AN5013
Application note

SPC584Cx/SPC58ECx 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 SPC584Cx/SPC58ECx (40 nm - Body - ASIL B). Most of the concepts, however, are also valid for the other devices belonging to the 40nm and 55nm 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 SPC584Cx Microcontroller Reference Manual for further details on this module (see *Section A.1: Document reference*).

A reference code is available.

Contents

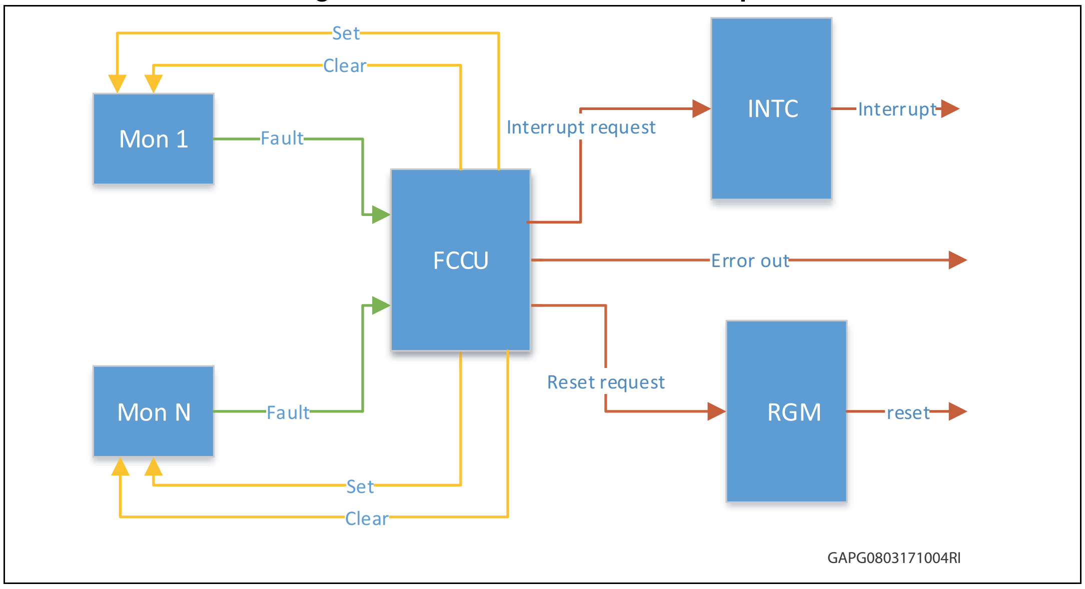
List of tables

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 path



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

FCCU Inner diagram shows how the FCCU is connected to the other blocks. The reader shall consider this figure – and all other figures in this document – as a logic schema 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. The user shall run this software test 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 previous requirement. Note that the user can't test the error reaction path for certain monitors.

Table 1 lists and describes all FCCU input fault sources for SPC584Cx/SPC58ECx.

a. The safety manual defines the 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	Voltage out of range from LVDs
2	PMC DIG	Voltage out of range from HVDs
3	PMC DIG	DCF Safety Error
4	PMC DIG	Voltage detector BIST
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	STCU Fault in case MBIST control signals go to wrong condition during user application
9	GLUE LOGIC	JTAG or NPC not in reset or activation of dangerous debug functionality. Spurious activation of SSCM
15	DMA1	TCD EDC after ECC error
16	FLASH	FLASH ECC errors
17	SWT	Software watchdog timer 3 Reset Request
18	SWT	Software watchdog timer 2 Reset Request
20	SWT	Software watchdog timer 0 Reset Request
21	MEMU	MEMU RAM correctable error
22	MEMU	MEMU RAM uncorrectable error
23	MEMU	MEMU RAM overflow error
24	MEMU	MEMU Peripheral correctable error
25	MEMU	MEMU Peripheral uncorrectable error
26	MEMU	MEMU Peripheral overflow error
27	MEMU	MEMU FLASH correctable error
28	MEMU	MEMU FLASH uncorrectable error
29	MEMU	MEMU FLASH overflow error
30	IMA	IMA SoC Active
32	SMPU	SMPU XBAR 1 Monitor incorrectly refuses an access
34	SMPU	SMPU XBAR 1 Monitor correctly refuses an access
35	CORE 0	D-MEM Bank 0 feedback checker error
36	CORE 0	D-MEM bank 1 feedback checker error
37	CORE 0	I-CACHE bank 0 feedback checker error
38	CORE 0	I-CACHE bank 1 feedback checker error
39	CORE 0	I-CACHE bank 2 feedback checker error
40	CORE 0	I-CACHE bank 3 feedback checker error

Table 1. List of faults (continued)

Channel #	Source	Description
41	CORE 0	I-CACHE Tag RAM error
42	CORE 0	D-CACHE bank 0 feedback checker error
43	CORE 0	D-CACHE bank 1 feedback checker error
44	CORE 0	D-CACHE bank 2 feedback checker error
45	CORE 0	D-CACHE bank 3 feedback checker error
46	CORE 0	D-CACHE Tag RAM error
48	DMA	DMA1 TCD RAM feedback checker error
49	PLL DIG	PLL0 Loss of Lock error
50	PLL DIG	PLL1 Loss of Lock error
51	CMU	XOSC less than IRC, XTAL Pin Floating, EXTAL Pin Floating
52	CMU	Sysclk frequency out of range
53	CMU	Comp_subsys frequency out of range
54	CMU	Other clocks frequency out of range
56	XBAR 1	Crossbar integrity checker (XBIC) fault
57	CORE 2	D-MEM bank 0 feedback checker error
58	CORE 2	D-MEM bank 1 feedback checker error
59	CORE 2	I-CACHE bank 0 feedback checker error
60	CORE 2	I-CACHE bank 1 feedback checker error
61	CORE 2	I-CACHE bank 2 feedback checker error
62	CORE 2	I-CACHE bank 3 feedback checker error
63	FLASH	FLASH read reference error
64	PFLASHC	EDC after ECC error for FLASH Array
65	PFLASHC	EDC after ECC error for FLASH Controller
66	PFLASHC	FLASH Encoding Error
67	PFLASHC	PFLASH Address feedback checker error
69	CORE 2	I-CACHE Tag RAM error
70	CORE 2	D-CACHE bank 0 feedback checker error
71	CORE 2	D-CACHE bank 1 feedback checker error
72	CORE 2	D-CACHE bank 2 feedback checker error
73	CORE 2	D-CACHE bank 3 feedback checker error
74	PRAM 2	System RAM Address Feedback or RAM Late-Write Buffer mismatch
75	PRAM 2	EDC after ECC for PRAM controller error
76	PRAM 3	System RAM Address Feedback or RAM Late-Write Buffer mismatch

Table 1. List of faults (continued)

Channel #	Source	Description
77	PRAM 3	EDC after ECC for PRAM controller error
78	TCU	Test circuitry Group 1 activation
79	TCU	Test circuitry Group 2 activation
80	TCU	Test circuitry Group 3 activation
81	TCU	Test circuitry Group 4 activation
82	CORE 0	Machine check exception indication
83	CORE 2	D-CACHE Tag RAM error
84	CORE 2	Machine check exception indication
88	AIPS	AIPS_1 gasket monitor error
89	AIPS	AIPS_1 e2eEDC error
90	AIPS	AIPS_2 gasket monitor error
91	AIPS	AIPS_2 e2eEDC error
92	PLATFORM	Checker Concentrator 1
93	PLATFORM	Checker Concentrator 0
94	MC_RGM	Safe mode entry request from RGM
95	COMPENSATION CELLS	Pad Compensation Disabled
96	GLUE LOGIC	Error input pin (from the external world)
97	PLATFORM	CORE_0 gasket error
98	PLATFORM	CORE_2 gasket error
99	PLATFORM	DSMC_0 monitor error
101	PLATFORM	DSMC_2 monitor error
103	PLATFORM	Frequency gasket of ZXA (instruction bus)
104	PLATFORM	Frequency gasket of ZXA (data bus)
105	CORE 0	CORE 0 Attempt to access non-existent I-MEM
106	CORE 2	CORE 2 Attempt to access non-existent I-MEM
107	PLATFORM	Checker Concentrator 3
108	PLATFORM	Checker Concentrator 1 XBIC
109	PLATFORM	Checker Concentrator 0 EDC error DMA
110	PLATFORM	Checker Concentrator 3 EDC error HSM
111	PLATFORM	Checker Concentrator 1 EDC error SIPI
112	PLATFORM	Checker Concentrator 1 EDC error FLEXRAY
113	PLATFORM	Checker Concentrator 1 EDC error ETHERNET
114	PLATFORM	DMA1 monitor error

Before starting safety application, the user shall configure a proper reaction for each FCCU input fault source. See “FCCU registers reset values” paragraph in SPC584Cx Microcontroller Reference Manual for the device default configuration (see *Section A.1: Document reference*).

Possible reactions to a failure are:

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

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[d] – 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.

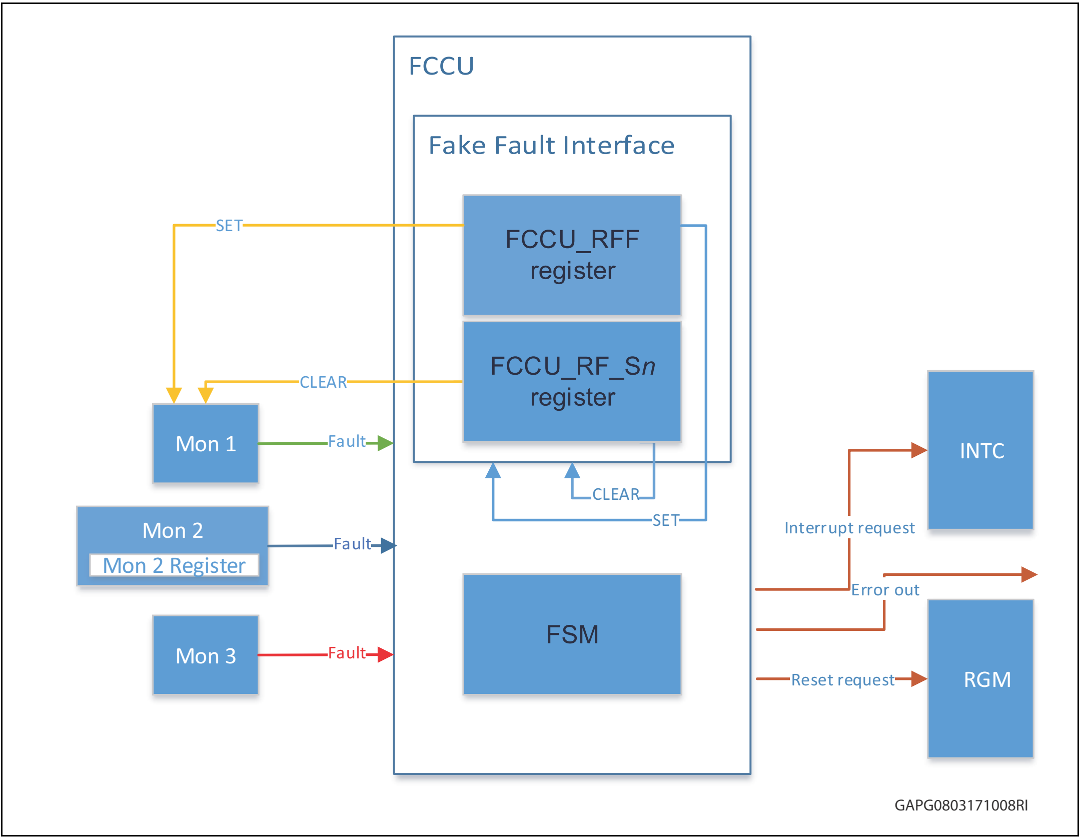
d. There is a 'do nothing' FOSU input coming from the FCCU that indicates that the FCCU is programmed for no reaction.

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 injectable by using the FCCU fake fault interface (“Mon 1” in *Figure 2*);
2. faults injectable by using a SW procedure to stimulate the fault (“Mon 2” in *Figure 2*);
3. faults not injectable (“Mon 3” in *Figure 2*).

Figure 2. FCCU Inner diagram



The FCCU can trigger a fault event directly in the monitor (even if no real failure is occurred) through the fake fault interface.

In order to generate this fault event, an additional - and optional[e] - signal is available (SET signal in FCCU Inner diagram, yellow arrow). The fake fault injection is executed by a write operation into the FCCU_RFF register and the corresponding reaction is not maskable. Some monitors miss the SET signal. In this case (SET signal in FCCU Inner diagram, blue arrow) the write operation into the FCCU_RFF register doesn’t affect the monitor but only the FCCU reaction.

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

In order to clear a fault directly in the monitor, an additional - and optional$^{(e)}$ - signal is available (CLEAR signal in FCCU Inner diagram, yellow arrow). The de-assertion of the FCCU_RF_Sn status bit indicates that the software has properly cleared the fault. Some monitors miss the CLEAR signal. In this case (CLEAR signal in FCCU Inner diagram, blue arrow) the fault can be cleared by a write operation into a specific register of the monitor.

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.

3 Faults description

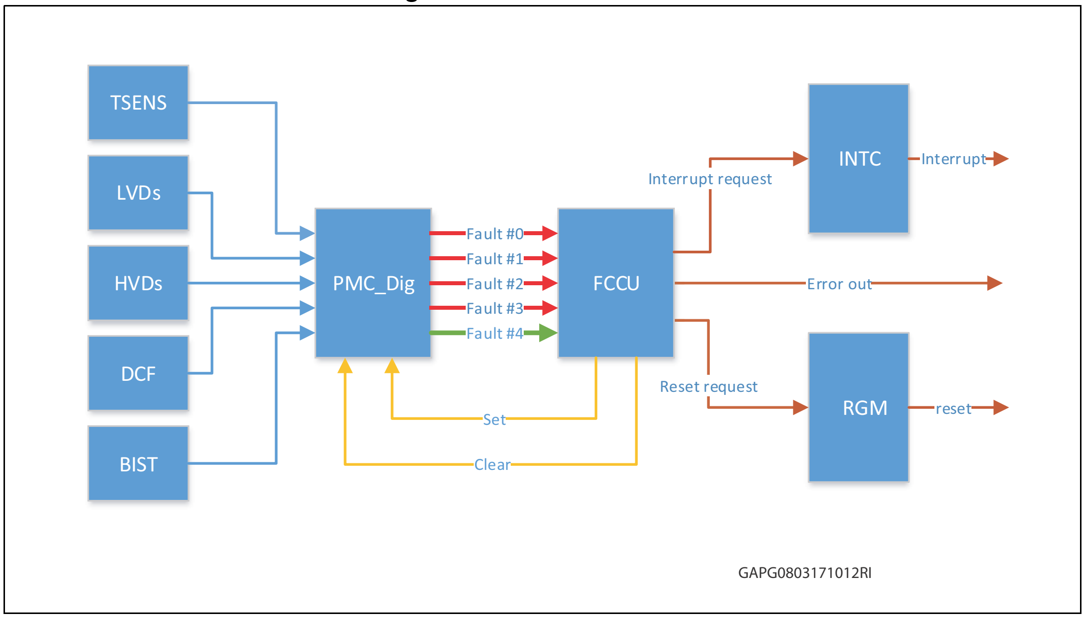
The following sections describe all the faults collected by FCCU for SPC584Cx/SPC58ECx device and how – if possible - to inject them for checking the integrity of the relevant reaction path.
The following convention is adopted in the following figures:

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

3.1 PMC_DIG Faults

PMC_DIG is the source of five different FCCU input faults. Refer to device SPC584Cx Microcontroller Reference Manual for further details on PMC_DIG (see *Section A.1: Document reference*).

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[f], 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.

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.

The MCU embeds seven LVDs (see SPC584Cx 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. The MCU embeds twelve HVDs (see SPC584Cx 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 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 trimming values 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 voltage monitors. In case the BIST fails, the PMC_DIG forwards this fault to the FCCU.

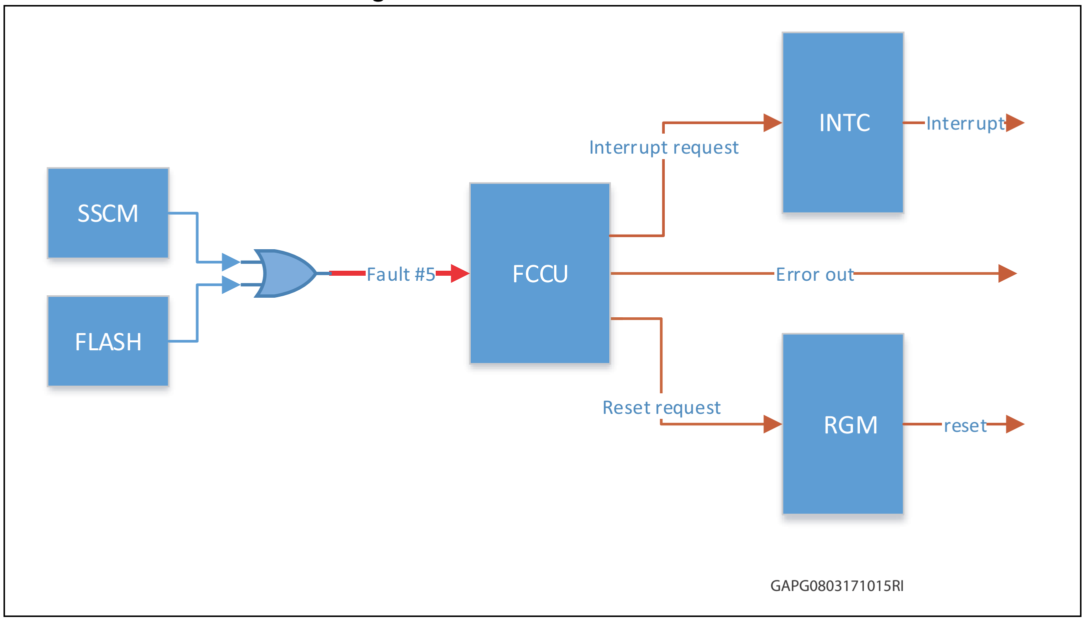
The user can inject this fault by the FCCU fake fault interface.

f. There are three thresholds: TS0, TS1 and TS2. User can configure them in the PMC_DIG relevant registers.

3.2 SSCM/FLASH Fault

The SSCM reads the system configuration from the FLASH[g] and pushes it to the various clients inside the microcontroller. Refer to device SPC584Cx Microcontroller Reference Manual for further details on SSCM (see *Section A.1: Document reference*).

Figure 4. SSCM / FLASH Faults



3.2.1 Transfer 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.

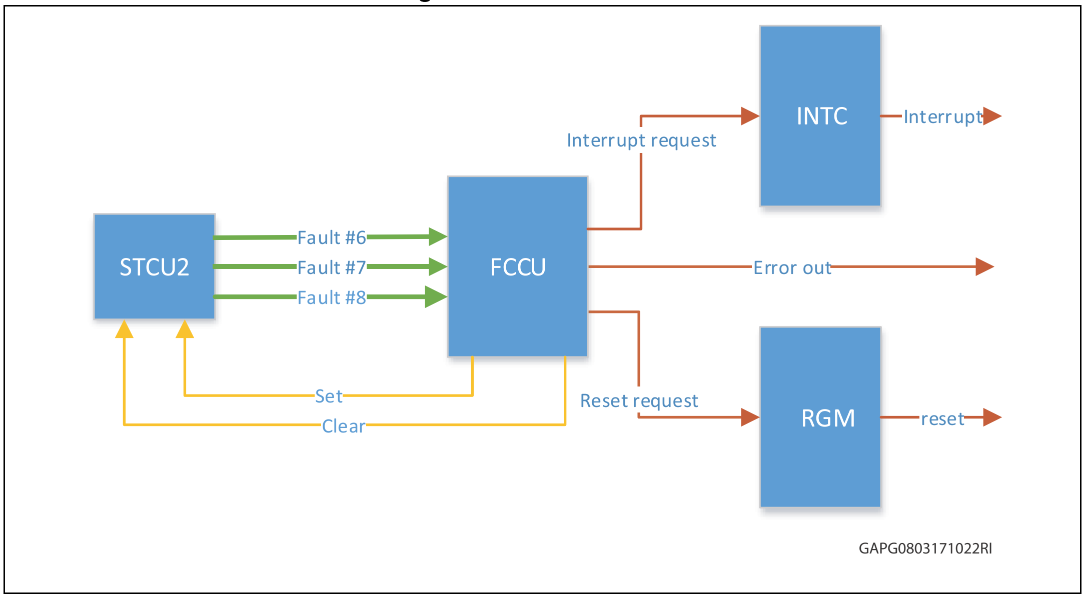
The user can’t inject this fault.

3.3 STCU Faults

The STCU is a comprehensive 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. Refer to device SPC584Cx Microcontroller Reference Manual for further details on STCU.

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

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(h), the STCU forwards this fault to the FCCU. Although BIST is able to detect permanent faults, this fault can be also triggered in case of transient faults. Therefore, the triggering of this fault does not mean that there is a permanent fault and re-running the self-test may be the appropriate action. 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(h), the STCU forwards this fault to the FCCU. Although BIST is able to detect permanent faults, this fault can be also triggered in case of transient faults. Therefore, the triggering of this fault does not mean that there is a permanent fault and a re-running the self-test may be the appropriate action. This fault can be triggered only during BIST execution.

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

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

If MBIST control signals go into an unexpected state during application run-time, it may move the RAM array into the BIST mode during the execution of the application. The STCU detects this condition and forwards it to the FCCU.

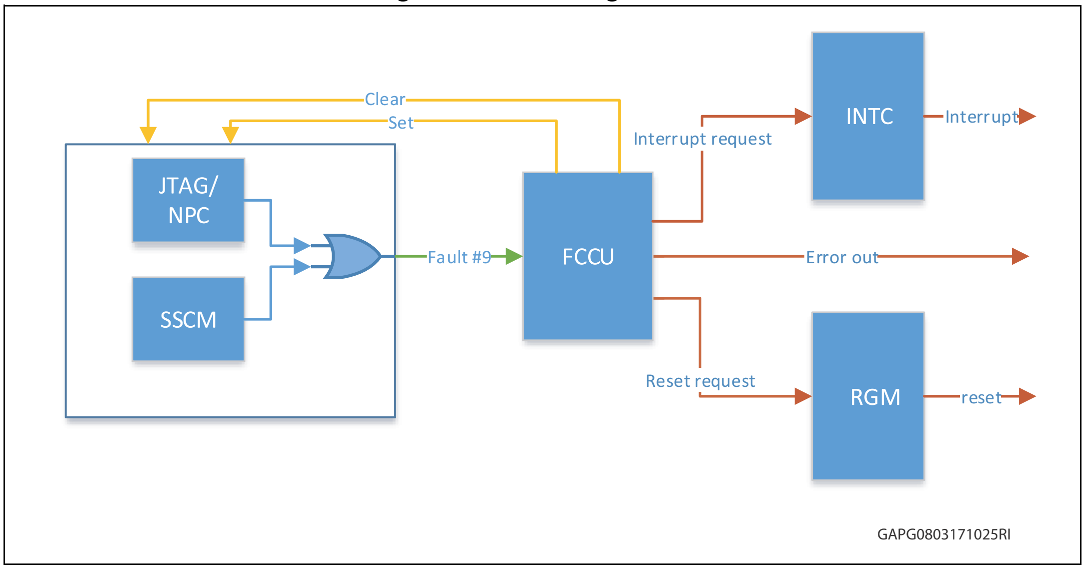
The user can inject this fault by the FCCU fake fault interface.

---

h. 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.

3.4 GLUE LOGIC Faults

Figure 6. GLUE Logic Fault

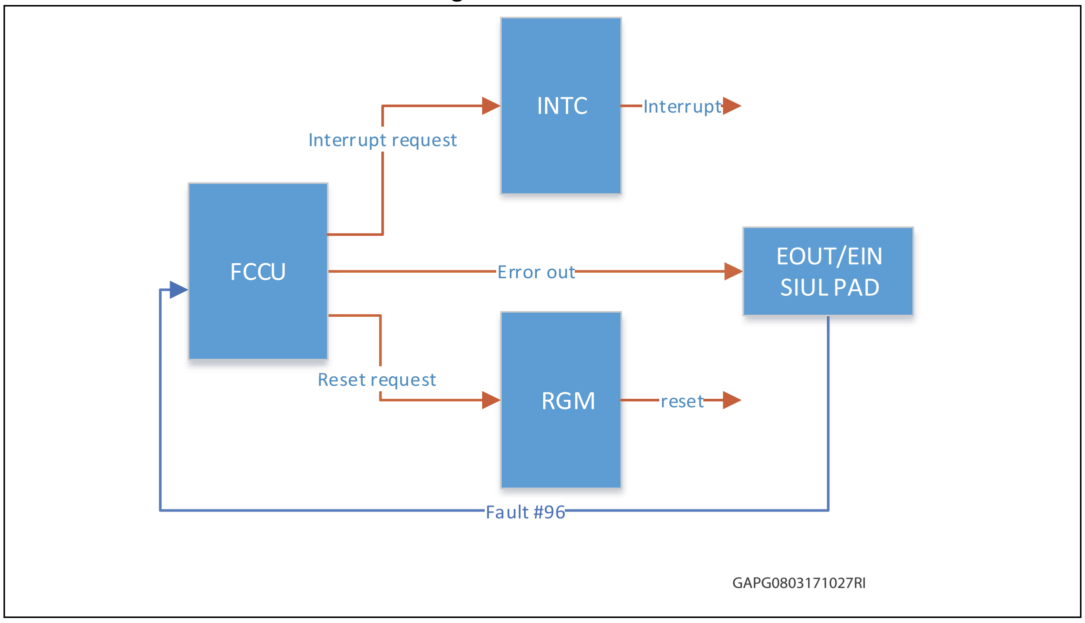


3.4.1 JTAG or NPC not in reset or activation of dangerous debug functionality. Spurious activation of SSCM (Fault #9)

The JTAG/NPC module monitors the unwanted activation of the JTAG/DEBUG Mode. If this event occurs, a dedicated glue logic forwards this fault to the FCCU. The hardware also monitors the unwanted activation of the SSCM and, if this event occurs, a dedicated glue 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)

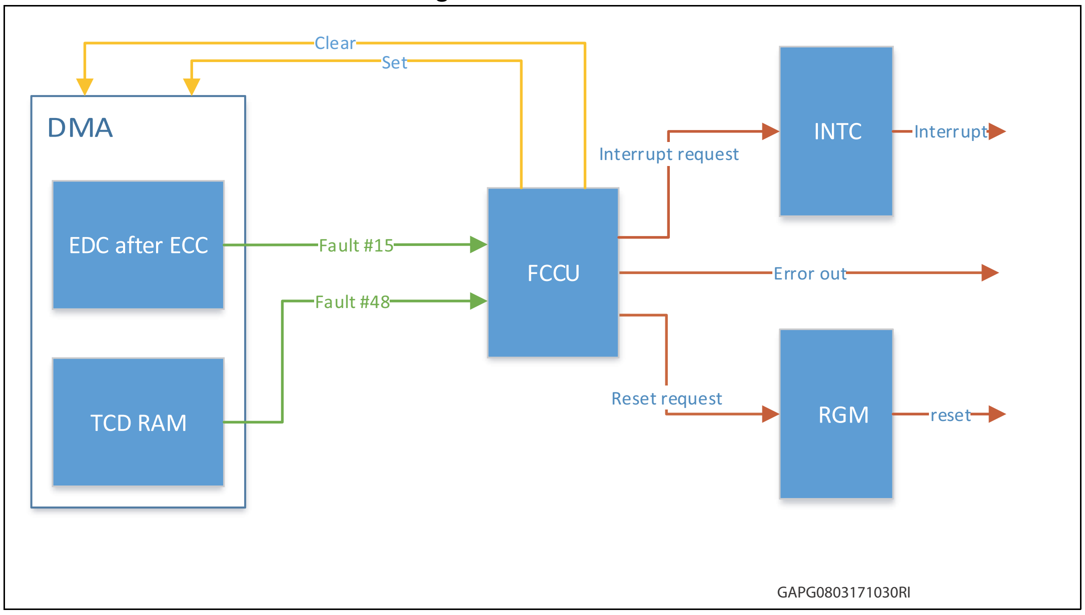
If an external device pulls down the EIN pin, the MCU receives a notification of a faulty condition detected by this external device. A dedicated glue logic forwards this fault to the FCCU.
The EOUT pin driven to FAULTY state (logic ‘0’) asserts back via the pad the EIN signal. This results in a loop where FCCU is in FAULT state and driving EOUT which in turn keeps driving EIN.

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

3.5 DMA Faults

The eDMA controller is a module capable of performing complex data transfers with minimal intervention of a host processor. Refer to device SPC584Cx Microcontroller Reference Manual for further details on eDMA (see *Section A.1: Document reference*).

Figure 8. DMA Faults



3.5.1 DMA TCD EDC after ECC (Fault #15)

The EDC after ECC signals to the FCCU whether a hardware fault in the ECC logic results in a corrupted ECC correction.

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

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

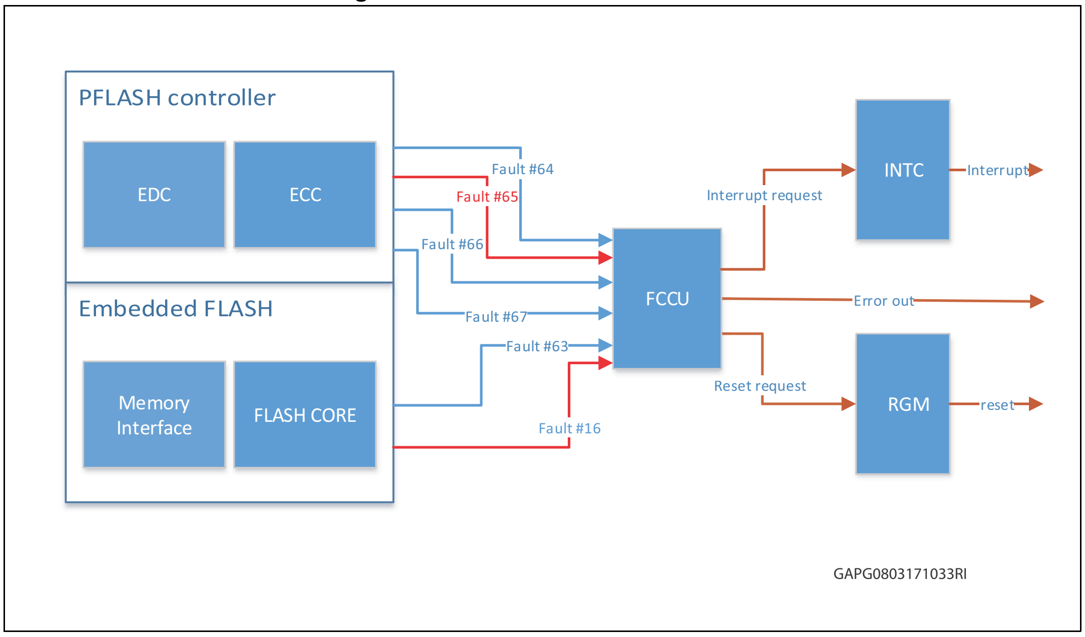
If latched control signals don't match with the ones originally sent to DMA 1 TCD RAM, the feedback checker forwards this fault to the FCCU.

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

3.6 FLASH/PFLASHC Faults

The PFLASH controller is the interface between the FLASH memory and the crossbar switch. It provides FLASH configuration and control functions. Refer to device SPC584Cx Microcontroller Reference Manual for further details on FLASH and PFLASHC (see *Section A.1: Document reference*).

Figure 9. FLASH/PFLASHC Faults



3.6.1 FLASH reset error (Fault #16)

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 copy (startup);
- Double ECC errors on KRAM[i] 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.

i. RAM not visible to the user

3.6.3 EDC after ECC for FLASH Array (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 (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 inject this fault by a SW procedure that accesses to the FLASH address in the Customer Programmable Detection area in the UTEST block.

3.6.5 FLASH Encoding Error (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 FLASH Controller Address Feedback Error (Fault #67)

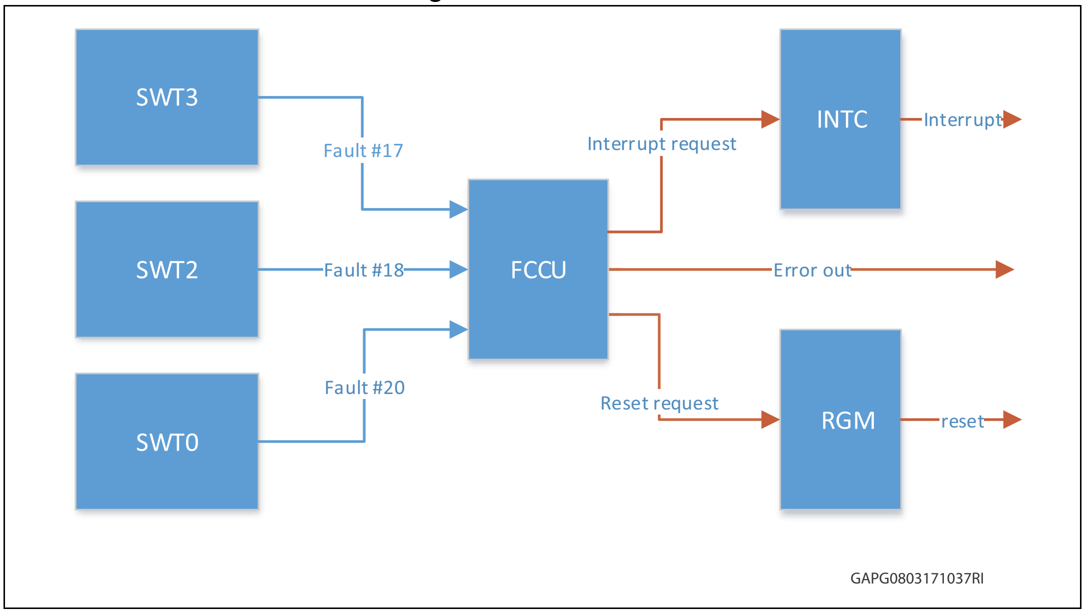
In case a hardware fault occurs resulting in a mismatch between the address from the crossbar and the feedback address from the FLASH, the PFLASHC forwards this fault to FCCU.

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.

3.7 SWT Faults

The SWT is a module that can prevent system lockup in situations such as software getting trapped in a loop or if a bus transaction 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 this servicing action does not occur before the timer expires the SWT generates an interrupt or a reset according to its configuration. Refer to device SPC584Cx Microcontroller Reference Manual for further details on SWT.

Figure 10. SWT Faults



3.7.1 SWT3 reset request (Fault #17)

If the SWT3 reaches a timeout[j] 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[j] 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.

3.8 SWT0 reset request (Fault #20)

If the SWT0 reaches a timeout[j] 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.

3.9 MEMU Faults

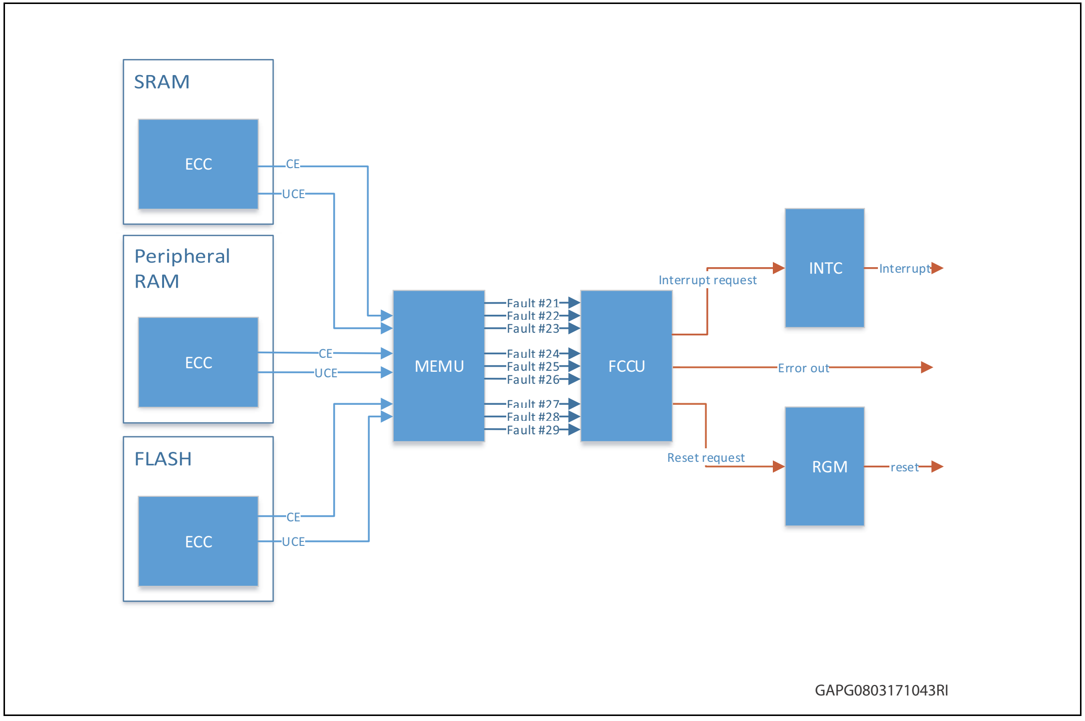
The MEMU is responsible for collecting and reporting error events captured by ECC/EDC logic used in SRAM, Peripheral SRAM and FLASH memories. When any of the following events occurs, the MEMU receives an error signal that causes an event to be recorded. The corresponding error flags is set and reported to FCCU.

j. 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.

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

Refer to device SPC584Cx Microcontroller Reference Manual for further details on MEMU (see *Section A.1: Document reference*).

Figure 11. MEMU Faults



3.9.1 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.

3.9.2 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.

3.9.3 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.

3.9.4 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.

3.9.5 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.

3.9.6 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.

3.9.7 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.

3.9.8 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.

3.9.9 MEMU FLASH overflow error (Fault #29)

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

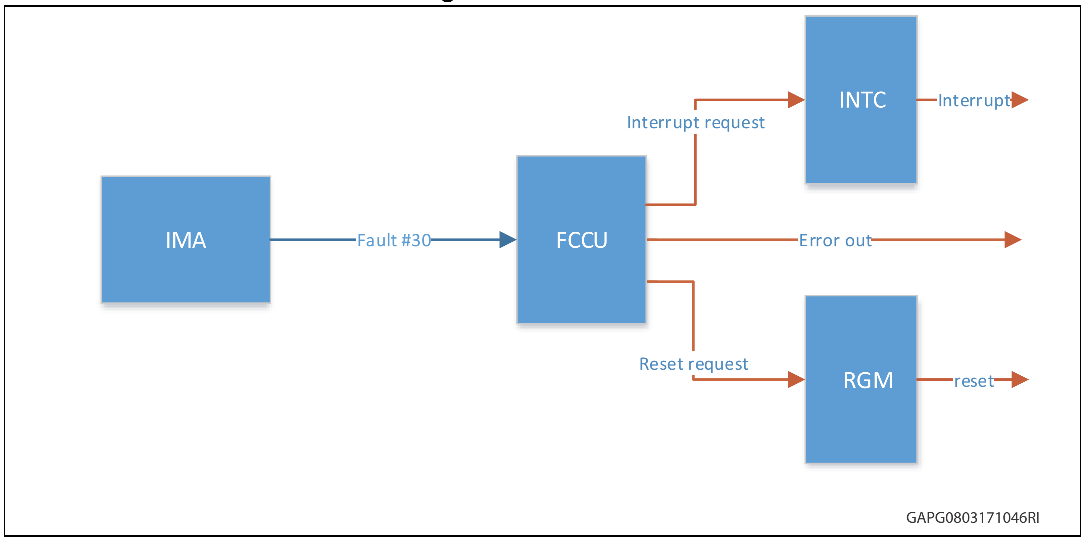
The user can inject this fault by a SW procedure that uses the MEMU DEBUG register.

3.10 IMA Fault

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

Refer to device SPC584Cx Microcontroller Reference Manual for further details on IMA (see *Section A.1: Document reference*).

Figure 12. IMA Fault



3.10.1 IMA SoC Active (Fault #30)

Since unwanted activation of the IMA can interfere with the safety function, the IMA signals to the FCCU when it is reading or writing to the memory. Before any intentional access to the memories by the IMA, the user shall disable the relevant FCCU error input. As a result, the FCCU can detect unwanted activation of IMA and reacts according to its configuration.

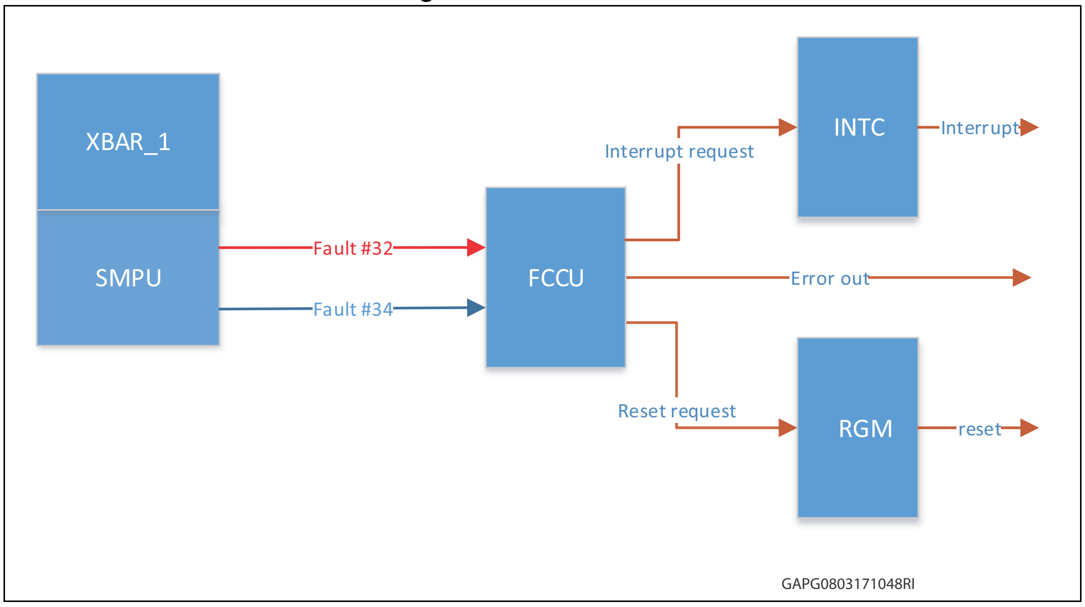
The user can inject this fault by a SW procedure using IMA to access a memory address without disabling the relevant FCCU error input.

3.11 SMPU Faults

The SMPU provides hardware access control for system bus memory references. 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.

Refer to device SPC584Cx Microcontroller Reference Manual for further details on SMPU (see *Section A.1: Document reference*).

Figure 13. SMPU Faults



3.11.1 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.

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

If the SPMU allows an access to a memory location not mapped to any descriptor or with insufficient rights, or the SPMU doesn't allow a memory access with sufficient rights, the hardware monitors inside the SMPU, detect this event and forwards this fault to the FCCU.

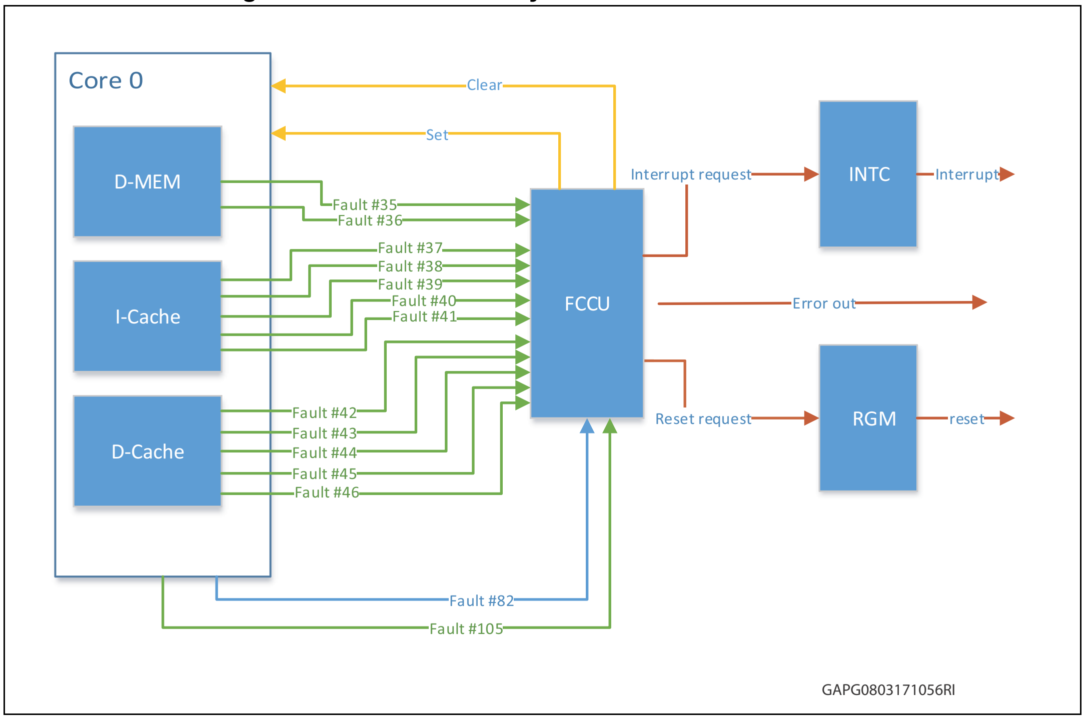
The user can't inject this fault.

3.12 CORE 0 Faults

Memory feedback checkers inside the core 0 are able to 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.

Figure 14. Core 0 Memory feedback checkers faults



3.12.1 D-MEM Bank 0 feedback checker error (Fault #35)

In case control signals latched back to the memory controller don't match the ones originally sent to D-MEM bank 0[k], 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 Bank 1 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 bank 1[k], 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 Bank 0 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 bank 0[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.4 I-CACHE Bank 1 feedback checker error (Fault #38)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 1[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

k. D-MEM, I-CACHE and D_CACHE include multiple banks of memory

3.12.5 I-CACHE Bank 2 feedback checker error (Fault #39)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 2[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.6 I-CACHE Bank 3 feedback checker error (Fault #40)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 3[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.7 I-CACHE Tag RAM 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 Tag RAM[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.8 D-CACHE Bank 0 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 bank 0[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.9 D-CACHE Bank 1 feedback checker error (Fault #43)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 1[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.10 D-CACHE Bank 2 feedback checker error (Fault #44)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 2[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.11 D-CACHE Bank 3 feedback checker error (Fault #45)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 3[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.12 D-CACHE Tag RAM 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 Tag RAM[k], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.12.13 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 the FLASH address in the Customer Programmable Detection area in the UTEST block containing uncorrectable errors.

3.12.14 Core 0 IMEM Illegal access (Fault #105)

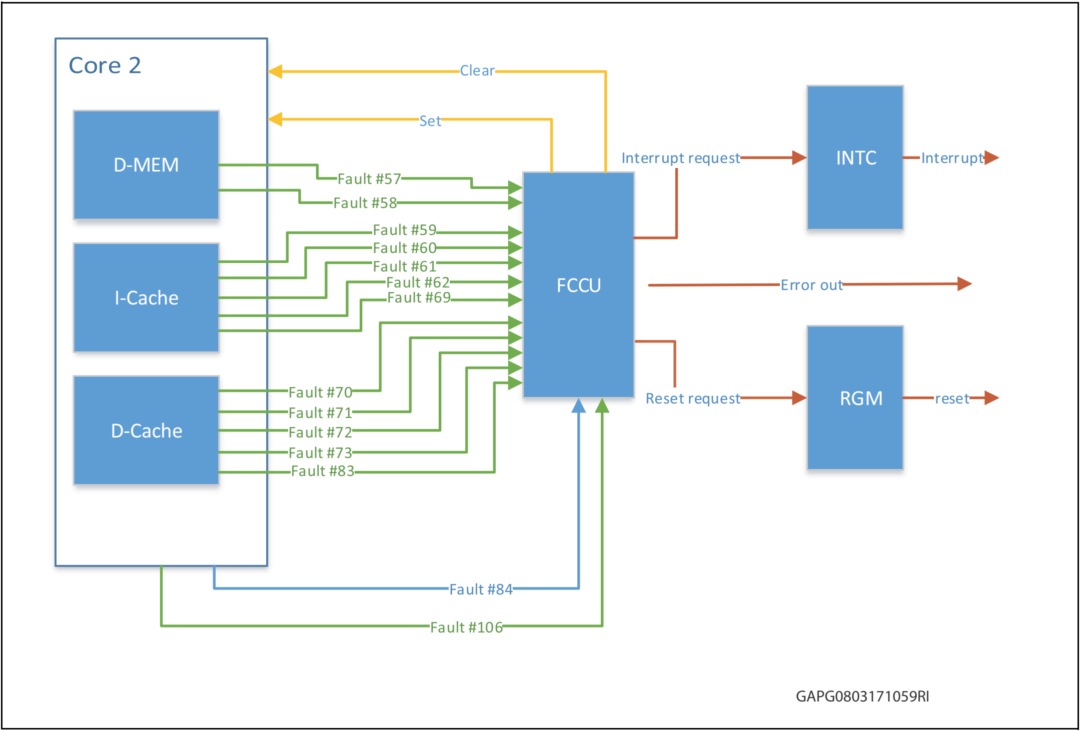
When the core tries to access a not implemented IMEM address, the hardware signals this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13 CORE 2 Faults

Memory feedback checkers inside the core 2 are able to 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.

Figure 15. Core 2 Memory feedback checkers faults



3.13.1 D-MEM Bank 0 feedback checker error (Fault #57)

In case control signals latched back to the memory controller don't match the ones originally sent to D-MEM bank 0[l], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

l. D-MEM, I-CACHE and D_CACHE include multiple banks of memory

3.13.2 D-MEM Bank 1 feedback checker error (Fault #58)

In case control signals latched back to the memory controller don't match the ones originally sent to D-MEM bank 1(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.3 I-CACHE Bank 0 feedback checker error (Fault #59)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 0(l), 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 Bank 1 feedback checker error (Fault #60)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 1(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.5 I-CACHE Bank 2 feedback checker error (Fault #61)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 2(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.6 I-CACHE Bank 3 feedback checker error (Fault #62)

In case control signals latched back to the memory controller don't match the ones originally sent to I-CACHE bank 3(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.7 I-CACHE Tag RAM feedback checker error (Fault #69)

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

3.13.8 D-CACHE Bank 0 feedback checker error (Fault #70)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 0(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.9 D-CACHE Bank 1 feedback checker error (Fault #71)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 1(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.10 D-CACHE Bank 2 feedback checker error (Fault #72)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 2(l), the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.11 D-CACHE Bank 3 feedback checker error (Fault #73)

In case control signals latched back to the memory controller don't match the ones originally sent to D-CACHE bank 3[l], the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.13.12 D-CACHE Tag RAM feedback checker error (Fault #83)

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

3.13.13 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 the FLASH address in the Customer Programmable Detection area in the UTEST block containing uncorrectable errors.

3.13.14 Core 2 IMEM Illegal access (Fault #106)

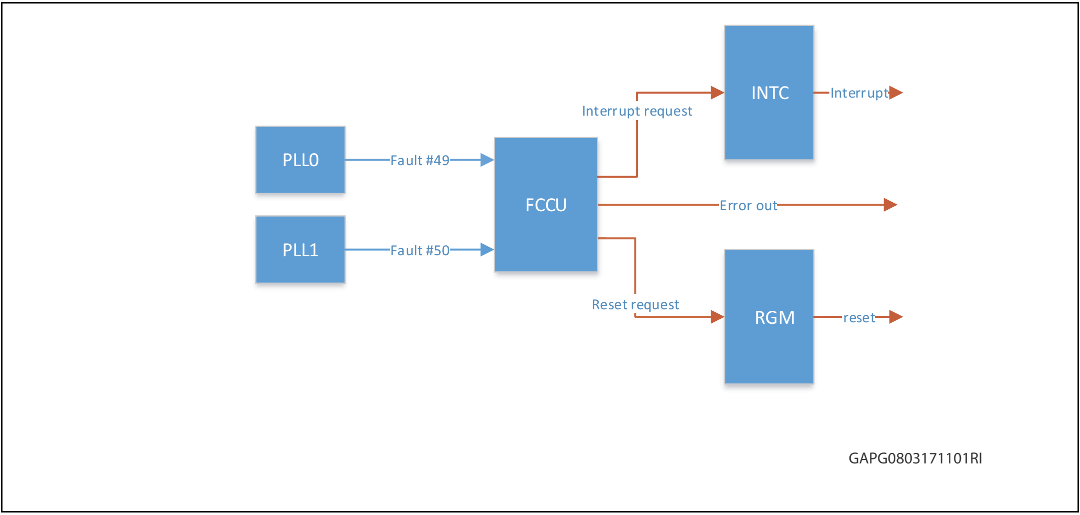
When the core tries to access a not implemented IMEM address, the hardware forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.14 PLLDIG Faults

The SPC584Cx embeds a dual PLL system which provides separate system and peripheral clocks.

Refer to device SPC584Cx Microcontroller Reference Manual for further details on dual PLL (see *Section A.1: Document reference*).

Figure 16. PLL DIG Faults



3.14.1 PLL0 Loss of lock (Fault #49)

A built-in mechanism can detect a loss of lock for PLL0. 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 that generates a loss of lock.

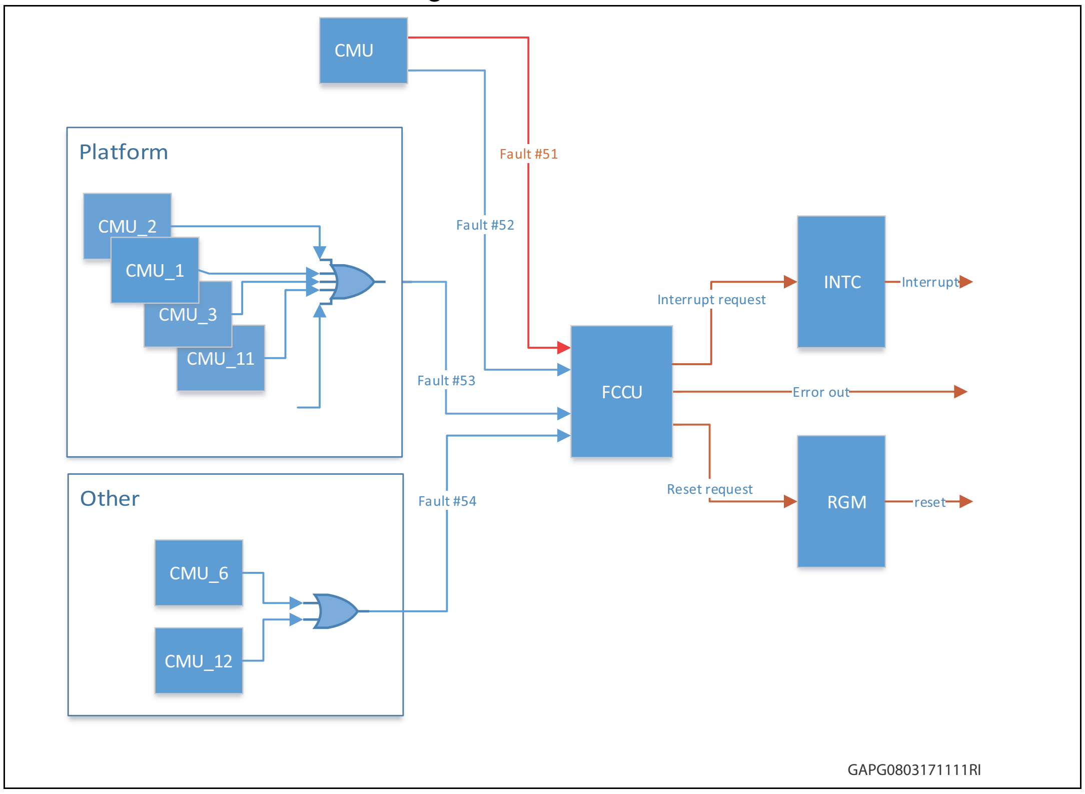
3.14.2 PLL1 Loss of lock (Fault #50)

A built-in mechanism can detect a loss of lock for PLL1. 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 that generates 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[m] this event to the FCCU. Refer to device SPC584Cx Microcontroller Reference Manual for further details on CMUs (see *Section A.1: Document reference*).

Figure 17. CMU Faults



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

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 one of the crystal oscillator pin is not connected), the CMU_0 can detect this event and forwards it to the FCCU.

The user can’t inject this fault.

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

3.15.3 Monitoring internal clocks of comp_subsys (Fault #53)

The CMU_1 monitors the clock frequency used by CORE_0, CORE_2, HSM and XBAR. The CMU_2 monitors the clock frequency used by DMA1, SIPI1, SWT, STM 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 SEM4. The CMU_14 monitors the clock frequency used by M_CAN. 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(m) (n). The user can inject this fault by a SW procedure that sets misconfigured values for the monitoring thresholds.

3.15.4 Monitoring internal clocks of other clocks (Fault #54)

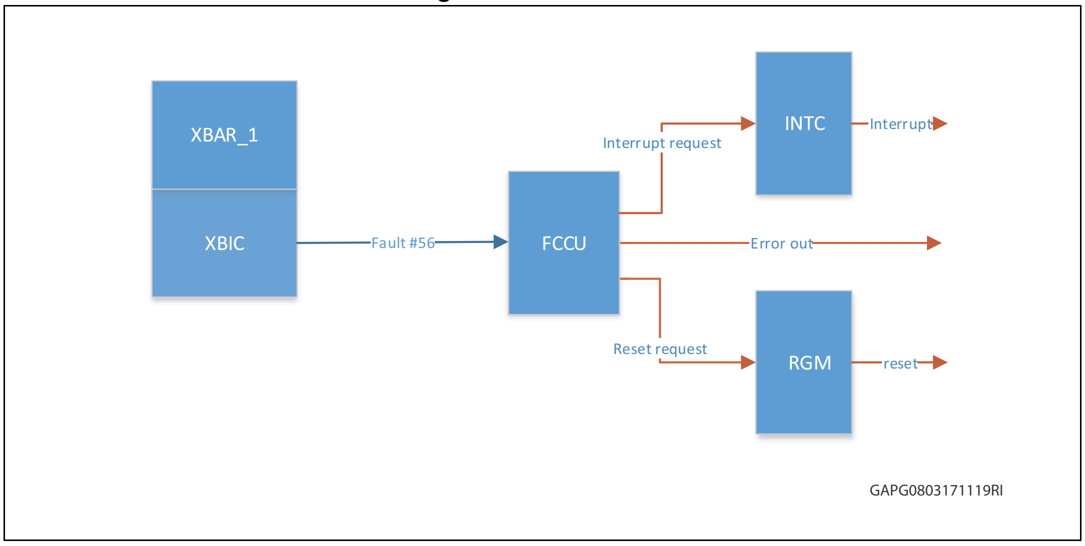
The CMU_6 monitors the clock frequency used by SARADC and the CMU_12 monitors the clock frequency used by EMIOS. If one of these frequencies is above or below the monitoring thresholds, the relevant CMU can detect this event and forwards it to the FCCU(m),(n). The user can inject this fault by a SW procedure 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. Refer to device SPC584Cx Microcontroller Reference Manual for further details on XBIC (*Section A.1: Document reference*).

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

Figure 18. XBIC fault



3.16.1 XBAR1 XBIC fault (Fault #56)

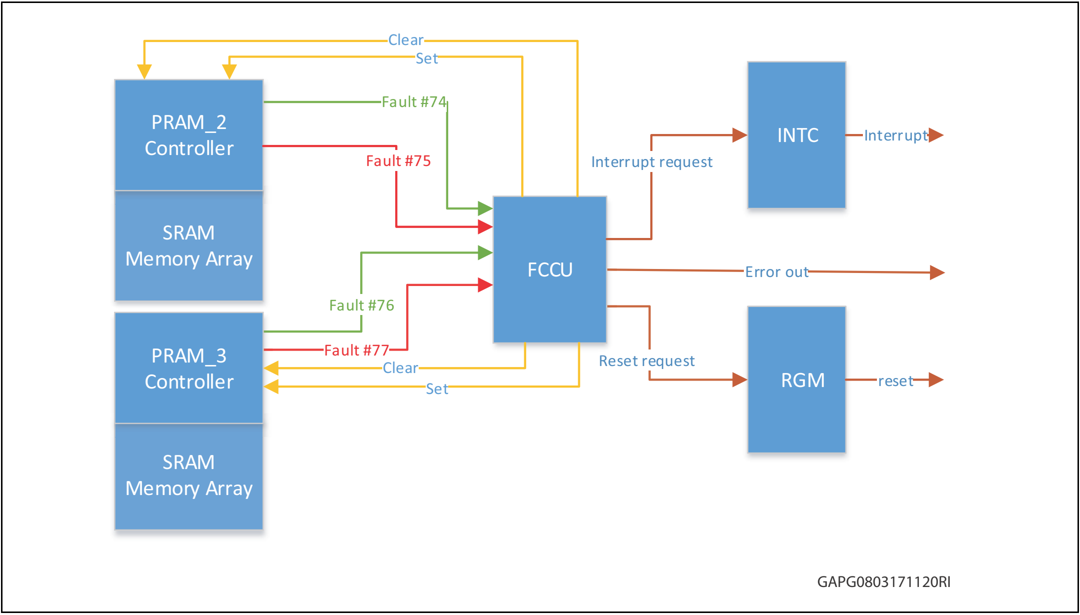
In case of corrupted transaction through the XBAR, the XBIC detects this event and forwards it to the FCCU.

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 acts as an interface between the system bus and the integrated system RAM. It converts the protocols between the system bus and the RAM array interface. The device embeds two controllers: PRAM2 and PRAM3. Refer to device SPC584Cx Microcontroller Reference Manual for further details on PRAM controllers (see *Section A.1: Document reference*).

Figure 19. PRAM Faults



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

In case of addressing error[o] the PRAM controller is able to detect this event and it forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.17.2 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.

3.17.3 System RAM Address Feedback or RAM Late-Write Buffer mismatch PRAM3 (Fault #76).

In case of addressing error[o] the PRAM controller is able to detect this event and it forwards this fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.17.4 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.

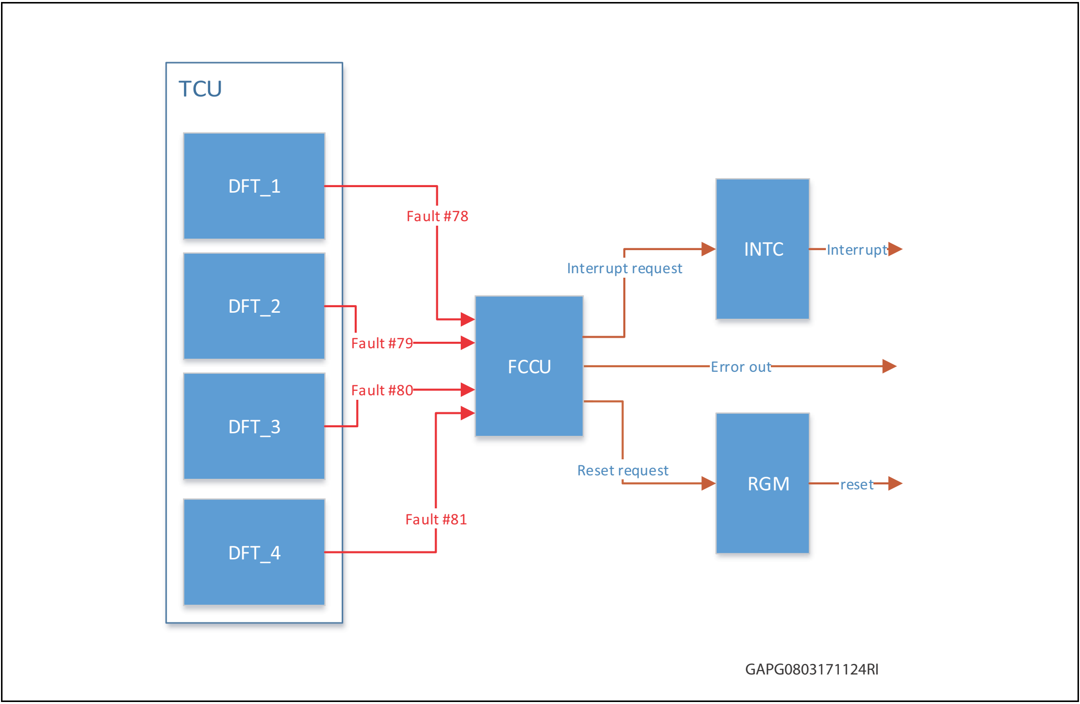
3.18 TCU Faults.

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

o. e.g. a write error in the RAM controller resulting in corrupted RAM access 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 20. 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.

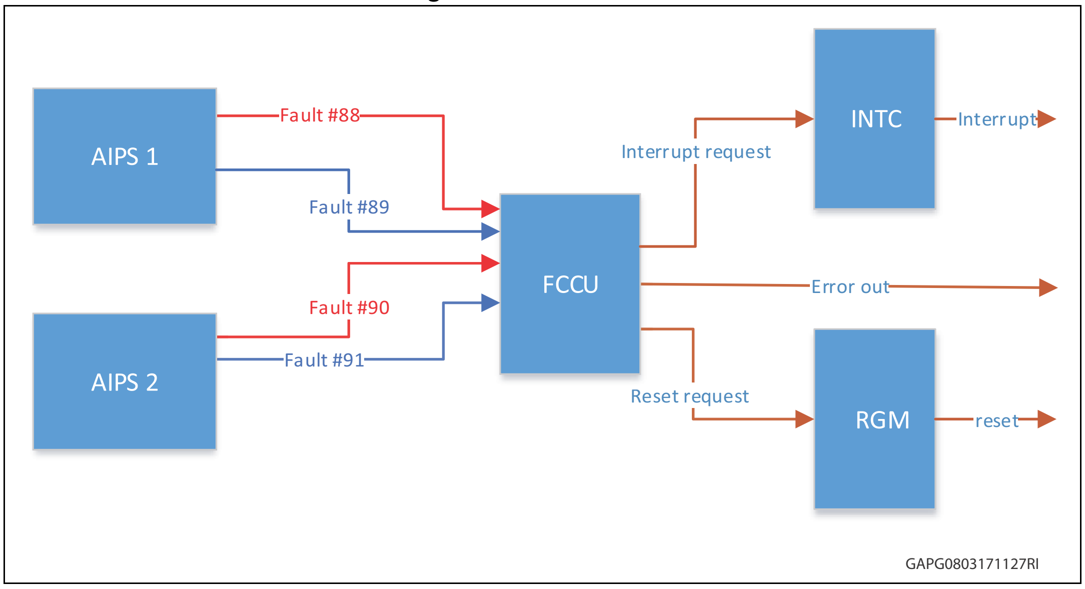
3.18.4 Test circuitry Group 4 activation (Fault #81)

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 two peripheral bridges: AIPS_1 and AIPS_2. Refer to device SPC584Cx Microcontroller Reference Manual for further details on AIPS (see *Section A.1: Document reference*).

Figure 21. AIPS Faults



3.19.1 AIPS_1 gasket monitor error (Fault #88)

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. In case of error, this monitor detects this condition and forwards it to the FCCU.

The user can’t inject this fault.

3.19.2 AIPS_1 e2eEDC error (Fault #89)

A random failure affecting the ECC correction logic of the corresponding master 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 a SW procedure that uses DEBUG core instructions e2ectl0 and e2eecsr0 and write single bit error data into physical CAN_1 RAM (AIPS_1).

3.19.3 AIPS_2 gasket monitor error (Fault #90)

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. In case of error, this monitor detects this condition and forwards it to the FCCU.

The user can’t inject this fault.

3.19.4 AIPS_2 e2eEDC error (Fault #91)

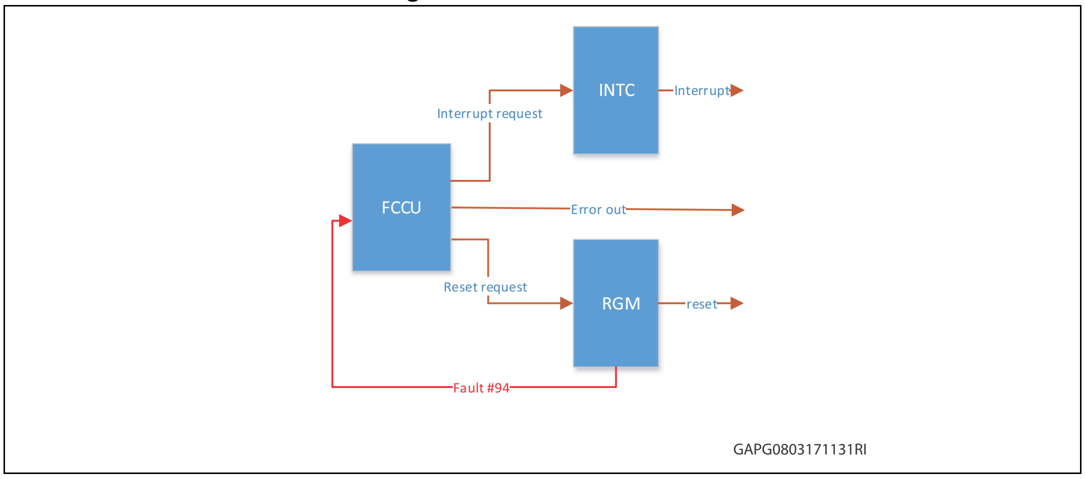
A random failure affecting the ECC correction logic of the corresponding master can cause a corrupted ECC correction. The EDC after ECC can detects this event and forward it to the FCCU.

The user can inject this fault by using DEBUG core instructions e2ectl0 and e2eecsr0 and writing single bit error data into physical CAN_0 RAM (AIPS_2).

3.20 MC_RGM Fault

The MC_RGM centralizes the different reset sources and manages the reset sequence of the chip. Refer to device SPC584Cx Microcontroller Reference Manual for further details on MC_RGM (see *Section A.1: Document reference*).

Figure 22. MC_RGM Fault



3.20.1 Safe Mode Entry Indication (Fault #94)

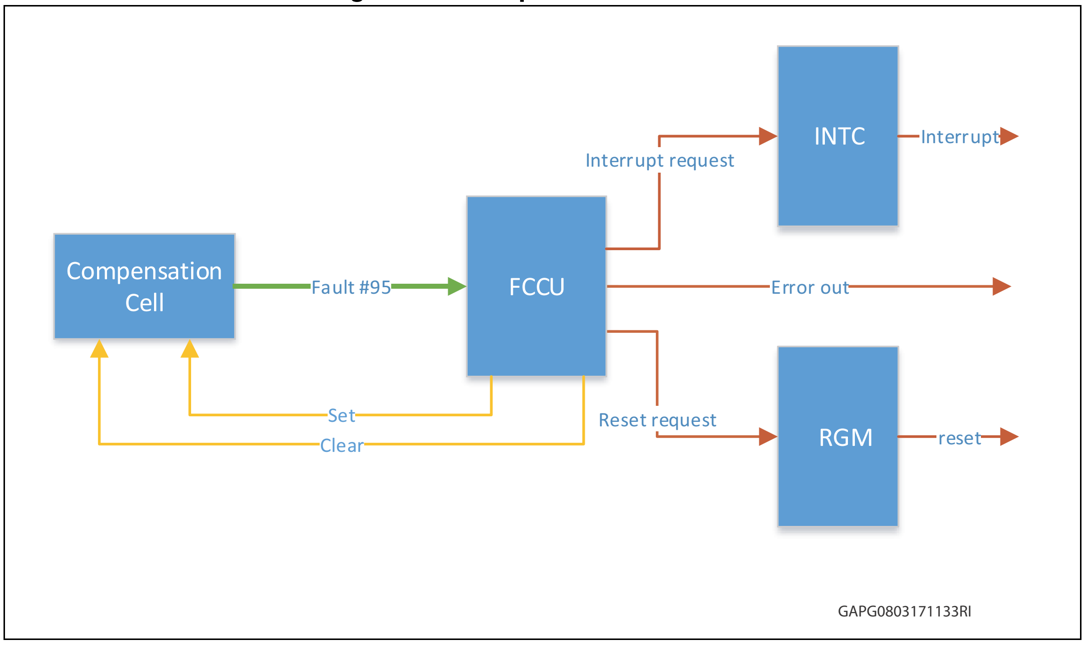
The MC_RGM can request a transition to SAFE mode to the MC_ME. In case of an unwanted safe mode request due to a random event, the hardware detects and forwards this event to the FCCU. The user can't inject this fault.

3.21 Compensation cells Fault

Compensation cells generate 8-bit compensation codes for IO buffers, depending on process, voltage, and temperature (PVT) conditions of the chip. Compensation reduces the spread of some circuit parameters(p) in the IO buffers over temperature, pressure and voltage.

p. e.g. current slew rate and output impedance

Figure 23. Compensation cell fault



3.21.1 Pad Compensation Disabled (Fault #95)

When the compensation cell is in normal mode, it generates the compensation codes. If the normal mode exists, the hardware detects this event and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.22 Platform Faults

3.22.1 CPU0 gasket (Fault #97)

Backdoor port allows a master to access the local memories of the other masters. In case of errors in AHB control signals and in the response signals that transit through CPU0 D-MEM backdoor, the hardware forwards this fault to the FCCU.

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

3.22.2 CPU2 gasket (Fault #98)

Backdoor port allows a master to access the local memories of the other masters. In case of error in AHB control signals and the response signals that transit through CPU0 D-MEM backdoor, the hardware forwards this fault to the FCCU.

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

3.22.3 DSMC 0 monitor error (Fault #99)

DSMC generates atomic read-modify-write bus transactions to the attached slave memory controller. In case of error in DMSC 0[q], the hardware forwards this fault to the FCCU.

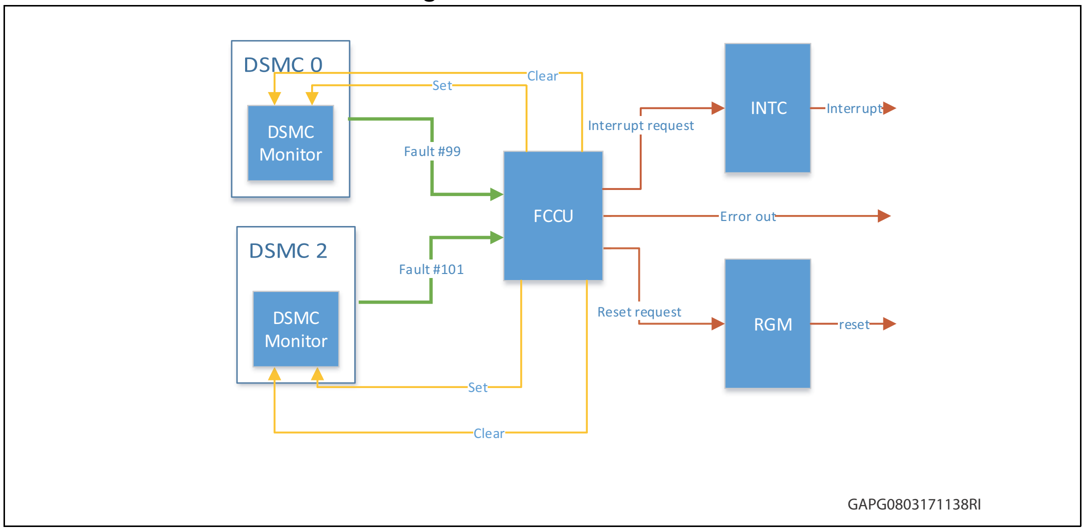
The user can inject this fault by the FCCU fake fault interface.

3.22.4 DSMC 2 monitor error (Fault #101)

DSMC generates atomic read-modify-write bus transactions to the attached slave memory controller. In case of error in DMSC 2[q], the hardware forwards this fault to the FCCU.

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

Figure 24. DSMC faults



3.22.5 Frequency gasket of ZXA (instruction bus) (Fault #103)

Intelligent frequency gaskets interface the Core 0 and the XBAR which run at different frequencies. The hardware embeds a *gasket monitor* that detects failures affecting the gasket connected to the instruction bus. In case of error, the hardware forwards this fault the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.22.6 Frequency gasket of ZXA (data bus) (Fault #104)

Intelligent frequency gaskets interface the Core 0 and the XBAR which run at different frequencies. The hardware embeds a *gasket monitor* that detects failures affecting the gasket connected to the data instruction bus. In case of error, the hardware forwards this fault the FCCU. The user can inject this fault by the FCCU fake fault interface.

3.22.7 Checker Concentrator 0/1/3 (Faults #92, #93, #107)

Three different concentrators exist in the device. They forward and handle data and control signals between crossbar and some master devices. Concentrator 0 is linked to DMA,

q. DMSC 0 is connected to Core 0 and DSMC 2 is connected to Core 2.

concentrator 1 is linked to SIPI, FLEXRAY and ETHERNET and concentrator 3 is linked to HSM. 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. In case of error, it forwards a fault to the FCCU. The user can inject this fault by the FCCU fake fault interface.

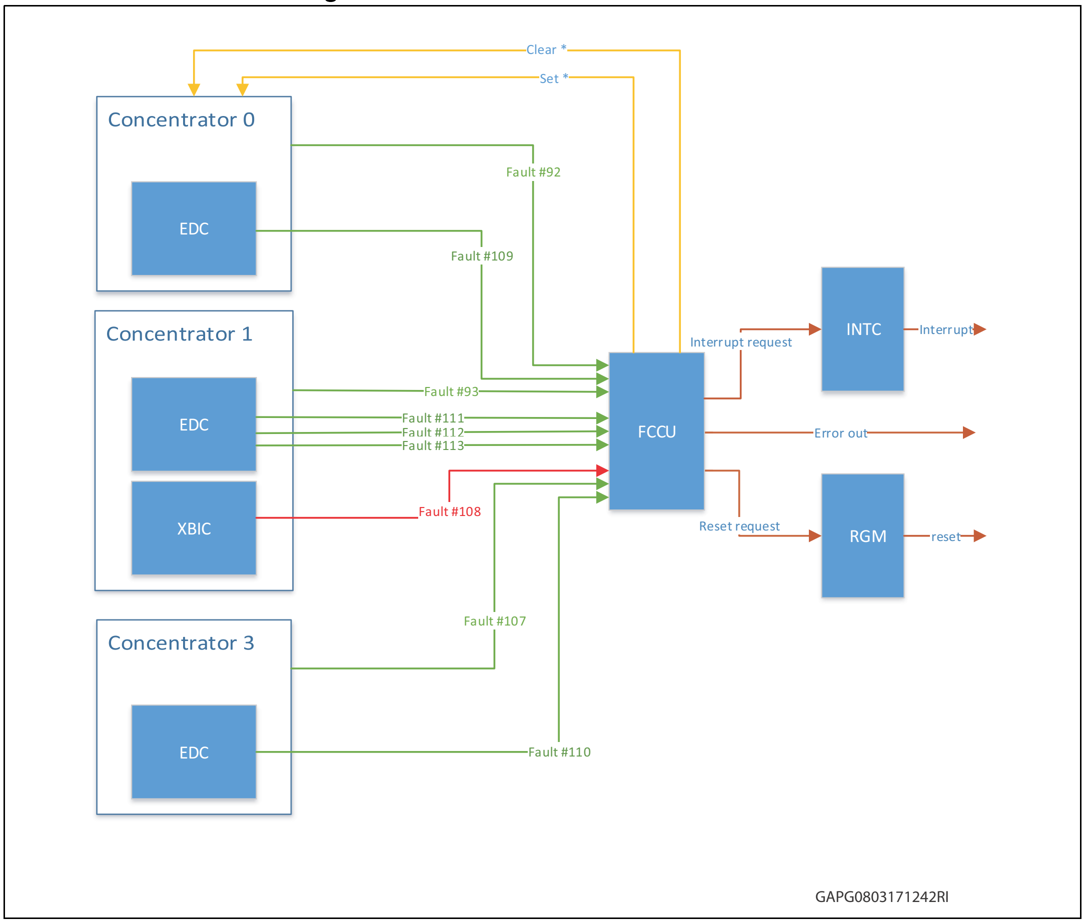
3.22.8 Checker Concentrator 1 XBIC (Fault #108)

Concentrator 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, it forwards a fault to the FCCU. The user can’t inject this fault.

3.22.9 Checker Concentrator 0/1/3 EDC errors (Faults #109, #110, #111, #112, #113)

Each concentrator embeds the EDC after ECC logic. In 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.

Figure 25. Checker concentrator faults



Note: Set and Clear signals apply to all three concentrators for each faults for which they are enabled.

3.22.10 DMA monitor error (Fault #114)

DMA is connected to the XBAR via a frequency gasket. In addition, the hardware embeds a *gasket monitor* that detects failures affecting the gasket. In case of error, it is detects and forwards it to the FCCU. The user can inject this fault by the FCCU fake fault interface.

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[(r)]. The methodology for these tests depends on the specific FCCU input. The idea, however, is to inject a fault and verifies whether the FCCU correctly receives it.

This document - with reference to SPC584Cx/SPC58ECx devices - describes the FCCU faults inputs and how to verify their reaction path (see *Section A.1: Document reference*).

r. Not all FCCU inputs are testable.

Appendix A Document management

A.1 Document reference

SPC584Cx Microcontroller Reference Manual, DocID028117 (RM0407).

A.2 Acronyms

Table 2. Acronyms

Acronym	Name
BIST	Built In Self-Test
DCF	Device Configuration Format
DSMC	Decorated Storage Memory Controller
EDC / ECC	Error Detection Code / Error Correction Code
eDMA	Enhanced Direct Memory Access
FCCU	Fault Collection and Control Unit
FOSU	FCCU output supervision unit
HVD	High Voltage Detector
IMEM	Instruction Memory
INTC	Interrupt Controller
IRCOSC	Internal 16 MHz RC oscillator
IRQ	Interrupt Request
JTAG/NPC	Joint Test Action Group / Nexus Debug Port
LBIST	Logic Built-in self-test
LVD	Low Voltage Detector
MC_ME	Mode entry module
NMI	Non-maskable interrupts
NPC	Nexus debug port
PFLASHC	Platform FLASH Controller
PLL	Phase Lock Loop
PMC	Power Management Control
PMC_DIG	Power Management Controller Digital Interface
PRAM	Platform RAM Controller
POR	Power On Reset
RGM	Reset Generation Module
RM	Reference Manual
SMPU	System Memory Protection Unit

Table 2. Acronyms (continued)

Acronym	Name
SoC	System On Chip
SRAM	System RAM
SSCM	System status and configuration module
STCU	Self-Test Control Unit
TCD	Transfer Control Descriptor
TCU	Test Control Unit
XBAR	CrossBAR
XBIC	Crossbar integrity checker
XOSC	External oscillator/crystal

Revision history

Table 3. Document revision history

Date	Revision	Changes
03-Apr-2017	1	Initial release.