AN5752

Application note

SPC582Bx FCCU fault sources and reaction

Introduction

This application note describes the FCCU failure input 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 SPC582Bx (40 nm–Body–ASIL B). 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.

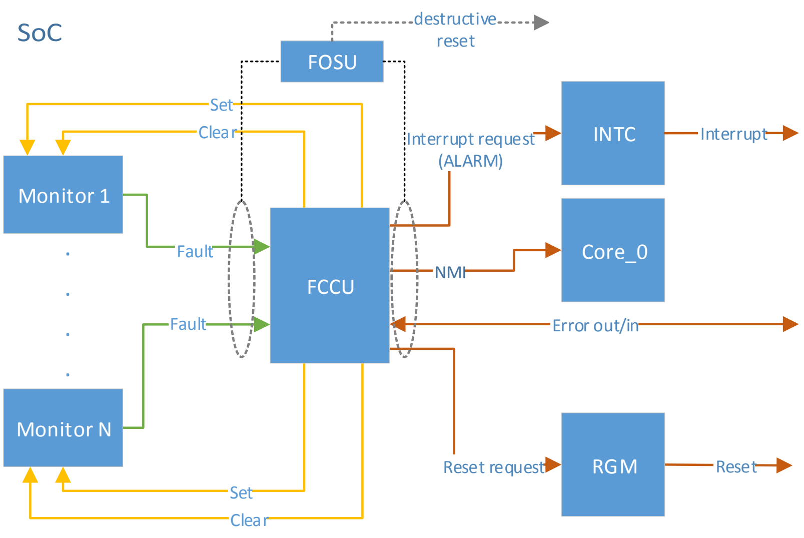
Before reading this document, the reader should have a clear understanding about the usage of FCCU. For further details on this module, refer to “Fault Collection and Control Unit (FCCU)” chapter of the SPC582Bx microcontroller reference manual RM0403. For description of the functional and electrical problems of the SPC582Bx devices, the reader should also refer to the SPC582Bx errata sheet ES0413.

A reference code is available.

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.

The Figure 1 shows how the FCCU is connected to the other blocks. The reader shall consider the above 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 (the safety manual defines the safe states) without any core 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.

Note: The safety analysis assumes a trip time of 12 hours.

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 to fulfill the previous requirement. Note that the user cannot test the error reaction path for certain monitors.

The Table 1 lists and describes all FCCU input fault sources for SPC582Bx MCUs.

Table 1. FCCU failure inputs

FCCU input #	Source	Failure description	Error reaction path
0	PMC DIG	Temperature out of range	Not testable
1	PMC DIG	Voltage out of range from LVDs	Not testable
2	PMC DIG	Voltage out of range from HVDs	Not testable
3	PMC DIG	Digital PMC initialization error during DCF data load	Not testable
4	PMC DIG	Digital PMC voltage detector BIST	Testable
5	SSCM/FLASH_0	SSCM transfer error OR Flash memory initialization error	Not testable
6	STCU	BIST result-wrong signature (STCU unrecoverable fault)	Testable

FCCU input #	Source	Failure description	Error reaction path
7	STCU	BIST result-wrong signature (STCU recoverable fault)	Testable
8	STCU	MBIST control activation	Testable
9	GLUE LOGIC	JTAG, NPC or debug functionality moved out of reset or SSCM activation	Testable
10	PLATFORM/DMA	DMA_1 gasket monitor error	Not testable
11	PLATFORM/DSMC	DSMC ECC error	Not testable
12	PLATFORM/CORE	Core_2 I-bus ECC error	Not testable
13	PLATFORM/CORE	Core_2 D-bus ECC error	Not testable
14	PLATFORM/DMA	DMA_1 ECC error	Not testable
15	PLATFORM/DMA	DMA_1 TCD EDC after ECC	Not testable
16	FLASH_0/1	Flash reset error	Not testable
18	PLATFORM/SWT	SWT_2 reset request	Testable
21	MEMU	System RAM ECC correctable error	Testable
22	MEMU	System RAM ECC uncorrectable error	Testable
23	MEMU	System RAM ECC overflow error(1)	Testable
24	MEMU	Peripheral RAM ECC correctable error	Testable
25	MEMU	Peripheral RAM ECC uncorrectable error	Testable
26	MEMU	Peripheral RAM ECC overflow error(2)	Testable
27	MEMU	Flash ECC correctable error	Testable
28	MEMU	Flash ECC uncorrectable error	Testable
29	MEMU	Flash ECC overflow error	Testable
30	IMA	IMA activation	Testable
32	PLATFORM/SMPU	SMPU XBAR 1 monitor incorrectly refuses an access	Not testable
34	PLATFORM/SMPU	SMPU XBAR 1 monitor correctly refuses an access	Testable
48	PLATFORM/DMA	DMA_1 TCD RAM feedback error	Testable
49	PLL DIG	PLL0 loss of lock error	Testable
50	PLL DIG	PLL1 Loss of Lock error	Testable
51	CMU	CMU_0 error (XOSC less than IRC, XTAL pin floating, EXTAL pin floating)	Not testable
52	CMU	Frequency out of range(3)	Testable
53	CMU	Frequency out of range(4)	Testable
54	CMU	Frequency out of range(5)	Testable
56	PLATFORM/XBAR 1	XBIC error detected	Testable
63	FLASH_0	Flash read reference error	Testable
64	PLATFORM/FLASH_1	EDC after ECC error for Flash array	Testable
65	PLATFORM/FLASH_1	EDC after ECC error for Flash controller	Not testable
66	PLATFORM/FLASH_1	Flash encoding error	Testable
67	PLATFORM/FLASH_1	PFLASH address feedback error	Testable
74	PLATFORM/PRAM 2	Corrupted system RAM access or late-write buffer mismatch	Not testable
75	PLATFORM/PRAM 2	EDC after ECC for system RAM	Not testable
78	TCU	Test circuitry group 1 activation	Not testable

FCCU input #	Source	Failure description	Error reaction path
79	TCU	Test circuitry group 2 activation	Not testable
80	TCU	Test circuitry group 3 activation	Not testable
81	TCU	Test circuitry group 4 activation	Not testable
84	PLATFORM/CORE	Safety Core_2 exception (machine check exception)	Testable
89	PLATFORM/PBRIDGE	PBRIDGE_1 e2eEDC error	Not testable
91	PLATFORM/PBRIDGE	PBRIDGE_2 e2eEDC error	Not testable
94	MC_RGM	Safe mode entry indication	Not testable
95	COMPENSATION CELLS	Pad compensation deactivation	Not testable
96	GLUE LOGIC	Error input pin (from the external world)	Testable

1. *Aggregate of RAM correctable error overflow flag, RAM uncorrectable error overflow flag, RAM error buffer overflow flag.*
2. *Aggregate of peripheral RAM correctable error overflow flag, peripheral RAM uncorrectable error overflow flag, peripheral RAM error buffer overflow flag.*
3. *FHH or FLL event for CMU_0.*
4. *FHH or FLL event from CMU_1, CMU_2, CMU_3, CMU_11, CMU_14.*
5. *FHH or FLL event from CMU_6, CMU_12.*

Before the safety application starts, the user shall configure a proper reaction for each FCCU failure input source. See “FCCU registers reset values” paragraph in SPC582Bx Reference Manual for the device default configuration.

Possible reactions to a failure are:

- Internal reactions:

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

  - 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 core intervention.

A dedicated module called 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 – fault.

Note: There is a 'do nothing' FOSU input coming from the FCCU that indicates that the FCCU is programmed for no reaction.
The FOSU does not require any configuration done by the user. A functional reset has no impact on the FCCU.

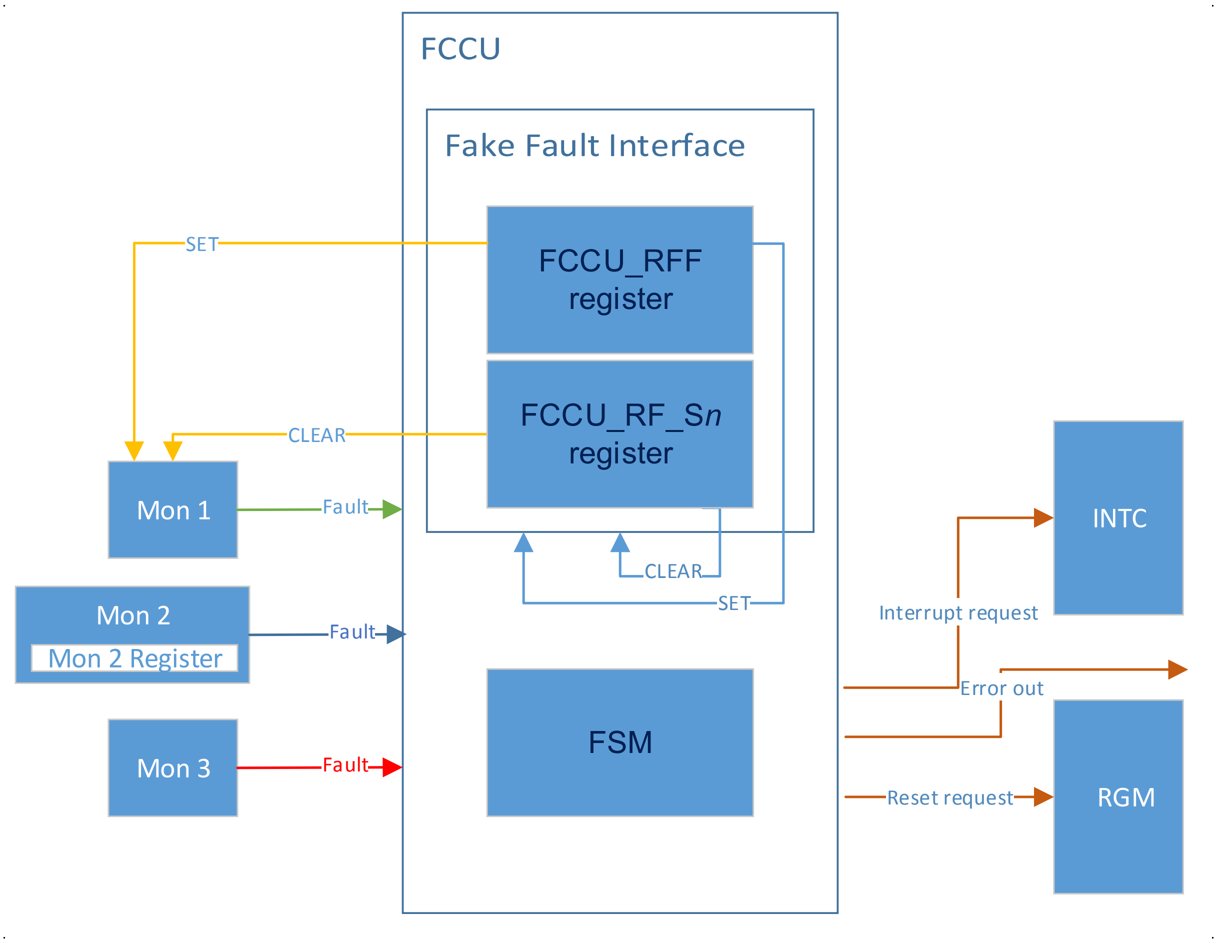
2 FCCU fault injection, clearing and fake fault interface

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.

Three different sets of fault inputs can be distinguished:

- faults injectable by using the FCCU fake fault interface ("Mon 1" in Figure 2);
- faults injectable by using a SW procedure to stimulate the fault ("Mon 2" in Figure 2);
- 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.

To generate this fault event, an additional (and optional) signal is available (set signal in Figure 2, yellow arrow).

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

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 Figure 2, blue arrow) the write operation into the FCCU_RFF register does not affect the monitor but only the FCCU reaction.

To clear a fault directly in the monitor, an additional (and optional) signal is available (clear signal in Figure 2, 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 Figure 2, 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, that is, 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, that is, 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, that is, 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 SPC582Bx 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:

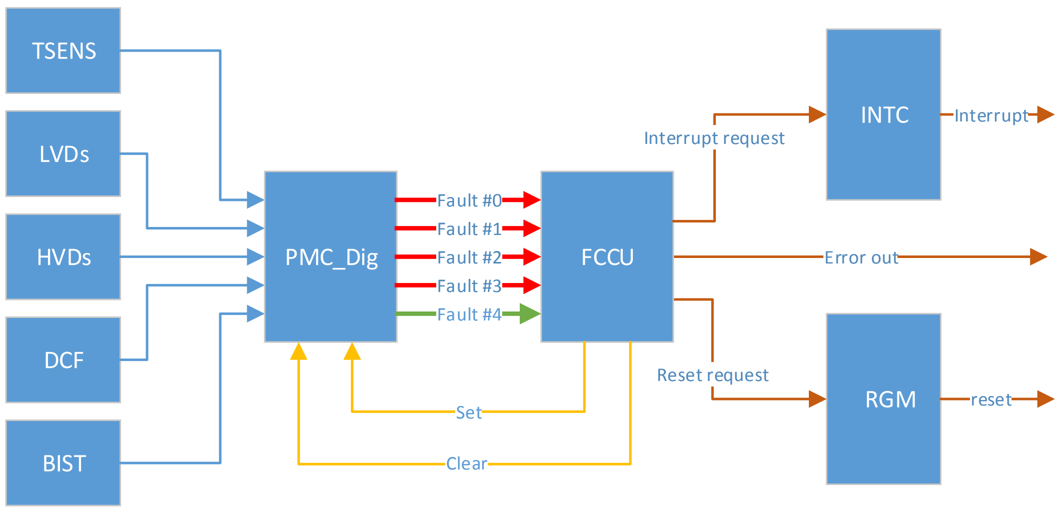
1. a green arrow marks the faults injectable by the FCCU fake fault interface;
2. a blue arrow marks the faults injectable by using a SW procedure to stimulate the fault;
3. a red arrow marks the faults that the user cannot inject.

3.1 PMC_DIG faults

PMC_DIG is the source of five different FCCU input faults.

For further details on PMC_DIG, refer to the device SPC582Bx microcontroller reference manual RM0403.

Figure 3. PMC_DIG faults



3.1.1 Temperature detector out of range (fault #0)

The temperature detector detects if the temperature exceeds the defined thresholds and the PMC_DIG forwards this fault to the FCCU. There are three thresholds: TS0, TS1, and TS2. Temperature detector thresholds are trimmed at testing phase and cannot be configured by the user.

The user cannot inject this fault.

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

Each LVD detects a voltage that drops below the defined threshold and the PMC_DIG forwards this fault to the FCCU. The MCU embeds some LVDs (for further details on LVDs, refer to SPC582Bx reference manual RM0403) and their output signals are put in OR before arriving at the FCCU failure input #1.

The user cannot inject this fault.

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

Each HVD detects a voltage that raises above the defined threshold and the PMC_DIG forwards this fault to the FCCU. The MCU embeds some HVDs (for further details on LVDs, refer to SPC582Bx reference manual RM0403) and their output signals are put in OR before arriving at the FCCU failure input #2.

The user cannot inject this fault.

3.1.4 Digital PMC initialization error during DCF data load (fault #3)

DCF records are used to configure certain registers in the device during system boot. If an error occurs while the SSCM loads the values into the PMC registers, the PMC_DIG forwards this fault to the FCCU.

The user cannot inject this fault.

3.1.5 Digital PMC voltage detector BIST (fault #4)

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.

The user must set the PMC_DIG_BIST_CTRL[NCFEN] bit to enable a user BIST not critical fault indication to the FCCU and inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x04. The FCCU error reaction path is verified if both the FCCU_RF_S0[RFS4] and the BIST_CTRL[NCFST] status bits are set.

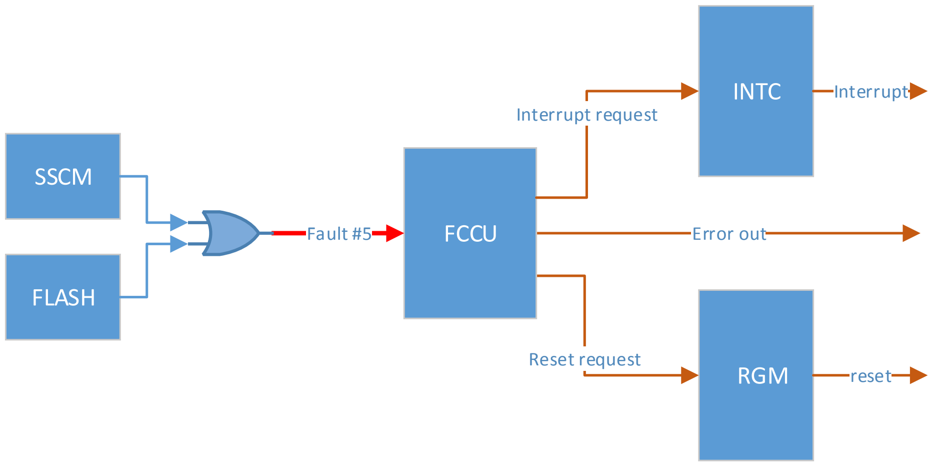
The BIST_CTRL[NCFST] status bit indicates a BIST fail on completion of BIST sequence, and it is cleared by HW on clearing the relevant FCCU_RF_S0[RFS4] bit.

3.2 SSCM/Flash_0 fault

The SSCM reads the system configuration from the Flash memory and pushes it to the various clients inside the microcontroller.

Note: *System configuration is mainly saved as DCF records that are located either in the test or UTest sectors of the Flash memory.*
For further details on SSCM, refer to the device SPC582Bx reference manual RM0403.

Figure 4. **SSCM/Flash faults**



3.2.1 SSCM transfer error OR Flash memory initialization error (fault #5)

A fault can occur while the SSCM loads the STCU configuration and the SSCM forwards this fault to the FCCU. Moreover, an unexpected condition, for example, ECC double-bit detections on the reset reads, can occur within the Flash memory during its initial configuration and the Flash memory forwards this fault to the FCCU.

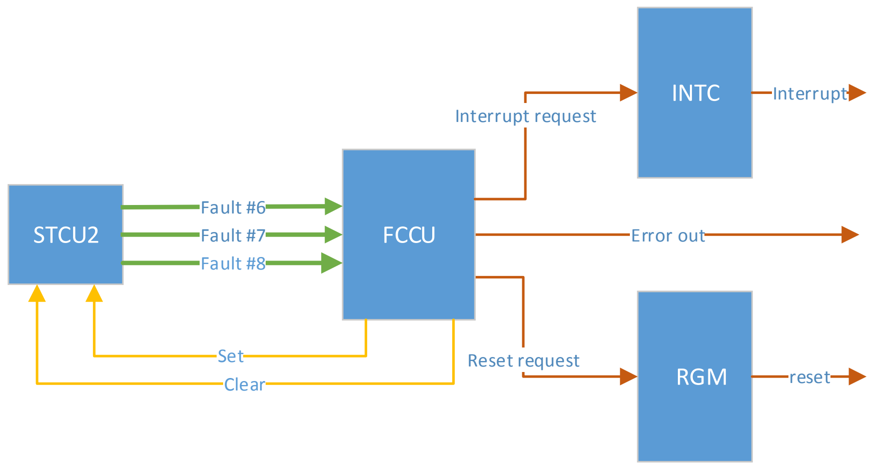
The two error signals are put in OR before arriving to the FCCU failure input #5.

The user cannot 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. For further details on STCU, refer to the device SPC582Bx microcontroller reference manual RM0403.

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 fault, the STCU forwards this fault to the FCCU.

Note: 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.
Although BIST is able to detect permanent faults, this fault can be also triggered in case of transient faults. The STCU can trigger this fault only during BIST execution.

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

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x06. The FCCU error reaction path is verified if both the FCCU_RF_S0[RFS6] and the STCU2_ERR_STAT[UFSF] status bits are set.

The STCU2_ERR_STAT[UFSF] status bit indicates errors that trigger the unrecoverable fault condition, and it is cleared by HW on clearing the relevant FCCU_RF_S0[RFS6] bit.

3.3.2 BIST result-wrong signature (STCU recoverable fault) (fault #7)

If the BIST detects a fault that is configured as recoverable fault, the STCU forwards this fault to the FCCU.

Note: 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.
Although BIST is able to detect permanent faults, this fault can be also triggered in case of transient faults. The STCU can trigger this fault only during BIST execution.

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

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x07. The FCCU error reaction path is verified if both the FCCU_RF_S0[RFS7] and the STCU2_ERR_STAT[RFSF] status bits are set.

The STCU2_ERR_STAT[RFSF] status bit indicates errors that trigger the recoverable faults condition, and it is cleared by HW on clearing the relevant FCCU_RF_S0[RFS7] bit.

3.3.3 MBIST control activation (fault #8)

Unexpected activation of MBIST control signals may move the system RAM array into the BIST mode during the execution of the application. The STCU detects this activation control and forwards this fault to the FCCU.

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

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x08. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS8] status bit is set.

3.4 Glue logic faults

Figure 6. Glue logic fault #9

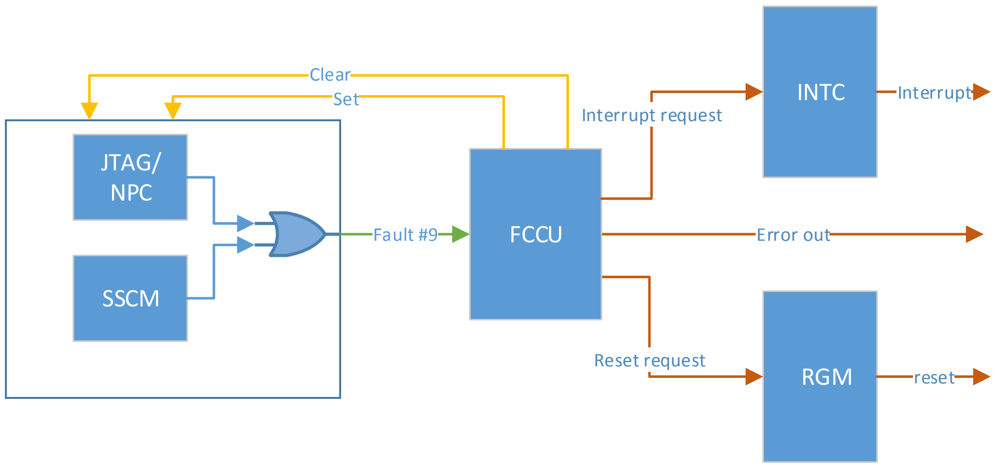
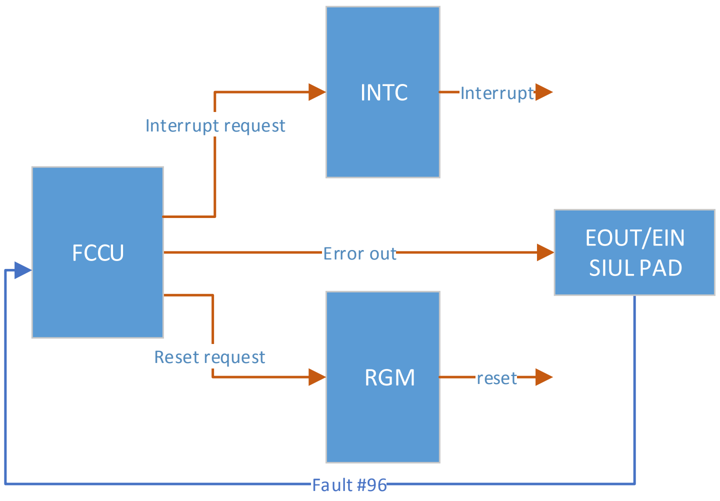


Figure 7. Glue logic fault #96



3.4.1 JTAG-NPC or debug functionality out of reset or SSCM activation (fault #9)

Unexpected activation of JTAG, NPC or debug functionality during the execution of the application can interfere with the application. 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 two error signals are put in OR before arriving at the FCCU failure input #9.

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

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x09. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS9] status bit is set.

3.4.2 Error input pin (from the external world) (fault #96)

If an external device pulls down the EIN (error input) 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.

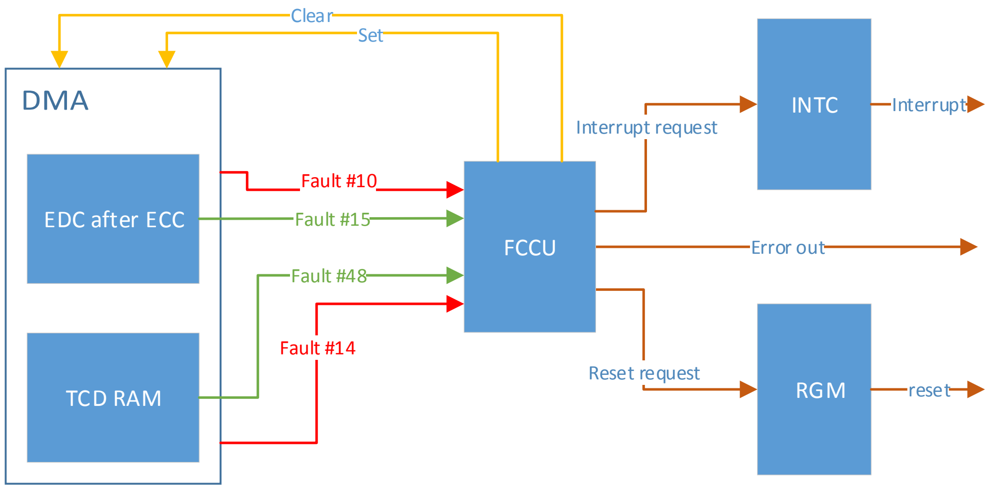
The user can inject this fault by:

1. Enabling EOUT control by FCCU (FCCU_CFG[FCCU_SET_AFTER_RESET] = 0x1);
2. Asserting the EOUT / EIN loopback (SIUL2_MSCR_IO27[SSS] = 0x5);
3. Driving the EOUT to logic 0 (FCCU_CFG [FCCU_SET_CLEAR] = 0x1). Assuming the fault configured as HW recoverable fault, the user can clear the fault by:
4. De-asserting the EOUT / EIN loopback (SIUL2_MSCR_IO27[SSS] = 0x0);
5. Switching back the EOUT control to the FCCU (FCCU_CFG [FCCU_SET_CLEAR] = 0x0);
6. Re-asserting the EOUT / EIN loopback (SIUL2_MSCR_IO27[SSS] = 0x5). The FCCU error reaction path is verified if the FCCU_RF_S3[RFS0] status bit is set after the step (3).

3.5 DMA faults

The eDMA controller is a module capable of performing complex data transfers with minimal intervention of a host processor. For further details on the eDMA, refer to device SPC582Bx reference manual RM0403.

Figure 8. DMA faults



3.5.1 DMA_1 gasket monitor (fault #10)

In case of hardware fault results in a wrong frequency conversion between the XBAR and the DMA_1, the hardware forwards the event to the FCCU.

The user cannot inject this fault.

3.5.2 DMA_1 ECC error (fault #14)

The ECC logic detects single bit error or double bits error and forwards this fault to the FCCU.

The user cannot inject this fault.

3.5.3 DMA_1 TCD EDC after ECC (fault #15)

The EDC after ECC logic detects a hardware fault in the ECC logic resulting in a corrupted ECC correction and forwards this fault to the FCCU.

The user cannot inject this fault.

3.5.4 DMA_1 TCD RAM feedback error (fault #48)

If the latched control signals do not match with the ones originally sent to DMA_1 TCD RAM, the HW feedback checker forwards this fault to the FCCU.

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

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x30. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS16] status bit is set.

3.6 DSMC fault

The DSMC generates the atomic read-modify-write bus transactions to the attached slave memory controller, and it is instantiated within the platform and physically resides between the core data AHB bus and the associated XBAR master port.

3.6.1 DSMC ECC error (fault #10)

The ECC logic detects single bit error or double bits error and forwards this fault to the FCCU.

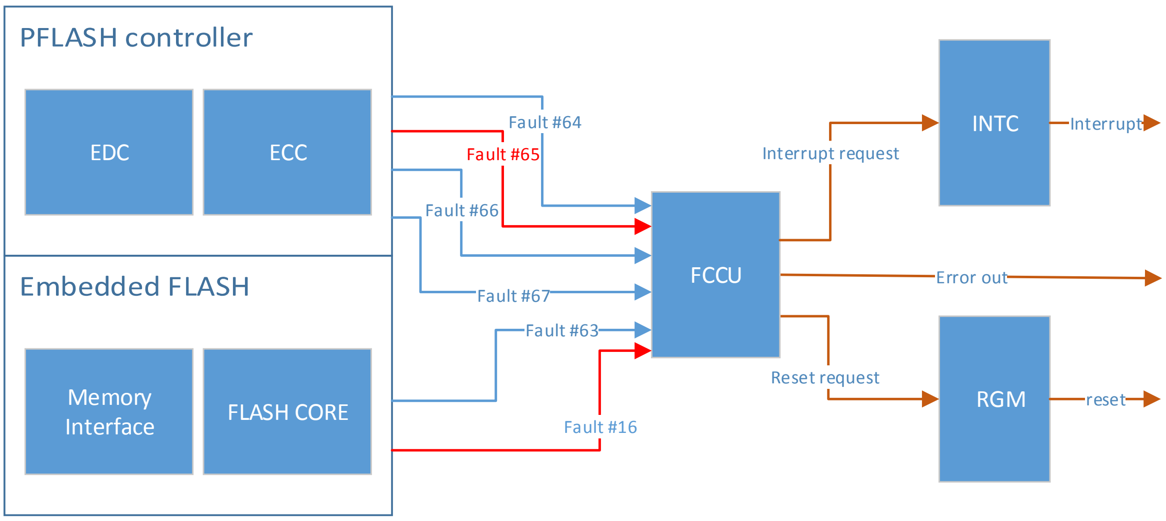
The user cannot inject this fault.

3.7 Flash/PFLASHC faults

The PFLASH controller is the interface between the Flash memory and the XBAR.

The PFLASH controller provides Flash configuration and control functions. For further details on Flash and PFLASHC, refer to the device SPC582Bx reference manual RM0403.

Figure 9. **Flash/PFLASHC faults**



3.7.1 Flash reset error (fault #16)

The Flash forwards this fault to the FCCU in case of 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 (the RAM is not visible to the user) during internal self-check routine (always running).

The user cannot inject this fault.

3.7.2 Flash read reference error (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 fault to the FCCU.

The user can inject this fault by:

1. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
2. Enabling the customer programmable read voltage and reference detection (FLASH_0_UT0[CPR] = 0x1);
3. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
4. Accessing the customer programmable detection area in the UTEST block (address 0x0040_02E0 to 0x0040_02FF). The user can clear the fault by:
5. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
6. Disabling the customer programmable read voltage and reference detection (FLASH_0_UT0[CPR] = 0x0);
7. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
8. Clear the relevant Flash error flag (FLASH_0_MCR[RRE]);
9. Clear the relevant FCCU_RF_S1[RFS31] bit. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS31] status bit is set after step (4).

3.7.3 EDC after ECC for Flash array (fault #64)

The EDC after ECC logic inside the PFLASHC detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for the Flash array and the PFLASHC forwards this fault to the FCCU.

The user can inject this fault by:

1. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
2. Enabling the customer programmable EDC after ECC detection (FLASH_0_UT0[CPE] = 0x1);
3. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
4. Accessing the customer programmable detection area in the UTEST block (address 0x0040_02E0 to 0x0040_02FF). The user can clear the fault by:
5. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
6. Disabling the customer programmable EDC after ECC detection (FLASH_0_UT0[CPE] = 0x0);
7. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
8. Clear the relevant FLASH error flag (FLASH_0_MCR[EEE]);
9. Clear the relevant FCCU_RF_S2[RFS0] bit. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS0] status bit is set after step (4).

3.7.4 EDC after ECC for Flash controller (fault #65)

The EDC after ECC logic inside the PFLASHC detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for the Flash controller and the PFLASHC forwards this fault to the FCCU.

The user cannot inject this fault.

3.7.5 Flash encoding error (fault #66)

The PFLASHC detects faults resulting in a corrupted Flash memory access and it forwards this fault to FCCU.

The user can inject this fault by:

1. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
2. Enabling the customer programmable address encode detection (FLASH_0_UT0[CPA] = 0x1);
3. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
4. Accessing the customer programmable detection area in the UTEST block (address 0x0040_02E0 to 0x0040_02FF). The user can clear the fault by:
5. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
6. Disabling the customer programmable address encode detection (FLASH_0_UT0[CPA] = 0x0);
7. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
8. Clear the relevant FLASH error flag (FLASH_0_MCR[AEE]);
9. Clear the relevant FCCU_RF_S2[RFS2] bit. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS2] status bit is set after step (4).

3.7.6 PFLASH address feedback error (fault #67)

The PFLASHC detects faults resulting in a mismatch between the address from the XBAR and the feedback address from the Flash and it forwards this fault to FCCU.

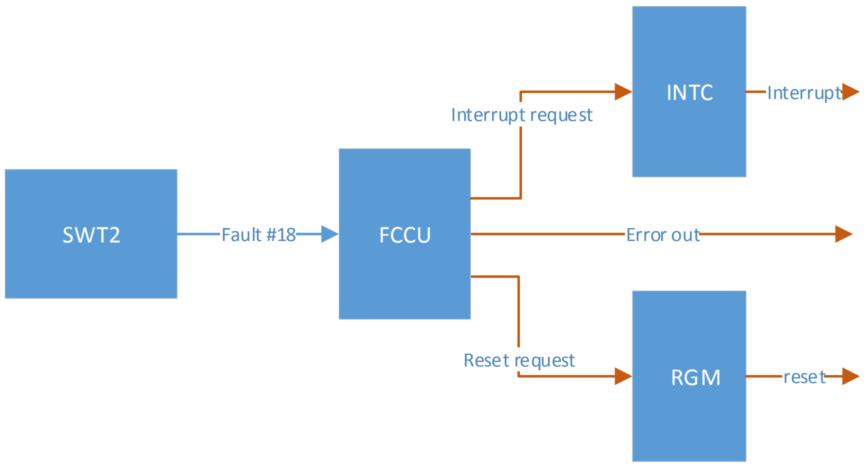
The user can inject this fault by:

1. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
2. Enabling the customer programmable address encode detection (FLASH_0_UT0[CPA] = 0x1);
3. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
4. Accessing the customer programmable detection area in the UTEST block (address 0x0040_02E0 to 0x0040_02FF). The user can clear the fault by:
5. Enabling the user test (FLASH_0_UT0[UTE] = 0x1);
6. Disabling the customer programmable address encode detection (FLASH_0_UT0[CPA] = 0x0);
7. Disabling the user test (FLASH_0_UT0[UTE] = 0x0);
8. Clear the relevant Flash error flag (FLASH_0_MCR[AEE]);
9. Clear the relevant FCCU_RF_S2[RFS3] bit. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS3] status bit is set after step (4).

3.8 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. For further details on SWT, refer to the device SPC582Bx reference manual RM0403.

Figure 10. **SWT faults**



3.8.1 SWT_2 reset request (fault #18)

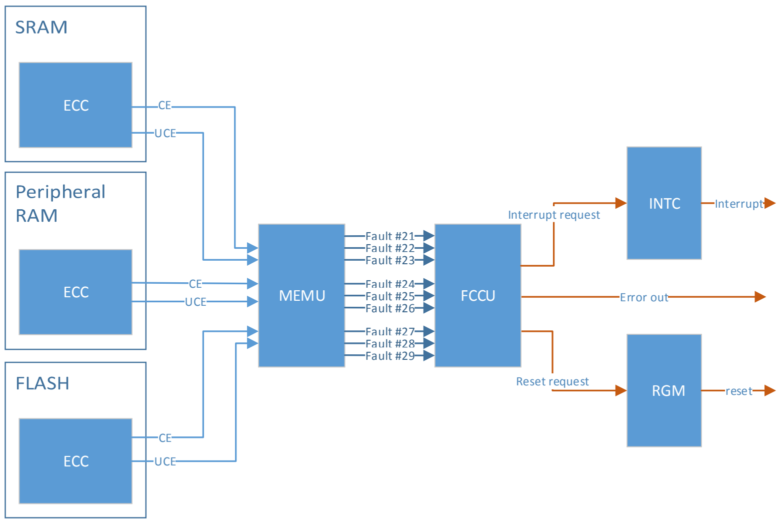
If the SWT_2 reaches a timeout, the SWT_2 forwards this fault to the FCCU.

The user can inject this fault by a SW procedure that enables the SWT2 and does not service it. Note that the user can clear this fault only by triggering a reset. The user can configure the FCCU to trigger a short or a long reset and can read the relevant flag (MC_RGM_FES[FCCU_SOFT] or MC_RGM_FES[FCCU_HARD]) to check if the FCCU has correctly triggered a reset on SWT_2 timeout. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS18] status bit is set after SWT_2 timeout.

3.9 MEMU faults

The MEMU is responsible for collecting and reporting error events captured by ECC/EDC logic used in system RAM, peripheral RAM and Flash memories. When any of the following events occurs, the MEMU receives an error signal that causes an event to be recorded. When multiple errors are indicated from various sources at the same instant, an overflow can be indicated by the MEMU to the FCCU. Overflow can also be indicated if the reporting table entries are full, and a new unique error is reported by the system. The corresponding error flags are set and reported to FCCU. The MEMU does not receive any error signal on ECC events occurring while accessing to the EEPROM sections of the Flash memory. For further details on the MEMU, refer to the device SPC582Bx reference manual RM0403.

Figure 11. **MEMU faults**



3.9.1 System RAM correctable error (fault #21)

In case a correctable error is detected when accessing to the system RAM, the MEMU records the event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_SR_CE] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS21] status bit is set. The user must clear the MEMU_DEBUG[FR_SR_CE] bit before clearing the relevant FCCU_RF_S0[RFS21] bit. Alternatively, the user can use the IMA to inject a correctable error in the system RAM location and cause the detection of this correctable error accessing to this system RAM location by the core. For further details on the IMA, refer to the device SPC582Bx reference manual RM0403

3.9.2 System RAM uncorrectable error (fault #22)

In case an uncorrectable error is detected when accessing to the system RAM, the MEMU records the event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_SR_UCE] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS22] status bit is set. The user must clear the MEMU_DEBUG[FR_SR_UCE] bit before clearing the relevant FCCU_RF_S0[RFS22] bit. Alternatively, the user can use the IMA to inject an uncorrectable error in the system RAM location and cause the detection of this uncorrectable error accessing to this system RAM location by the core. For further details on the IMA, refer to the device SPC582Bx reference manual RM0403.

3.9.3 System RAM overflow error (fault #23)

In case of system RAM overflow, errors reporting table, the MEMU forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_SR_CEO] bit or the MEMU_DEBUG[FR_SR_UCO] bit or the MEMU_DEBUG[FR_SR_EBO] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS23] status bit is set. The user must clear the relevant bit (MEMU_DEBUG[FR_SR_CEO] or MEMU_DEBUG[FR_SR_UCO] or MEMU_DEBUG[FR_SR_EBO]) before clearing the relevant FCCU_RF_S0[RFS23] bit.

3.9.4 Peripheral RAM correctable error (fault #24)

In case a correctable error is detected when accessing a peripheral RAM, the MEMU records the event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_PR_CE] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS24] status bit is set. The user must clear the MEMU_DEBUG[FR_PR_CE] bit before clearing the relevant FCCU_RF_S0[RFS24] bit. Alternatively, the user can use the IMA to inject a correctable error in a peripheral RAM location and cause the detection of this correctable error accessing to this peripheral RAM location by the core. For further details on the IMA, refer to the device SPC582Bx reference manual RM0403.

3.9.5 Peripheral RAM uncorrectable error (fault #25)

In case an uncorrectable error is detected when accessing a peripheral RAM, the MEMU records the event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_PR_UCE] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS25] status bit is set. The user must clear the MEMU_DEBUG[FR_PR_UCE] bit before clearing the relevant FCCU_RF_S0[RFS25] bit. Alternatively, the user can use the IMA to inject an uncorrectable error in a peripheral RAM location and cause the detection of this uncorrectable error accessing to this peripheral RAM location by the core. For further details on the IMA, refer to the device SPC582Bx reference manual RM0403.

3.9.6 Peripheral RAM overflow (fault #26)

In case a peripheral RAM errors reporting table overflow, the MEMU forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_PR_CEO] bit or the MEMU_DEBUG[FR_PR_UCO] bit or the MEMU_DEBUG[FR_PR_EBO] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS26] status bit is set. The user must clear the relevant bit (MEMU_DEBUG[FR_PR_CEO] or MEMU_DEBUG[FR_PR_UCO] or MEMU_DEBUG[FR_PR_EBO]) before clearing the relevant FCCU_RF_S0[RFS26] bit.

3.9.7 Flash correctable error (fault #27)

In case a correctable error is detected when accessing to the Flash, the MEMU records the event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_F_CE] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS27] status bit is set. The user must clear the MEMU_DEBUG[FR_F_CE] bit before clearing the relevant FCCU_RF_S0[RFS27] bit.

3.9.8 Flash uncorrectable error (fault #28)

In case an uncorrectable error is detected when accessing to the Flash, the MEMU records the event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_F_UCE] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS28] status bit is set. The user must clear the MEMU_DEBUG[FR_F_UCE] bit before clearing the relevant FCCU_RF_S0[RFS28] bit.

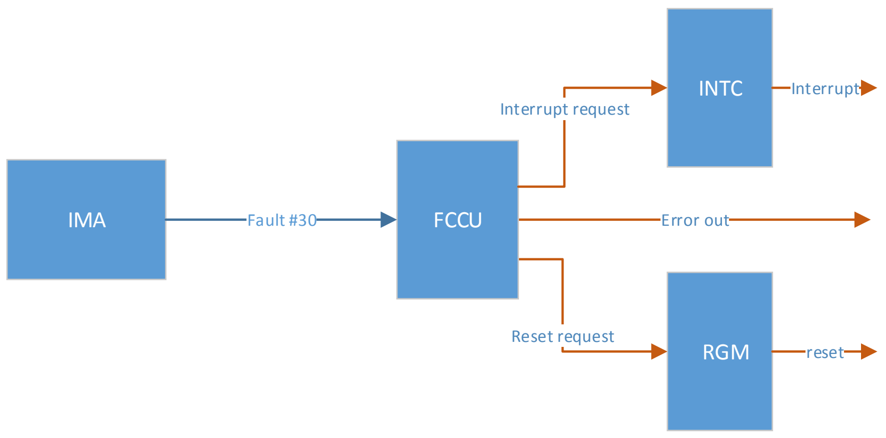
3.9.9 Flash overflow error (fault #29)

In case of the Flash errors reporting table overflow, the MEMU forwards this fault to the FCCU. The user can inject this fault by a SW procedure that sets the MEMU_DEBUG[FR_F_CEO] bit or the MEMU_DEBUG[FR_F_UCO] bit or the MEMU_DEBUG[FR_F_EBO] bit. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS29] status bit is set. The user must clear the relevant bit (MEMU_DEBUG[FR_F_CEO] or MEMU_DEBUG[FR_F_UCO] or MEMU_DEBUG[FR_F_EBO]) before clearing the relevant FCCU_RF_S0[RFS29] bit.

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, for example, verifying the integrity of the ECC logic. For further details on the IMA, refer to the SPC582Bx reference manual RM0403.

Figure 12. IMA fault



3.10.1 IMA activation (fault #30)

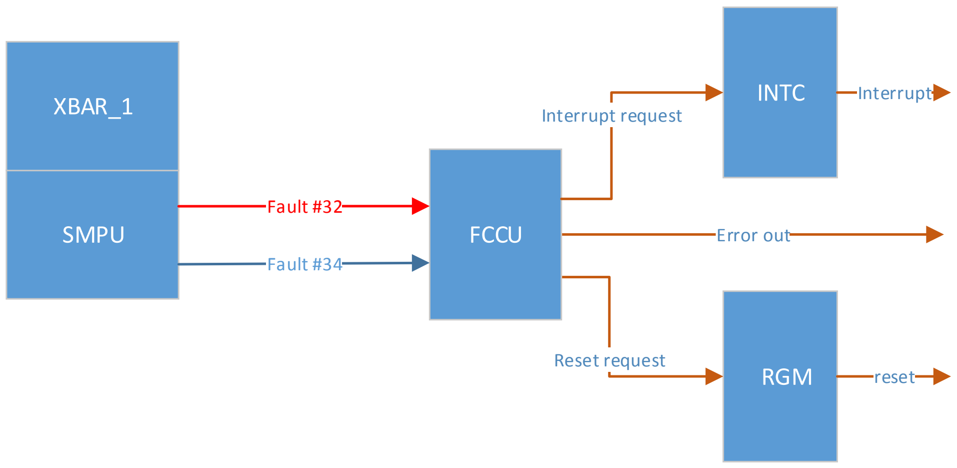
Since unwanted activation of the IMA can interfere with execution of the application, the IMA signals to the FCCU when its activation has happened. As a result, the FCCU can react to an unwanted activation of IMA according to its configuration and, before any intentional activation of the IMA, the user shall disable the relevant FCCU input.

The user can inject this fault by a SW procedure activating the IMA without disabling the relevant FCCU input. The IMA is activated setting a proper value for the IMA_SLCT[ARRAY_SLCT] field. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS30] status bit is set. The user must deactivate the IMA (IMA_SLCT[ARRAY_SLCT] = 0x00) before clearing the relevant FCCU_RF_S0[RFS30] bit.

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. A memory access that has sufficient access control rights is allowed to complete, while a memory access that is not mapped to any region descriptor or has insufficient rights terminates with an access error response. For further details on the SMPU, refer to the device SPC582Bx reference manual RM0403.

Figure 13. SMPU faults



3.11.1 SMPU XBAR 1 monitor incorrectly refuses an access (fault #32)

In case the SMPU denies an access to a mapped memory location with sufficient rights, the HW monitors inside the SMPU detects this event and forwards this fault to the FCCU. The user cannot inject this fault.

3.11.2 SMPU XBAR 1 monitor correctly refuses an access (fault #34)

In case of a memory access not mapped to any region descriptor or with insufficient rights, it terminates with an access error response and the HW monitors inside the SMPU detects this event and forwards this fault to the FCCU. The user can inject this fault by a SW procedure that accesses a memory address not allowed by the SMPU. A data storage interrupt (IVOR2) must be handled. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS2] status bit is set.

3.12 Core_2 faults

3.12.1 Core_2 I-bus ECC error (fault #12)

Instruction and data busses between XBAR and the Core_2 are protected by e2e_ECC.

In case an ECC error is detected on Core_2 instruction bus, the e2e_ECC forwards this fault to the FCCU.

The user cannot inject this fault.

3.12.2 Core_2 D-bus ECC error (fault #13)

Instruction and data busses between XBAR and the Core_2 are protected by e2e_ECC.

In case of an ECC error is detected on Core_2 data bus, the e2e_ECC forwards this fault to the FCCU.

The user cannot inject this fault.

3.12.3 Safety Core_2 exception (machine check exception) (fault #84)

When the Core_2 runs into a machine check condition, the Core_2 forwards this fault to the FCCU.

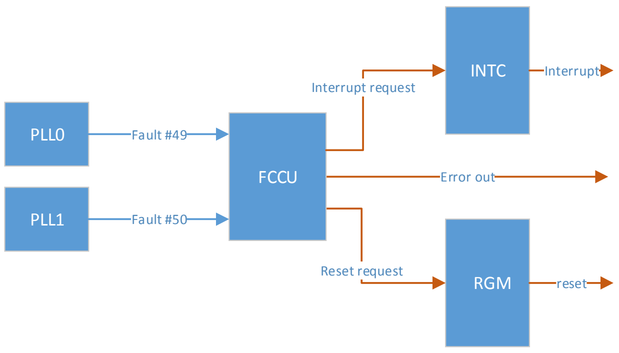
The user can inject this fault by a SW procedure that accesses to a not existing address. The machine check interrupt (IVOR1) must be handled. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS20] status bit is set.

3.13 PLLDIG faults

The SPC582Bx embeds a dual PLL system which provides separate system and peripheral clocks.

For further details on dual PLL, refer to the device SPC582Bx reference manual RM0403.

Figure 14. PLL DIG faults



3.13.1 PLL0 loss of lock error (fault #49)

A built-in mechanism can detect a loss of lock for the PLL0. The relevant PLLDIG forwards this fault to the FCCU. The user can inject this fault by a SW procedure that enables the loss of lock interrupt (PLLDIG_PLL0CR[LOLIE] = 1) and changes on-the-fly the PLL configuration (for example, change on-the-fly the value of the PLLDIG_PLL0DV[PREDIV] field) that generates a temporary loss of lock. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS17] and the PLLDIG_PLL0SR[LOLF] status bits are set. The user must restore on-the-fly the PLL configuration, wait for the new lock and clear PLLDIG_PLL0SR[LOLF] status bit before clearing the relevant FCCU_RF_S1[RFS17] bit.

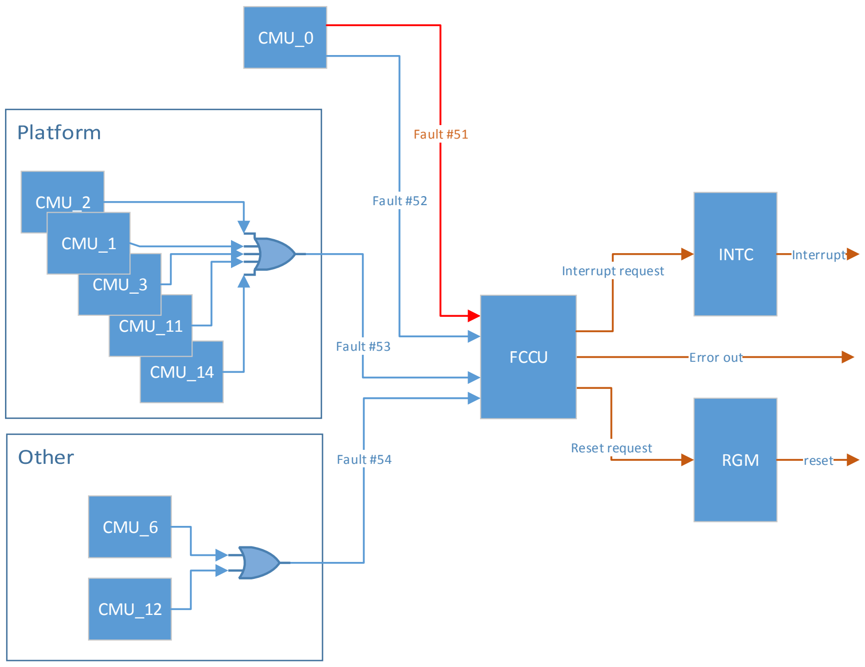
3.13.2 PLL1 loss of lock error (fault #50)

A built-in mechanism can detect a loss of lock for the PLL1. The relevant PLLDIG forwards this fault to the FCCU. The user can inject this fault by a SW procedure that enables the loss of lock interrupt (PLLDIG_PLL1CR[LOLIE] = 1) and changes on-the-fly the PLL configuration (for example, change on-the-fly the value of the PLLDIG_PLL1DV[PREDIV] field) that generates a temporary loss of lock. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS18] and the PLLDIG_PLL1SR[LOLF] status bits are set. The user must restore on-the-fly the PLL configuration, wait for the new lock and clear PLLDIG_PLL1SR[LOLF] status bit before clearing the relevant FCCU_RF_S1[RFS18] bit.

3.14 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 (the user must enable the CMU that is disabled by default) these faults to the FCCU. For further details on the CMUs, refer to SPC582Bx reference manual RM0403.

Figure 15. CMU faults



3.14.1 CMU_0 error (fault #51)

The CMU_0 (XOSC less than IRC, XTAL pin floating, EXTAL pin floating) monitors the XOSC frequency. If the XOSC frequency is less than the IRC frequency (or one of the crystal oscillator pins is not connected), the CMU_0 can detect this event and forwards it to the FCCU.

The user cannot inject this fault.

3.14.2 Frequency out of range (fault #52)

The CMU_0 monitors the PHI output frequency of PLL_0 using the IRCOSC frequency as monitor references. If the PLL_0 output frequency is above or below the monitoring thresholds, the CMU_0 detects this fault and forwards it to the FCCU. Moreover, only the CMU0 implements the XOSC monitor to calibrate the IRCOSC frequency. The user can inject this fault by a SW procedure that sets a misconfigured value for one of the monitoring thresholds, for example, the user can set the CMU_0_ HFREFR[HFREF] field to a value lower than the correct one. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS20] and the CMU_0_ISR[FHHI] status bits are set. The user must clearing CMU_0_ISR[FHHI] status bit before clearing the relevant FCCU_RF_S1[RFS20] bit.

3.14.3 Frequency out of range (fault #53)

Using the IRCOSC frequency as monitor references, the CMU_1 monitors the clock frequency used by Core_2 and XBAR, the CMU_2 monitors the clock frequency used by HPBM, the CMU_3 monitors the clock frequency used by the PBRIDGE, the CMU_11 monitors the clock frequency used by the FBRIDGE and the CMU_14 monitors the clock frequency used by the PFBRIDGE. If one of the monitored frequencies is above or below the relevant monitoring thresholds, the relevant CMU detects this fault and forwards it to the FCCU. The FCCU receives the signal in OR from these CMU modules. The user can inject this fault by a SW procedure that sets a misconfigured value for one of the monitoring thresholds, for example, the user can set the CMU_x_HFREFR[HFREF] field (with x = 1, 2, 3, 11, 14) to a value lower than the correct one. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS21] and the CMU_x_ISR[FHHI] status bits (with x = 1, 2, 3, 11, 14) are set. The user must clear CMU_x_ISR[FHHI] status bit (with x = 1, 2, 3, 11, 14) before clearing the relevant FCCU_RF_S1[RFS21] bit.

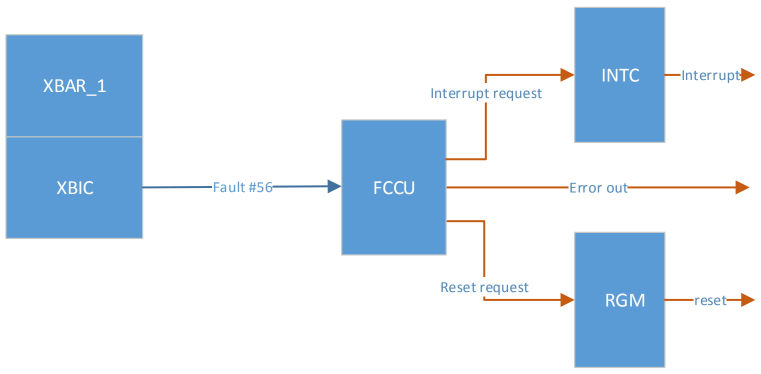
3.14.4 Frequency out of range (fault #54)

Using the IRCOSC frequency as monitor references, the CMU_6 monitors the clock frequency used by the SARADCs and the CMU_12 monitors the clock frequency used by the eMIOS. If one of the monitored frequencies is above or below the relevant monitoring thresholds, the relevant CMU detects this fault and forwards it to the FCCU. The FCCU receives the signal in OR from these CMU modules.The user can inject this fault by a SW procedure that sets a misconfigured value for one of the monitoring thresholds, for example, the user can set the CMU_y_HFREFR[HFREF] field (with y = 6, 12) to a value lower than the correct one. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS22] and the CMU_y_ISR[FHHI] status bits (with y = 6, 12) are set. The user must clear CMU_y_ISR[FHHI] status bit (with y = 6, 12) before clearing the relevant FCCU_RF_S1[RFS22] bit.

3.15 XBIC fault

The XBIC verifies the integrity of each XBAR transfer. The XBAR transfer attribute information for all master and slave ports is routed to the XBIC that verifies the coherence between input and output signals of the XBAR. For further details on the XBIC, refer to SPC582Bx microcontroller reference manual RM0403.

Figure 16. **XBIC fault**



3.15.1 XBIC error detected (fault #56)

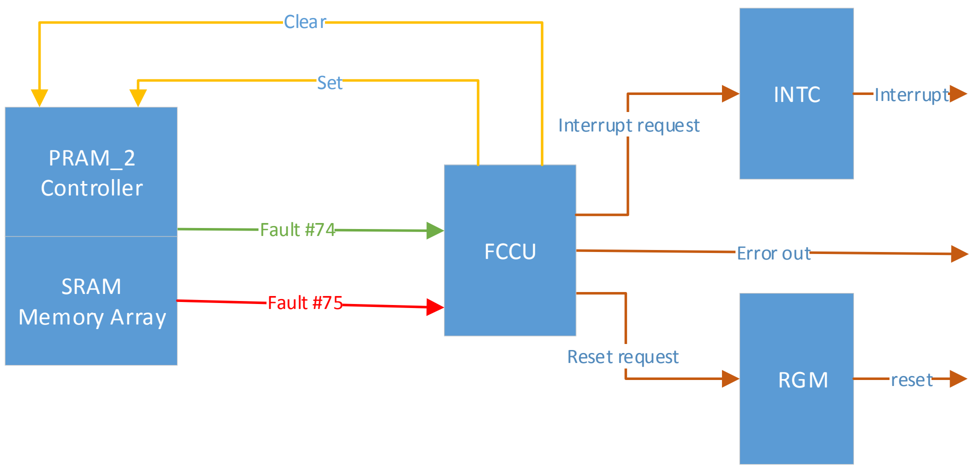
In case of corrupted transaction through the XBAR, the XBIC detects this event and forwards it to the FCCU.

The user can inject this fault by a SW procedure that uses the XBIC error injection feature through the XBIC_EIR register. The user can select the system RAM as target (XBIC_1_EIR[SLV] = 2) and the Core_2 as master (XBIC_1_EIR[MST] = 2), can set the syndrome (for example, setting XBIC_1_EIR[SYN] = 0x80, the bit 7 is modified to inject the fault), can enable the fault injection (XBIC_1_EIR[EIE] = 1) and can access to the system RAM by the Core_2. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS24] status bit is set after the system RAM access. The user must clear the XBIC_EIR register before clearing the relevant FCCU_RF_S1[RFS24] bit.

3.16 PRAM_2 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 one controller, the PRAMC_2. For further details on the PRAM controller, refer to the SPC582Bx microcontroller reference manual RM0403.

Figure 17. **PRAM faults**



3.16.1 Corrupted system RAM access or late-write buffer mismatch (fault #74).

In case of addressing error (for example, a write error in the RAM controller resulting in corrupted RAM access), the PRAM controller detects this fault, and it forwards it to the FCCU.

The user cannot inject this fault.

3.16.2 EDC after ECC for system RAM (fault #75).

The EDC after ECC detects faults affecting the ECC logic of the PRAM controller and forwards it to the FCCU.

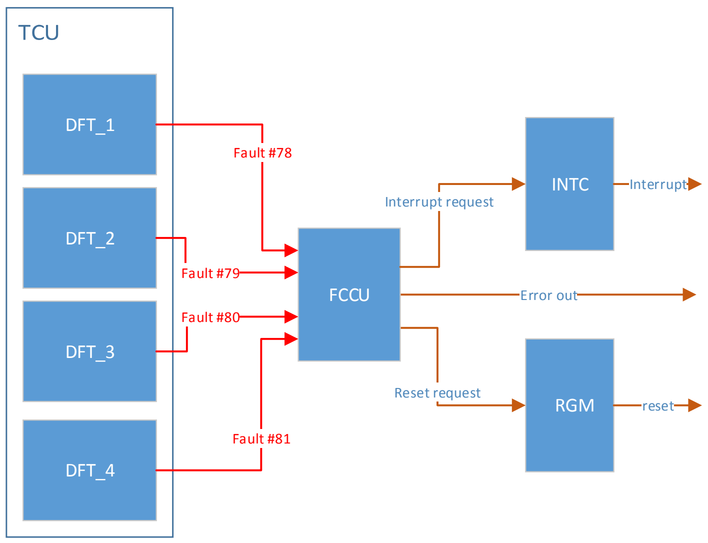
The user cannot inject this fault.

3.17 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 application runs. For this reason, if a random event asserts one of these signals the fault is detected and forwarded to the FCCU.

A different FCCU channel monitors each diagnostic function test domain.

Figure 18. TCU faults



3.17.1 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 cannot inject this fault.

3.17.2 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 cannot inject this fault.

3.17.3 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 cannot inject this fault.

3.17.4 Test circuitry group 4 activation (fault #81)

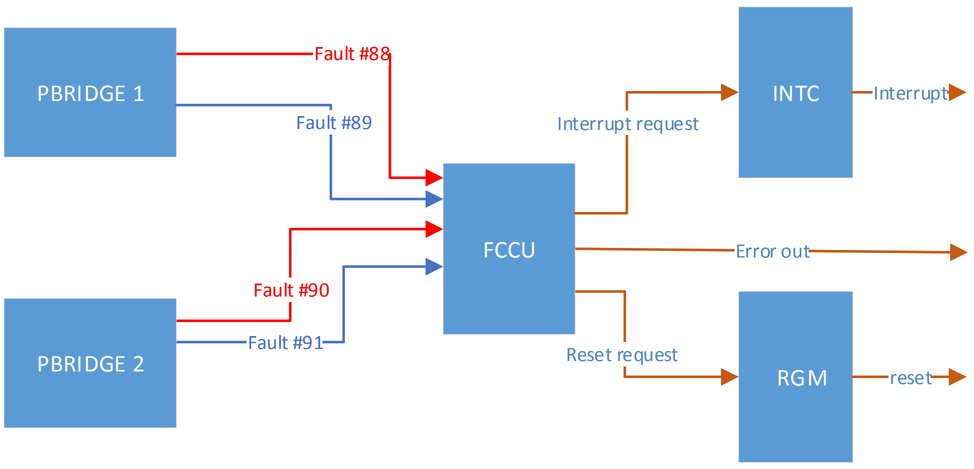
In case of unwanted activation of the test circuitry in the related diagnostic function test domain, the event is detected and forwarded to the FCCU.

The user cannot inject this fault.

3.18 PBRIDGE faults

The PBRIDGE connects the XBAR interface to the register interface of the peripherals embedded within the device. The device incorporates two peripheral bridges: PBRIDGE_1 and PBRIDGE_2. For further details on the PRIDGEs, refer to the SPC582Bx microcontroller reference manual RM0403.

Figure 19. PBRIDGE faults



3.18.1 PBRIDGE_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 detect this fault and forward it to the FCCU.

The user cannot inject this fault.

3.18.2 PBRIDGE_1 e2eEDC error (fault #91)

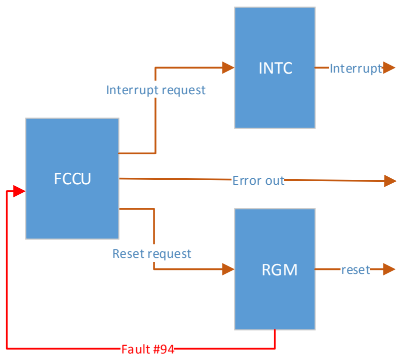
A random failure affecting the ECC correction logic of the corresponding master can cause a corrupted ECC correction. The EDC after ECC can detect this fault and forward it to the FCCU.

The user cannot inject this fault.

3.19 MC_RGM fault

The MC_RGM centralizes the different reset sources and manages the reset sequence of the chip. For further details on the MC_RGM, refer to the SPC582Bx reference manual RM0403.

Figure 20. MC_RGM fault



3.19.1 Safe mode entry indication (fault #94)

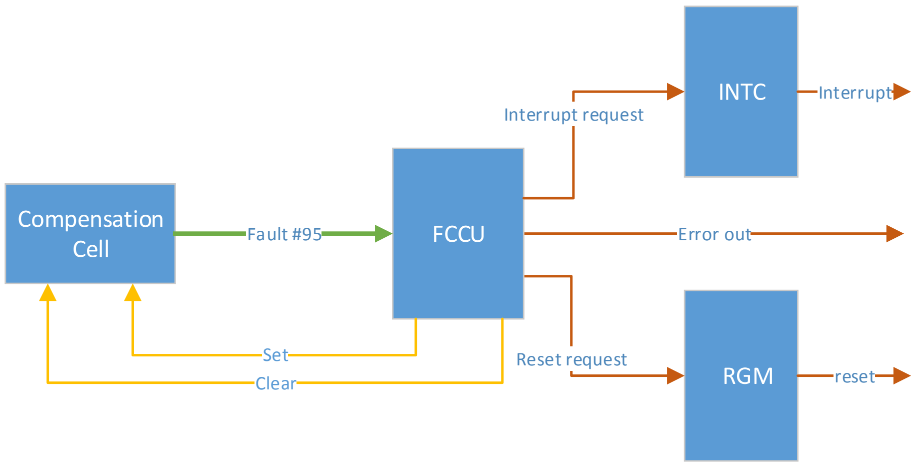
The MC_RGM can request a transition to safe mode to the MC_ME. In case of an unwanted safe mode request due to a random event, the hardware detects this event and forwards it to the FCCU.

The user cannot inject this fault.

3.20 Compensation cells faults

Compensation cells generate an 8-bit compensation code for I/O buffers, depending on process, voltage, and temperature (PVT) conditions of the chip. Compensation reduces the spread of some circuit parameters (for example, current slew rate and output impedance) in the I/O buffers over temperature, pressure and voltage.

Figure 21. Compensation cell fault



3.20.1 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 cannot inject this fault.

4 Example code

An example code that includes the FCCU settings and how to inject the 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 IPs);
- Reset the RGM and clear its registers;
- Initialize the FCCU for all the testable faults:
  - Enabled the FCCU input
  - Set the FCCU input as SW recoverable
  - No reset action (except for SWT_2 input)
  - Enable Interrupt with timeout (FCCU state machine goes to alarm state in case of fault)
  - Enable output pins
- For each fault identified as "testable" the software:
  - Verify the FCCU status before injection
    - If it is in normal state, proceed, otherwise, if it is in alarm or fault state, stop
  - Inject it (using the registers of the monitor or using fake fault injection or a SW procedure, if possible)
  - Verify the FCCU status after injection
    - If it is in alarm state, proceed, otherwise, if it is in normal or fault state, stop
    - Check FCCU reaction (IRQ and relevant FCCU fault flag)
    - Clear the monitor fault and the FCCU alarm state
    - Verify the FCCU status after recovering from alarm
      - If it is in normal state, proceed, otherwise, if it is in alarm or fault state, stop

5 Summary

Safety analysis requires that the user verifies the integrity of the FCCU error reaction path (not all FCCU inputs are testable) periodically with a period lower than the trip time (for example, 12 hours). The methodology for these tests is based on fault injection and verification whether the FCCU correctly receives it and depends on the specific FCCU input.

This document, with reference to SPC582Bx devices, describes the FCCU faults inputs and how to verify their reaction path.

Appendix A Acronyms, abbreviations and reference documents

Table 2. Acronyms and abbreviations

Terms	Description
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
eMIOS	Enhanced modular input-output system
FCCU	Fault Collection and Control Unit
FOSU	FCCU output supervision unit
HVD	High-voltage detector
IMA	Indirect memory access
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
MCU	Microcontroller Unit
NMI	Non-maskable interrupt
NPC	Nexus debug port
PBRIDGE	Peripheral bridge
PFLASHC	Platform Flash controller
PLL	Phase-Locked 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
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

Terms	Description
XBAR	CrossBAR
XBIC	CrossBAR integrity checker
XOSC	External oscillator/crystal

Table 3. **Reference documents**

Document name	Document title
RM0403	SPC58 2B Line - 32 bit Power Architecture automotive MCU z2 core 80 MHz, 1 MByte Flash, ASIL-B
ES0413	SPC582Bx devices errata JTAG_ID = 0x1114_0041

Revision history

Table 4. **Document revision history**

Date	Revision	Changes
26-Nov-2021	1	Initial release.

Contents

List of tables

List of figures