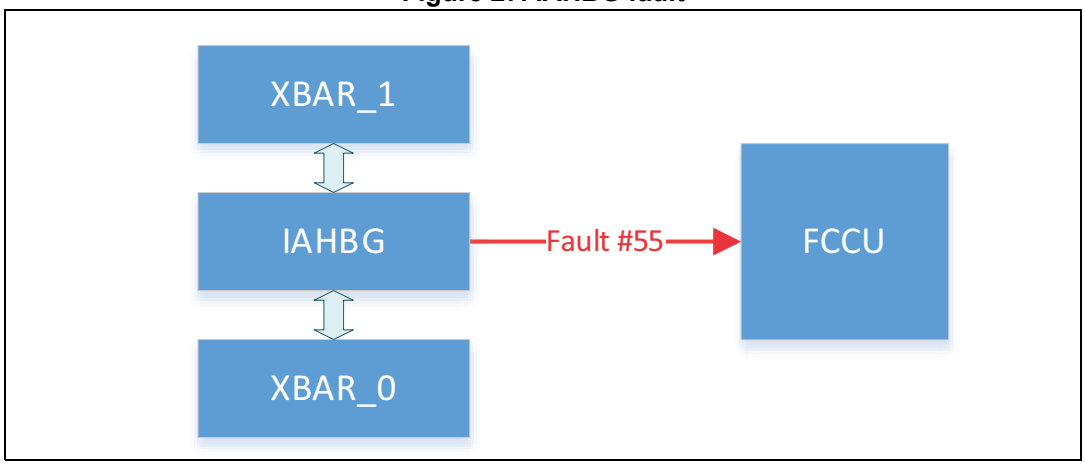
In case of hardware fault results in a wrong frequency conversion between the XBAR and the concentrator, the hardware forwards this fault to the FCCU.

The user can inject the fault by the FCCU fake fault interface.

3.24 Intelligent AHB Gasket (IAHBG) faults

The IAHBG modules do intelligent bridging between different clock domains.

Figure 27. IAHBG fault



3.24.1 Frequency error between XBAR 0 and the XBAR 1 (Fault #55)

For all the frequency error faults above, the IAHBG signals to the FCCU whether a hardware fault results in a bad frequency conversion between the different domains.

The user can't inject these faults.

4 Example code

An example code that includes the FCCU settings and how to inject FCCU faults according to the above list is available upon request. This is the summary of the actions done in the example code:

- Initialize the MCU (clocks and modules);
  - Reset the RGM and clear its registers
  - Initialize the FCCU in the way that for all the testable faults:

    They are enabled

    They are SW recoverable

    No reset action

    Interrupt is enabled

    Output pins are enabled

    FCCU state machine goes to Alarm state in case of fault
- For each faults identified as "Testable" the software:
  - Verify the FCCU status before injection

    If in Normal state, proceed

    If in Alarm or Fault state, stop
  - Inject it (using the monitor's registers or using fake fault injection or a SW procedure, if possible)
  - Verify the FCCU status after injection

    If in Alarm state, proceed

    If in Normal or Fault state, stop
  - Clear the monitor and FCCU Alarm state
  - Verify the FCCU status after recovering from Alarm

    If in Normal state, proceed

    If in Alarm or Fault state, stop
- Check FCCU reaction (IRQ, Reset Request, EOUT)

5 Summary

Safety analysis requires that the user verifies the integrity of the error reaction path periodically with a period of a trip time (i.e. 10h). The user shall verify the connection between monitors and the FCCU[y]. The methodology for these tests depend on the specific FCCU input. The idea, however, is to inject a fault and verifies whether the FCCU correctly receives and react[z] to it.

This document - with reference to SPC58xEx/SPC58xNx/xGx devices - describes the FCCU faults inputs and how to verify their reaction path.

y. Not all FCCU inputs are testable.

z. The user doesn't need to check the FCCU reaction for all testable fault. It's enough to test the FCCU reaction only for a single fault.

6 Revision history

Table 2. Document revision history

Date	Revision	Changes
18-Feb-2019	1	Initial release.