AN6042

Application note

SR5 E1 line – FCCU fault sources and reaction

Introduction

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

Before reading this document, the reader should have a clear understanding about the usage of FCCU module itself. Refer to the SR5E1 microcontroller reference manual for further details on each module (see the Appendix A: Reference documents). A reference code is available.

This application note applies to the devices listed in the following table.

Table 1. **Device summary**

Series	Part number
SR5E1x	SR5E1E3, SR5E1E5, SR5E1E7

1 General information

This document applies to Arm® - based devices of SR5 E1 line, Stellar electrification MCUs - 32-bit Arm® Cortex® M7 architecture microcontroller for electrical vehicle applications.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

1.1 Acronyms

Table 2. **Acronyms**

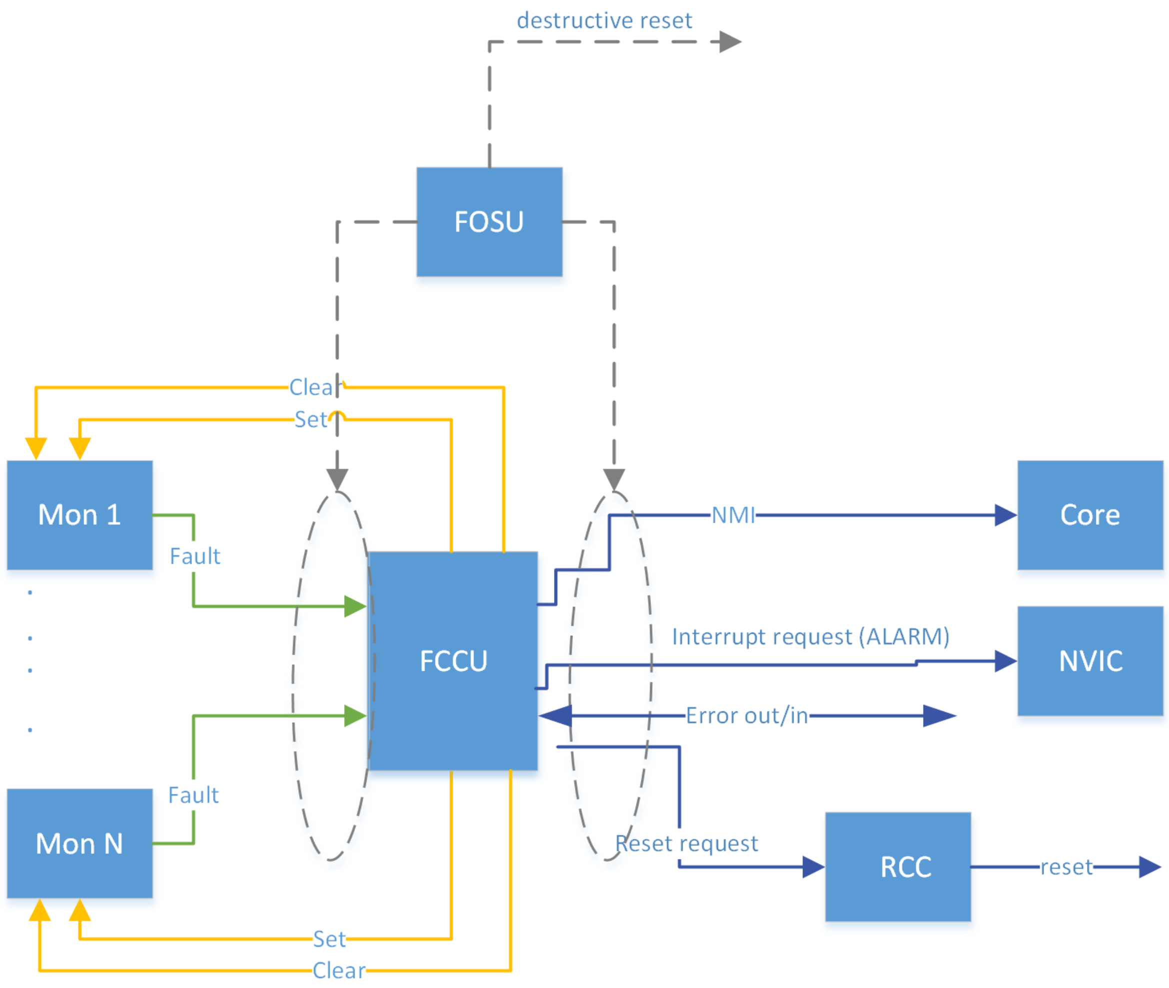
Acronym	Name
AHB	Advanced High-performance Bus
AXI	Advance Extensible Interface
BIST	Built In Self-Test
CBIST	Comparator Built-In Self Test
CEM	Combined Error Management
CMU	Clock Monitoring Unit
CPU	Central Processing Unit
DCF	Device Configuration Format
DMA	Direct Memory Access
EDC/ECC	Error Detection Code/Error Correction Code
EOUT	Error Out
FCCU	Fault Collection and Control Unit
FOSU	FCCU output supervision unit
HVD	High Voltage Detector
IMA	Indirect Memory Access
IRCOSC	Internal 16 MHz RC oscillator
IRQ	Interrupt Request
IP	Intellectual Propriety
JTAG	Joint Test Action Group
LBIST	Logic Built-in self-test
LVD	Low Voltage Detector
MCU	Microcontroller Unit
NMI	Non-maskable interrupts
NVM	Non-volatile Memory
NVIC	Nested vectored interrupt controller
NPC	Nexus debug port
OTA	Over The Air
PBRIDGE	Peripheral Bridge
PFLASHC	Platform FLASH Controller
PLL	Phase Lock Loop
PMC	Power Management Control
PMC_DIG	Power Management Controller Digital Interface
PRAM	Platform RAM Controller

Acronym	Name
POR	Power On Reset
RCC	Reset and Clock Control Module
RCCU	Redundancy Control Checker Unit
RM	Reference Manual
SMPU	System Memory Protection Unit
SoC	System On Chip
SRAM	System RAM
SSCM	System status and configuration module
STCU3	Self-Test Control Unit
TCU	Test Control Unit
XBAR	CrossBAR
XBIC	CrossBAR integrity checker
XOSC	External oscillator/crystal

2 Overview

The FCCU is a key element of the functional safety concept of SR5 E1 line devices. This module is responsible for collecting and reacting to failure notifications coming from different modules indicated as "monitors". Examples of monitors are CMU, MEMU2 and so forth.

Figure 1. **FCCU monitor to reaction path**



The Figure 1 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.

If a fault occurs the FCCU can move the device into the safe state (the safety manual defines the safe states, requirement SM_MCU_1_9) without any CORE intervention.

Note: *Since the FCCU and the whole error reaction path are prone to latent failures, the safety manual requires the execution of a software test to verify the integrity of the error reaction path (requirement SM_MCU_3_31). The user shall run this software test at least once per trip time (the safety analysis assumes a trip time of 12 hours).*

This document goes through the list of the faults reported to the FCCU. For each of them it describes how to test the error reaction path to fulfill the previous requirement. Note that the user cannot test the error reaction path for certain monitors (refer to "FCCU fake fault" in the Figure 2).

The Table 3 lists and describes all FCCU input fault sources present on SR5 E1 line MCUs. It provides the recommended method for injecting each fault and for determinating the verification and feasibility of the error reaction path.

Table 3. **FCCU failure inputs**

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
0	-	PMC DIG	Temperature detector	FCCU fake fault injection(1)	NO
1	-	PMC DIG	Voltage out of range from LVDs (non-destructive reset)	FCCU fake fault injection(1)	NO
2	-	PMC DIG	Voltage out of range from LVDs (non-destructive reset)	FCCU fake fault injection(1)	NO
3	-	PMC DIG	Digital PMC DCF safety error	FCCU fake fault injection(1)	NO
4	-	PMC DIG	Digital PMC voltage detector BIST	FCCU fake fault injection(1)	YES
5	-	Flash	Flash memory initialization error	FCCU fake fault injection(1)	NO
6	-	Flash	Flash reset error	FCCU fake fault injection(1)	NO
7	-	Flash	FLASH read reference error	FCCU fake fault injection(1)	NO
8	-	IWDG1	Independent WDG1 reset request	Software procedure(2)	YES
9	-	IWDG2	Independent WDG2 reset request	Software procedure(2)	YES
10	-	WWDG1	Window watchdog1 reset request	Software procedure(2)	YES
11	-	WWDG2	Window watchdog2 reset request	Software procedure(2)	YES
12	-	PLL DIG	PLL0 Loss of Lock (Interrupt)	Software procedure(2)	YES
13	-	PLL DIG	PLL1 Loss of Lock (Interrupt)	Software procedure(2)	YES
14	-	CMU	XOSC less than IRC	FCCU fake fault injection(1)	NO
15	-	CMU	PLL0 out of frequency	Software procedure(2)	YES
16	-	CMU	Sysclk frequency out of range (including HRTIM clock)	Software procedure(2)	YES
17	-	CMU	Monitoring other internal clocks	Software procedure(2)	YES
18	-	-	-	-	-
19	-	STCU3	BIST result - wrong signature (STCU recoverable fault)	FCCU fake fault injection(1)	YES
20	-	-	-	-	-
21	-	MEMU2	The fault TRIG_0 from SYS_RAM table	Software procedure(2)	YES
22	-	MEMU2	The fault TRIG_1 from SYS_RAM table	Software procedure(2)	YES
23	-	MEMU2	The fault TRIG_2 from SYS_RAM table	Software procedure(2)	YES

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
24	-	MEMU2	The fault TRIG_3 from SYS_RAM table	Software procedure[(2)]	YES
25	-	MEMU2	The fault TRIG_0 from PERIPH_RAM table	Software procedure[(2)]	YES
26	-	MEMU2	The fault TRIG_1 from PERIPH_RAM table	Software procedure[(2)]	YES
27	-	MEMU2	The fault TRIG_2 from PERIPH_RAM table	Software procedure[(2)]	YES
28	-	MEMU2	The fault TRIG_3 from PERIPH_RAM table	Software procedure[(2)]	YES
29	-	MEMU2	The fault TRIG_0 from NVM_RAM table	Software procedure[(2)]	YES
30	-	MEMU2	The fault TRIG_1 from NVM_RAM table	Software procedure[(2)]	YES
31	-	MEMU2	The fault TRIG_2 from NVM_RAM table	Software procedure[(2)]	YES
32	-	MEMU2	The fault TRIG_3 from NVM_RAM table	Software procedure[(2)]	YES
33	CEM_10	MEMU2	Sys RAM single bit error table overflow	CEM software procedure[(3)]	YES
			Sys RAM uncorrectable error table overflow		
			Periph RAM single bit error table overflow		
			Periph uncorrectable error table overflow		
			NVM single bit error table overflow		
			NVM uncorrectable error table overflow		
			NVM double correctable table overflow		
34	CEM_11	SYS RAM FIFOs to MEMU2 overflow	SYS_RAM_OVRFLOW_FIF0_0	CEM software procedure[(3)]	NO
			SYS_RAM_OVRFLOW_FIF0_1		
			SYS_RAM_OVRFLOW_FIF0_2		
			SYS_RAM_OVRFLOW_FIF0_3		
			SYS_RAM_OVRFLOW_FIF0_4		
			SYS_RAM_OVRFLOW_FIF0_5		
			SYS_RAM_OVRFLOW_FIF0_6		
			SYS_RAM_OVRFLOW_FIF0_7		
			SYS_RAM_OVRFLOW_FIF0_8		
			SYS_RAM_OVRFLOW_FIF0_32		
			SYS_RAM_OVRFLOW_FIF0_33		
			SYS_RAM_OVRFLOW_FIF0_34		
			SYS_RAM_OVRFLOW_FIF0_35		
			SYS_RAM_OVRFLOW_FIF0_36		
			SYS_RAM_OVRFLOW_FIF0_37		
			SYS_RAM_OVRFLOW_FIF0_38		
			SYS_RAM_OVRFLOW_FIF0_39		

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
34	CEM_11	SYS RAM FIFOs to MEMU2 overflow	SYS_RAM_OVRFLOW_FIF0_40	CEM software procedure[3]	NO
			SYS_RAM_OVRFLOW_FIF0_41		
			SYS_RAM_OVRFLOW_FIF0_42		
			SYS_RAM_OVRFLOW_FIF0_43		
			SYS_RAM_OVRFLOW_FIF0_44		
35	CEM_12	PERIPH RAM FIFOs to MEMU2 overflow	PERIPH_RAM_OVRFLOW_FI F0_0	CEM software procedure[3]	NO
			PERIPH_RAM_OVRFLOW_FI F0_1		
			PERIPH_RAM_OVRFLOW_FI F0_2		
			PERIPH_RAM_OVRFLOW_FI F0_32		
			PERIPH_RAM_OVRFLOW_FI F0_33		
			PERIPH_RAM_OVRFLOW_FI F0_34		
			PERIPH_RAM_OVRFLOW_FI F0_35		
			PERIPH_RAM_OVRFLOW_FI F0_36		
			PERIPH_RAM_OVRFLOW_FI F0_37		
36	-	MEMU2	MEMU2 FLASH FIFO to MEMU2 overflow	FCCU fake fault injection[1]	NO
37	CEM_9	Boot errors	SSCM transfer error	CEM software procedure[3]	NO
			Memory repair safety error		
			TDM DCF safety error		
			RCC DCFs + security miscellaneous DCF		
38	-	ERRIN1	Error from unidirectional input error signal (External failure to MCU)	Software procedure[2]	YES
39	-	IMA	IMA SoC active	Software procedure[2]	YES
40	-	RCC	Transition to RCOSC in case of critical faults on clock sources	Software procedure[2]	YES
41	CEM_13	SPURIOUS activation of boot/reset functionalities	Unexpected activation of JTAG or debug signals	CEM software procedure[3]	NO
			Unexpected activation of SSCM CS to DCF clients during runtime		
			Unexpected activation of STCU3 during runtime		
42	-	TCU	Test circuitry group spurious activation	FCCU fake fault injection[1]	NO
43	-	-	-	-	-
44	-	COMPENSATION CELLS	Pad compensation disabled	FCCU fake fault injection[1]	NO
45	-	ERRIN0	Error from bidirectional input error signal (External or internal failure to MCU)	Software procedure[2]	YES
46	-	CORE LOCK ALARM	Core lock/split change state alarm	FCCU fake fault injection[1]	YES
47	-	DMA LOCK ALARM	Dma lock/split change state alarm	FCCU fake fault injection[1]	YES
48	-	OTA ALARM	OTA-X1 swap error	Software procedure[2]	NO

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
49	-	SMPU	SMPU region violation	Software procedure[2]	YES
50	-	SMPU	SMPU monitors that no signal is altered by the SMPU logic	FCCU fake fault injection[1]	YES
51	-	NVMC1	EDC after ECC for code NVMC1	Software procedure[2]	YES
52	-	NVMC1	EDC after ECC for data NVMC1	Software procedure[2]	YES
53	-	NVMC1	Flash encoding errors	Software procedure[2]	YES
54	-	NVMC1	PFlashC address feedback error	Software procedure[2]	YES
55	-	NVMC2	EDC after ECC for code NVMC2	Software procedure[2]	YES
56	-	-	-	-	-
57	-	NVMC2	Flash encoding error	Software procedure[2]	YES
58	-	NVMC2	PFlashC address feedback error	Software procedure[2]	YES
59	-	NVMC1	Protocol error on the 2 ports of NVMC1	FCCU fake fault injection[1]	NO
60	-	NVMC2	Protocol error on the 2 ports of NVMC2	FCCU fake fault injection[1]	NO
61	-	SRAMC1	EDC after ECC PFlashC address feedback error	FCCU fake fault injection[1]	YES
62	-	SRAMC1	PRAMC memory feedback error	FCCU fake fault injection[1]	YES
63	-	SRAMC1	Address/Control EDC/Parity check PFlashC address feedback error	FCCU fake fault injection[1]	YES
64	-	SRAMC2	EDC after ECC PRAMC memory feedback error	FCCU fake fault injection[1]	YES
65	-	SRAMC2	PFlashC address feedback error	FCCU fake fault injection[1]	YES
66	-	SRAMC2	Address/Control EDC/Parity check FCCU alarm	FCCU fake fault injection[1]	YES
67	CEM_0	Core1 - AXIM	e2eECC data correctable error Core1 AXIM	CEM software procedure[3]	NO
		Core1 - AHBM	e2eECC data correctable error Core1 AHBM		
		Core2 - AXIM	e2eECC data correctable error Core2 AXIM		
		Core2 - AHBM	e2eECC data correctable error Core2 AHBM		
		HSM - AHB	e2eECC data correctable error HSM AHB		
		DMA1 - AHBMem	e2eECC data correctable error DMA1 AHB memory		
		DMA1 - AHBPer	e2eECCData correctable error DMA1 AHB peripheral		

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
67	CEM_0	DMA2 - AHBMem	e2eECCData correctable error DMA2 AHB memory	CEM software procedure[3]	NO
		DMA2 - AHBPer	e2eECC data correctable error DMA2 AHB peripheral		
68	CEM_1	Core1 - AXIM	e2eECC data uncorrectable error Core1 AXIM	CEM software procedure[3]	NO
		Core1 - AHBM	e2eECC data uncorrectable error Core1 AHBM		
		Core2 - AXIM	e2eECC data uncorrectable error Core2 AXIM		
		Core2 - AHBM	e2eECC data uncorrectable error Core2 AHBM		
		HSM - AHB	e2eECC data uncorrectable error HSM AHB		
		DMA1 - AHBMem	e2eECC data uncorrectable error DMA1 AHB memory		
		DMA1 - AHBPer	e2eECCData uncorrectable error DMA1 AHB peripheral		
		DMA2 - AHBMem	e2eECCData uncorrectable error DMA2 AHB memory		
		DMA2 - AHBPer	e2eECC data uncorrectable error DMA2 AHB peripheral		
69	CEM_2	Core1 - AXIM	e2eECC protocol error Core1 AXIM	CEM software procedure[3]	NO
		Core1 - AHBM	e2eECCProtocol error Core1 AHBM		
		Core2 - AXIM	e2eECC protocol error Core2 AXIM		
		Core2 - AHBM	e2eECC protocol error Core2 AHBM		
		HSM - AHB	e2eECC protocol error HSM AHB		
		DMA1 - AHBMem	e2eECC protocol error DMA1 AHB memory		
		DMA1 - AHBPer	e2eECC protocol error DMA1 AHB peripheral		
		DMA2 - AHBMem	e2eECC protocol error DMA2 AHB memory		
		DMA2 - AHBPer	e2eECC protocol error DMA2 AHB peripheral		
70	-	AXI watchdog	-	FCCU fake fault injection[1]	NO
71	CEM_3	Completer port to Cores AHBP	e2eECC data correctable error Cores AHBP	CEM software procedure[3]	NO
		Completer port to Cores AHB1	e2eECC data correctable error AHB1		
		Completer port to Cores AHB2	e2eECC data correctable error AHB2		
		Completer port to Cores APB1	e2eECC data correctable error APB1		
		Completer port to Cores APB2	e2eECC data correctable error APB2		
		Completer port to Cores HRTIM1	e2eECC data correctable error HRTIM1 AXI		

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
71	CEM_3	Completer port to Cores HRTIM2	e2eECC data correctable error HRTIM2 AXI	CEM software procedure(3)	NO
72	CEM_4	Completer port to Cores AHBP	e2eECC data uncorrectable error AHBP	CEM software procedure(3)	NO
		Completer port to Cores AHB1	e2eECC data uncorrectable error AHB1		
		Completer port to Cores AHB2	e2eECC data uncorrectable error AHB2		
		Completer port to Cores APB1	e2eECC data uncorrectable error APB1		
		Completer port to Cores APB2	e2eECC data uncorrectable error APB2		
		Completer port to Cores HRTIM1	e2eECC data uncorrectable error HRTIM1 AXI		
		Completer port to Cores HRTIM2	e2eECC data uncorrectable error HRTIM2 AXI		
73	CEM_5	Completer port to Cores AHBP	e2eECC protocol error AHBP	CEM software procedure(3)	NO
		Completer port to Cores AHB1	e2eECC protocol error AHB1		
		Completer port to Cores AHB2	e2eECC protocol error AHB2		
		Completer port to Cores APB1	e2eECC protocol error APB1		
		Completer port to Cores APB2	e2eECC protocol error APB2		
		Completer port to Cores HRTIM1	e2eECC protocol error HRTIM1 AXI		
		Completer port to Cores HRTIM2	e2eECC protocol error HRTIM2 AXI		
74	CEM_6	Bridge protection - AHB1		CEM software procedure(3)	NO
		Bridge protection - AHB2			
		Bridge protection - APB1			
		Bridge protection - APB2			
75	-	RCCU Core	RCCUS for Cores lockstep	FCCU fake fault injection(1)	YES
76	-	RCCU DMA	RCCUS for DMA lockstep	FCCU fake fault injection(1)	YES
77	CEM_7	RCCU vs. AHB1 subordinators	from RCCUSs for subordinators dataless duplication lockstep	CEM software procedure(3)	NO
		RCCU vs. AHB2 subordinators			
		RCCU vs. APB1 subordinators			
		RCCU vs. APB2 subordinators			

FCCU channel	CEM instance	Source	Failure description	Error injection mechanism	Error path verification
77	CEM_7	RCCU vs. AHBS subordinators	from RCCUSs for subordinators dataless duplication lockstep	CEM software procedure[3]	NO
		RCCU vs. HRTIM1 subordinators			
		RCCU vs. HRTIM2 subordinators			
		RCCU vs. NVMC1subordinators			
		RCCU vs. NVMC2 subordinators			
		RCCU vs. RAMC1 subordinators			
		RCCU vs. RAMC2 subordinators			
78	-	Core1	Core1 lockup error	FCCU fake fault injection[1]	NO
79	-	Core2	Core2 lockup error	FCCU fake fault injection[1]	NO
80	CEM_8	Core1 I-TCM	I-TCM Core1 address feedback err	CEM software procedure[3]	NO
		Core1 D0-TCM	D0-TCM Core1 address feedback err		
		Core1 D1-TCM	D1-TCM Core1 address feedback err		
		Core1 I-TCM	I-TCM Core1 EDC after ECC		
		Core1 D0-TCM	D0-TCM Core1 EDC after ECC		
		Core1 D1-TCM	D1-TCM Core1 EDC after ECC		
		Core2 I-TCM	I-TCM Core2 address feedback err		
		Core2 D0-TCM	D0-TCM Core2 address feedback err		
		Core2 D1-TCM	D1-TCM Core2 address feedback err		
		Core2 I-TCM	I-TCM Core2 EDC after ECC		
		Core2 D0-TCM	D0-TCM Core2 EDC after ECC		
		Core2 D1-TCM	D1-TCM Core2 EDC after ECC		
81	CEM_14	Upsizer error - Core1 AHB	Error response on the bus	CEM software procedure[3]	NO
		Upsizer error - Core2 AHB			
		Upsizer error - HSM			
		Upsizer error - DMA1 AHBP			
		Upsizer error - DMA1 AHBM			
		Upsizer error - DMA2 AHBP			
		Upsizer error - DMA2 AHBM			

1. *Faults injectable by using the FCCU fake fault interface.*
2. *Faults injectable by using a software procedure to stimulate the fault.*
3. *Faults injectable by using the CEM interface.*

Before the safety application starts, the user must configure a proper reaction for each FCCU failure input source. See the “FCCU registers reset values” paragraph in the SR5E1x reference manual for the device default configuration.

Possible fault reactions are:

- Internal reactions (the user can configure separately the internal reactions for each FCCU input):
  - No reset reaction (default)
  - IRQ (NMI or alarm)
  - Short functional reset
  - Long functional reset
- External reaction:
  - Error out (EOUT) signaling the status of the MCU

The FCCU controls the EOUT pins without any CORE intervention.

The correctness of FCCU behavior is checked by the FCCU output supervision unit (FOSU). This module monitors the integrity of the FCCU itself, by waiting for any reaction of the FCCU in a fixed time window after an error arrives. 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. The FOSU does not require any configuration done by the user. A functional reset has no impact on the FCCU.

3 FCCU fault injection, clearing and fake fault interface

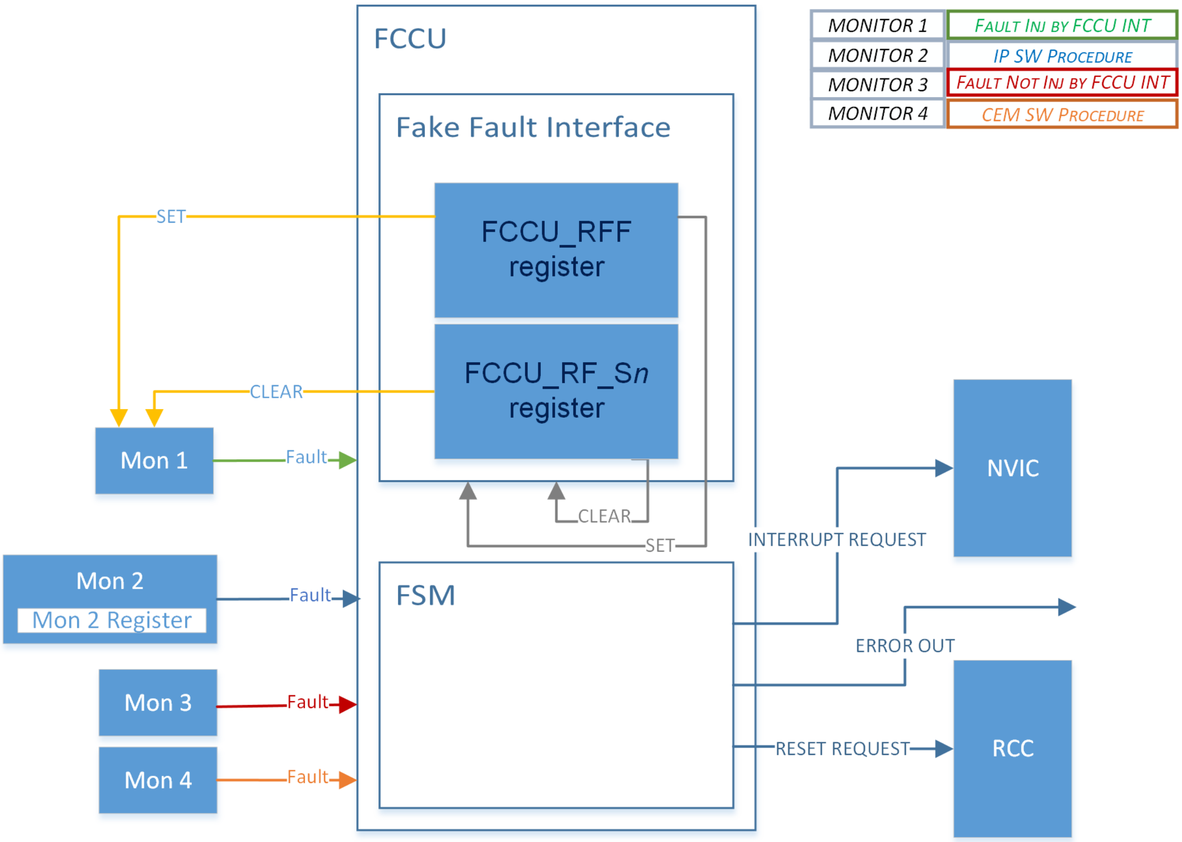
The application can use the fault injection mechanism to diagnose physical defects affecting the connections between the hardware monitors and the FCCU. The procedure to inject a fault depends on the specific monitor.

We can distinguish among four different sets of error path, see the table below:

Table 4. **FCCU error path and monitors**

Monitor type	Fault injection mechanism	Error path verification	Error path test interface
MON 1	Fault injectable by FCCU interface	YES	FCCU fake fault
MON 2	Fault injectable by software procedure	YES	Software procedure (for example, PLL0 loss of lock)
MON 3	Fault NOT injectable by FCCU interface	NO	FCCU fake fault
MON 4	Fault injectable by CEM software procedure	NO/YES (see the dedicate section)	CEM software procedure

Figure 2. **FCCU inner**



FCCU fake fault interface can inject faults to verify the entire error path and reaction. (Refer to the Table 3)

When the error path is not testable directly by FCCU interface (see Mon 4 – Mon 2) the error injection is still available by programming the interface of some monitors.

Referring to the Figure 2:

- To generate the FCCU fake fault event in the Mon 1, an optional signal is available (SET signal in the Figure 2, yellow arrow). The fake fault injection is executed by a write operation into the FCCU_RFF register, and the corresponding reaction is not maskable. When available, the fake fault injection method is suggested in the following sections.

Note: Some monitors miss the SET signal. In this case (SET signal in the Figure 2, grey 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 Mon 1, an optional signal is available (CLEAR signal in the Figure 2, yellow arrow). The de-assertion of the FCCU_RF_Sn status bit indicates that the software has properly cleared the fault.

Note: Some monitors miss the CLEAR signal. In this case (CLEAR signal in the Figure 2, grey arrow) the fault can be cleared by a write operation into a specific register of the monitor.

CEM interface provides an injection procedure in Mon 4 (see in the Figure 2), orange arrow.

Refers to the Section 3.1: CEM module.

Depending on the type of monitor, fault indication can either be a pulse (edge-triggered) or a constant value (level-triggered).

The fault management shall consider that the user can configure a fault input channel in the FCCU as:

- Hardware 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.
- Software 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 software recoverable. In such a way, the FCCU clears the respective status flag only after an explicit request from the software. In case of hardware recoverable, the status flag automatically clears, and the application may not react properly to the incoming fault.

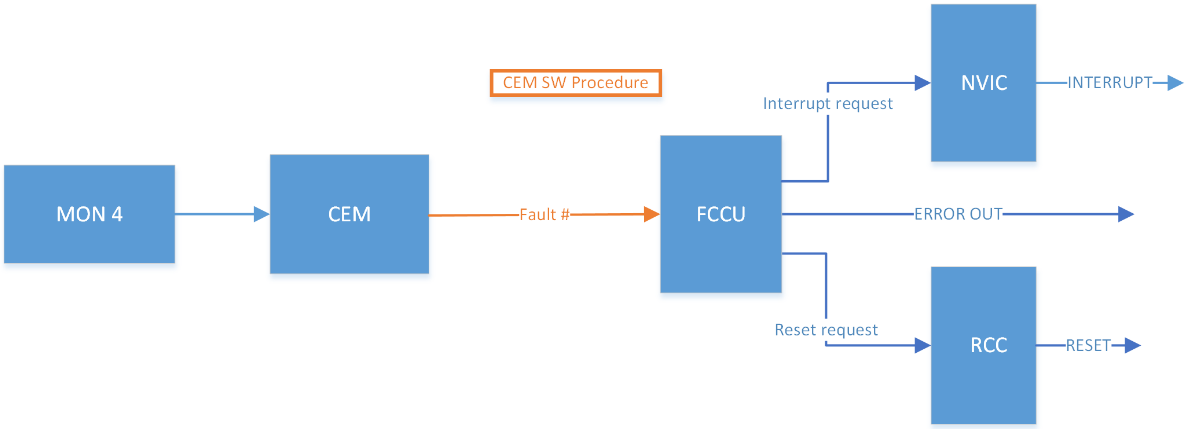
3.1 CEM module

The collective error manager (CEM) module contains registers dedicated to control and status reporting of errors from safety monitors to FCCU module. The error signals connected to a CEM module are OR-ed together to generate one FCCU trigger. The module contains internal registers (per error group) for controlling and capturing status of errors from safety monitors, as well as fake fault injection, and IPS programmable registers for accessing internal registers.

IPS programmable registers (32-bit) are used to control and capture error status of the CEM internal registers. Refer to device SR5E1 Reference Manual for further details on the CEM module.

Note: The SR5E1x contains 15 CEM instances (CEM0 to CEM14). Each instance has only one error group (Group0).

Figure 3. CEM error reaction path



The fake fault mechanism inside CEM module is available to verify the error reaction path by a software procedure: CEM channel must be enabled (default is already enabled), the fault must be injected through the CEM CMD register.

The user can inject this type of fault by:

1. Enabling fault (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x1, CMD[FAULT_OR_GRP_NUM] = Fault number or group number
2. Injecting a CEM fault((CMD[KEY] = 0xA5A5, CMD[CMD] = 0x3,CMD[FAULT_OR_GRP_NUM] = Fault number or group number)

The FCCU error reaction path is verified if the FCCU_RF_Sx[RFSyy] status bit is set after step (2)

The user can clear the fault by:

1. Clearing the injection mechanism at CEM level((CMD[KEY] = 0xA5A5, CMD[CMD] = 0x4, CMD[FAULT_OR_GRP_NUM] = Fault number or group number)
2. Clearing the relevant FCCU_RF_Sx[RFSyy] bit

4 Faults description

The following sections describe all the faults incoming to FCCU for SR5E1 device and how, if possible, to inject them for checking the integrity of the relevant reaction path.

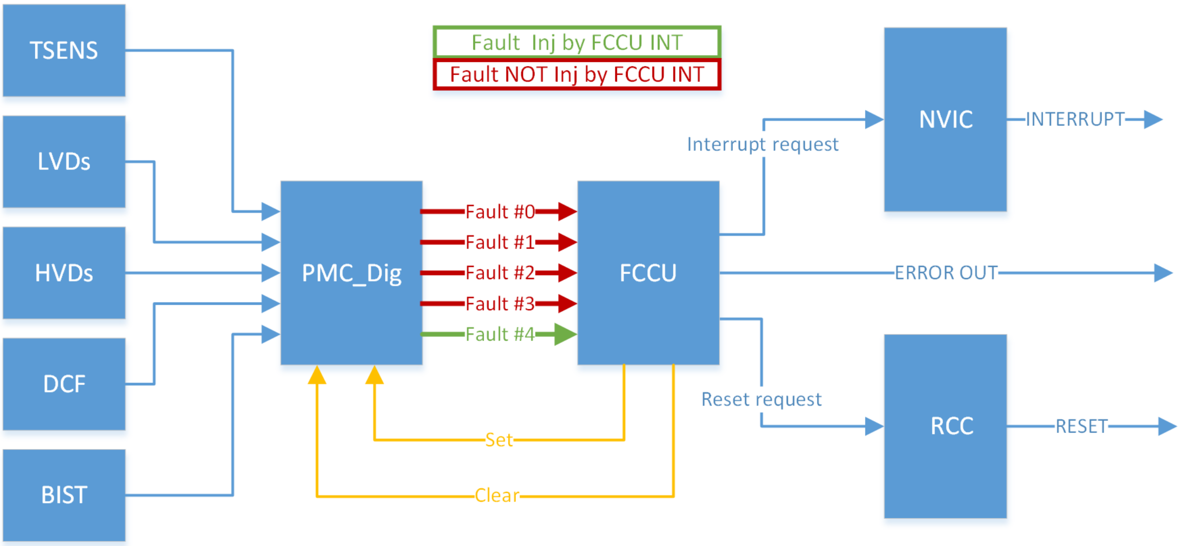
The following color convention is adopted in the following figures:

- The **GREEN** arrow marks the faults injectable inside the safety monitor (MON 1) by the FCCU fake fault interface.
- The **BLUE** arrow marks the faults injectable inside the safety monitor (MON 2) by a software procedure that stimulates the error path.
- The **RED** arrow marks the faults that cannot be directly injectable (MON3).
  There is no direct connection to stimulate the error path between safety monitor and FCCU. Only the fake fault, internally to FCCU module, can be injected (GREY arrows in the Figure 2).
- The **ORANGE** arrow marks the faults injectable inside the safety monitor (MON 4) by a software procedure using the CEM interface.

4.1 PMC_DIG faults

PMC_DIG is the source of five different FCCU input faults. Refer to device SR5E1 microcontroller reference manual for further details on PMC_DIG.

Figure 4. **PMC_DIG faults**



4.1.1 Temperature detector out of range (Fault #0)

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

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x00 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS0] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS0] bit be reset.

Note: For TS0, TS1, TS2 value refer to the datasheet.

4.1.2 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 (see SR5E1 reference manual for further details on LVDs) and their output signals are OR-ed before arriving at the FCCU failure input #1.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x01 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS1] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS1] bit be reset.

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

Each HVD detects a voltage that rises above the defined threshold and the PMC_DIG forwards this fault to the FCCU. The MCU embeds some HVDs (see SR5E1 reference manual for further details on HVDs) and their output signals are OR-ed before arriving at the FCCU failure input #2.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x02 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS2] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS2] bit be reset.

4.1.4 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 error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x03 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS3] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS3] bit be reset.

4.1.5 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 error path between safety monitor and FCCU is stimulated.

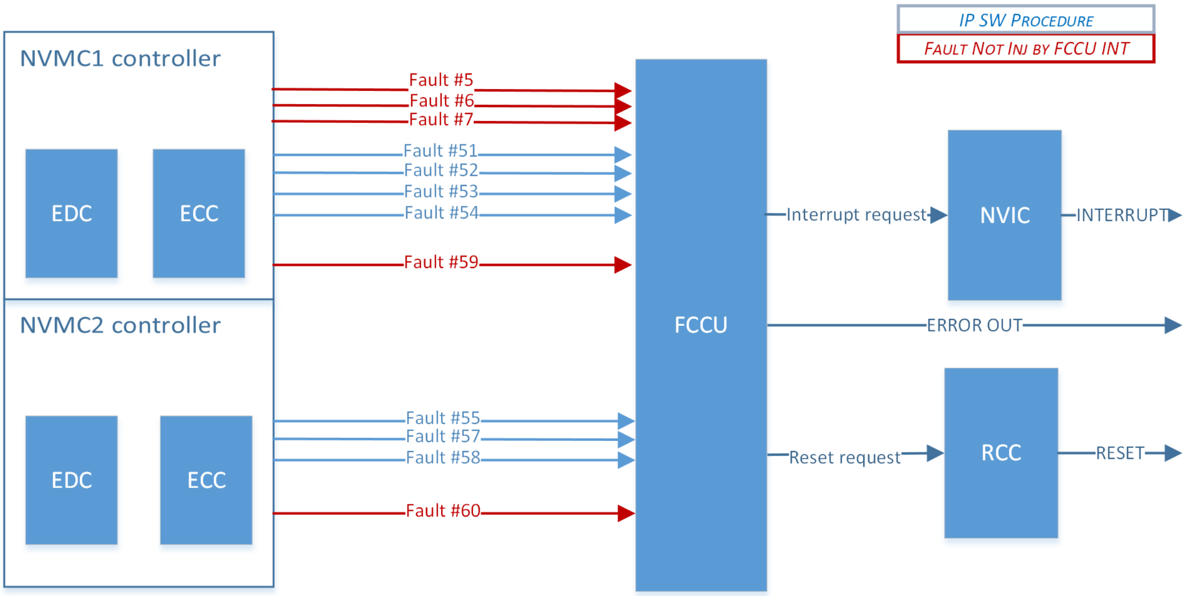
The user must set the PMC_DIG_BIST_CTRL[NCFEN] bit to enable a user BIST not critical fault indication to the FCCU and inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x04. The FCCU error reaction path is verified if both the FCCU_RF_S0[RFS4] and the PMC_DIG_BIST_CTRL[NCFST] status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

The PMC_DIG_BIST_CTRL [NCFST] status bit indicates a BIST fail on completion of BIST sequence. The fault clear mechanism requires that the status FCCU_RF_S0[RFS4] bit be reset.

4.2 FLASH/PFLASHC faults

Throughout the document, both flash and nonvolatile memory (NVM) are used indistinctly because SR5E1x uses flash memory as a type of NVM. Note that for flash memory controller PFLASHC and NVMC are used indistinctly in the document as well. The flash memory controllers (NVMPC1, NVMPC2) manage CPU AXI accesses to the flash memory. They implement the erase and program flash memory operations and the read and write protection mechanisms. Refer to device SR5E1x reference manual for further details on FLASH and PFLASHC.

Figure 5. **FLASH/PFLASHC faults**



4.2.1 Flash fatal error (Fault #5)

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 error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x05 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS5] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS5] bit be reset.

4.2.2 Flash reset error (Fault #6)

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 (RAM not visible to the user) during an internal self-check routine (always running).

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x06 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS6] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS6] bit be reset.

4.2.3 Flash read reference error (Fault #7)

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

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x07 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS7] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS7] bit be reset.

4.2.4 NVMC1 EDC after ECC for code FLASH (Fault #51)

The EDC after ECC logic inside the NVMC1 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for the code flash array and the NVMC1 forwards this fault to the FCCU.

The user can inject this fault by:

1. Enabling the error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMCEDC] = 0x1, of NVM1 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTFRC_NVMCEDC] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS19] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTFRC_NVMCEDC] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS19] bit

4.2.5 NVMC1 EDC after ECC for data FLASH (Fault #52)

The EDC after ECC logic inside the NVMC1 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for the data flash and the NVMC1 forwards this fault to the FCCU.

The user can inject this fault by:

1. Enabling the error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMDEDC] = 0x1, of NVM1 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTENA_NVMDEDC] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS20] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTENA_NVMDEDC] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS20] bit

4.2.6 NVMC1 FLASH memory access fault (Fault #53)

The NVMC1 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 error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMENCE] = 0x1, of NVM1 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTENA_NVMENCE] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS21] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTENA_NVMENCE] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS21] bit

4.2.7 NVMC1 address feedback error (Fault #54)

The NVMC1 flash controller detects a transaction monitor mismatch when compared with the flash safety feedback outputs and it forwards this fault to FCCU.

The user can inject this fault by:

1. Enabling the error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMPCENC] = 0x1, of NVM1 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTENA_NVMPCENC] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS22] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTENA_NVMPCENC] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS22] bit

4.2.8 NVMC2 EDC after ECC for code FLASH (Fault #55)

The EDC after ECC logic inside the NVMC2 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for the code flash array and the NVMC2 forwards this fault to the FCCU.

The user can inject this fault by:

1. Enabling the error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMCEDC] = 0x1, of NVM2 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTFRC_NVMCEDC] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS23] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTFRC_NVMCEDC] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS23] bit

4.2.9 NVMC2 FLASH memory access fault (Fault #57)

The NVMC2 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 error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMENCE] = 0x1, of NVM2 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTENA_NVMENCE] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS24] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTENA_NVMENCE] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS24] bit

4.2.10 NVMC2 address feedback error (Fault #58)

The NVMC2 flash controller detects a transaction monitor mismatch when compared with the flash safety feedback outputs and it forwards this fault to FCCU.

The user can inject this fault by:

1. Enabling the error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMPCENC] = 0x1, of NVM2 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTENA_NVMPCENC] = 0x1, of NVM2 module)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS25] status bit is set after step (2).

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTENA_NVMPCENC] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS25] bit

4.2.11 e2eECC NVMC1 protocol error (Fault #59)

The ECC logic detects protocol bits error and forwards this fault to FCCU.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x3B in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S1[RFS27] is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS27] bit be reset.

4.2.12 e2eECC NVMC2 protocol error (Fault #60)

The ECC logic detects protocol bits error and forwards this fault to FCCU.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

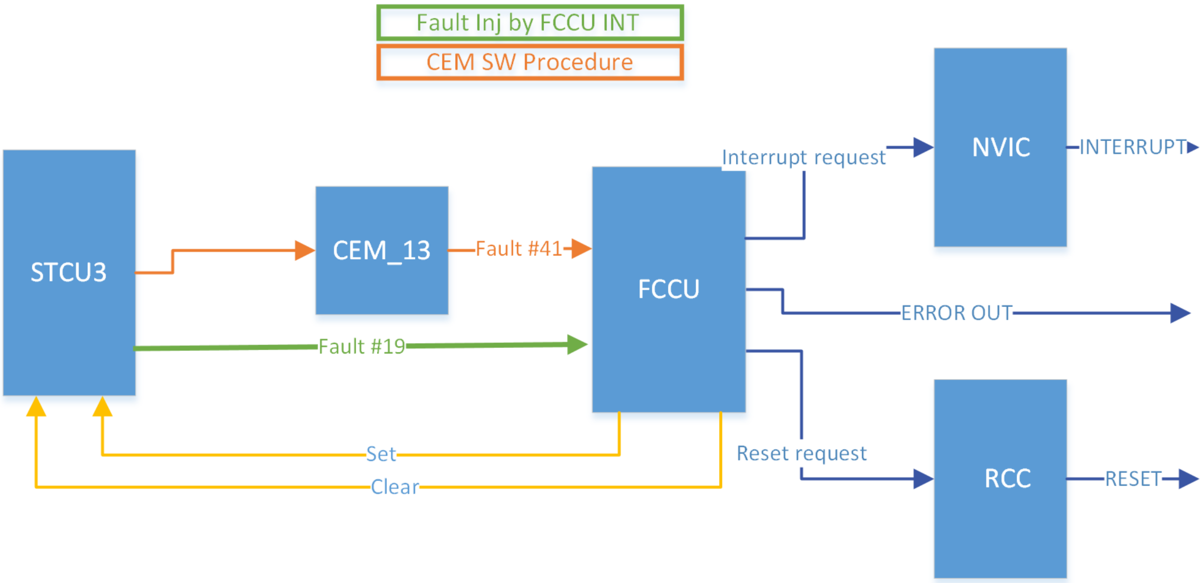
The user injects the fake fault by setting the error code 0x3C in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S1[RFS28] is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS28] bit be reset.

4.3 STCU3 faults

The STCU3 is a comprehensive programmable hardware module that controls the execution of the self-test, that runs a specific logic built in self-test (CBIST) and a memory built-in self-test (MBIST). The STCU3 is the source of one FCCU input faults. Refer to device SR5E1x microcontroller reference manual for further details on STCU3.

Figure 6. STCU3 faults



4.3.1 BIST result - wrong signature (STCU3 recoverable fault), (Fault #19)

If the BIST detects a fault that is configured as recoverable fault, the STCU3 forwards this fault to the FCCU. Although BIST can detect permanent faults, this fault can also be triggered in case of transient faults. The STCU3 can trigger this fault only during BIST execution (SR5E1x device only supports offline BIST. For more details, refer to device SR5E1 reference manual).

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user injects the fake fault by setting the error code 0x13 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S0[RFS19] is set.

The fault clear mechanism requires that the status FCCU_RF_S0[RFS19] bit be reset.

4.3.2 SPURIOUS STCU3 activation (Fault #41, CEM_13, bit#2)

Unexpected activation of STCU3 during the application execution can interfere with the application. If this event occurs, a dedicated glue logic forwards this fault to the FCCU.

This error signal is collected by CEM_13, before arriving to the FCCU failure input #41.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details see the Section 3.1: CEM module.

Table 5. CEM reg bit # for spurious STCU3 activation

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
2	SPURIOUS STCU3 ACTIVATION	0x2

4.4 DMA faults

Direct memory access (DMA) provides high-speed data transfer between peripherals and memory and between memory and memory. Data can be quickly moved by DMA without any CPU action. Refer to device SR5E1x reference manual for further details on the DMA.

Figure 7. DMA faults

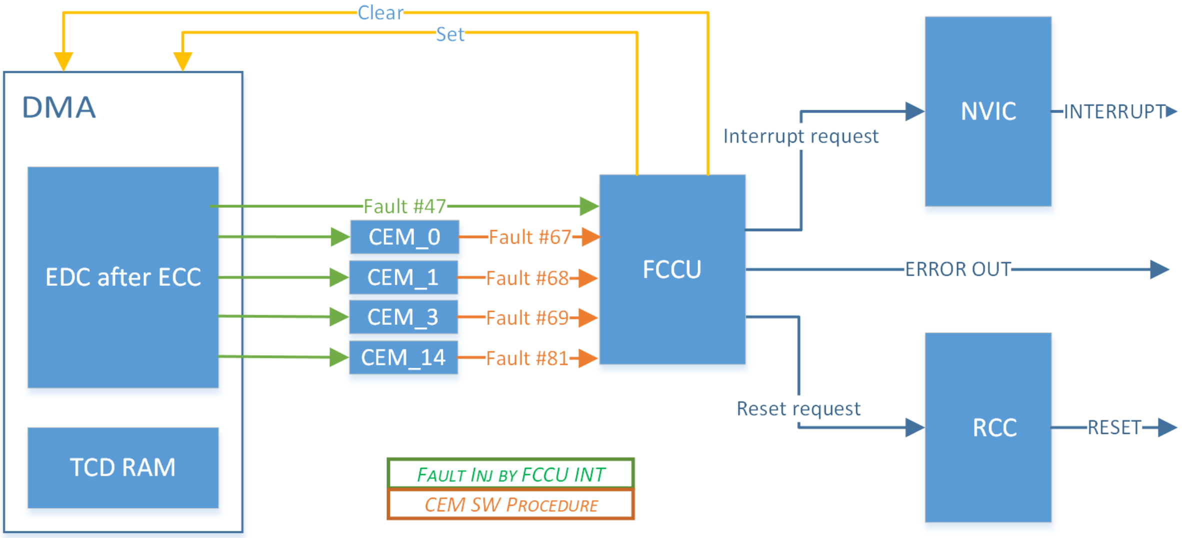
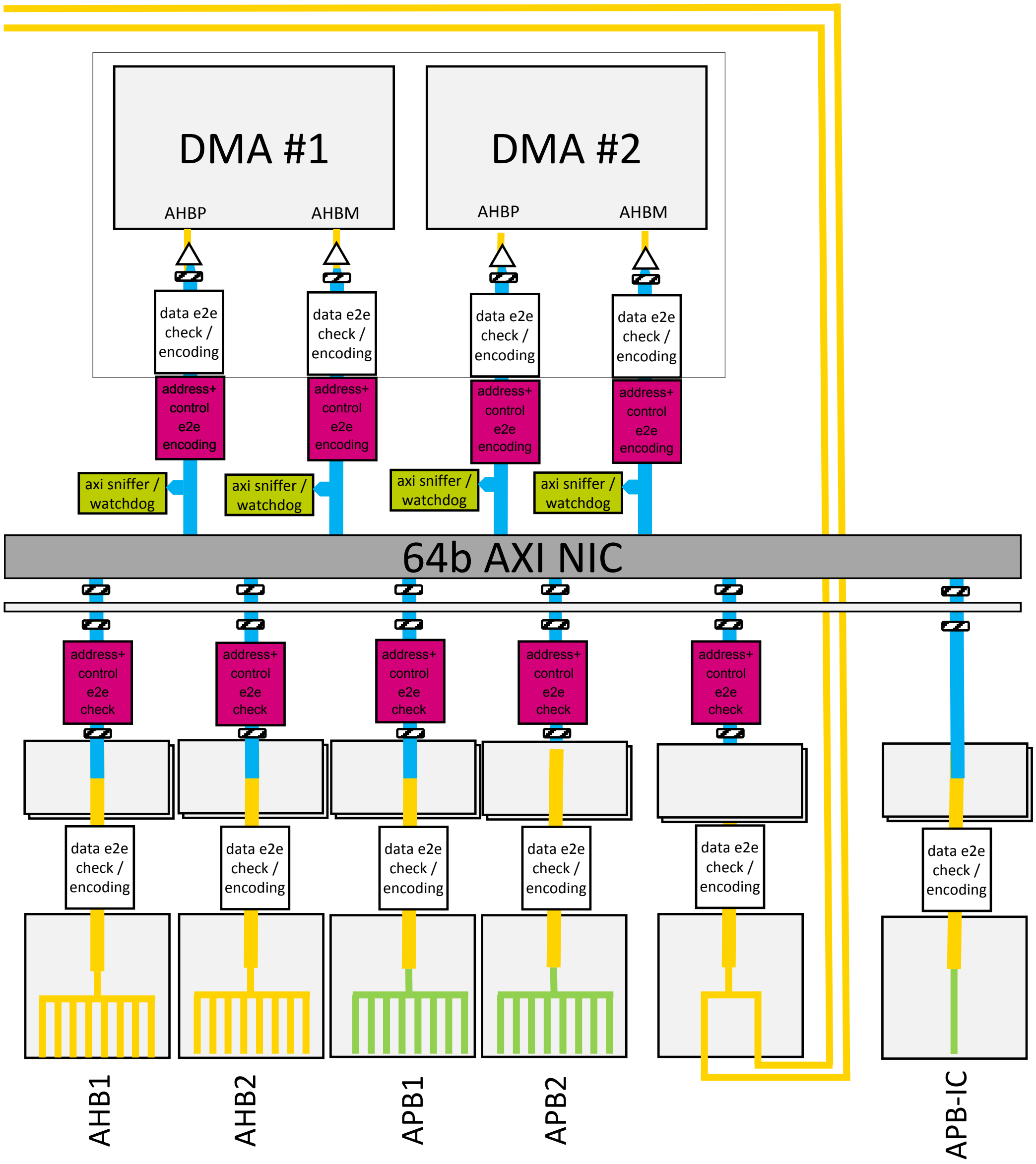


Figure 8. **DMA e2eECC schematic**



4.4.1 DMA lock/split change state alarm (Fault #47)

Each DMA instance has a replica that can be enabled in lockstep to reach a high level of safety.

The DMA controllers can work in lock or split mode, an involuntary change in mode triggers to FCCU an error.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user must set the FCCU_RFF[FRFC] field to the value 0x2F. The FCCU error reaction is verified if the FCCU_RF_S1[RFS15] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS15] bit be reset.

4.4.2 e2eECC data correctable error DMA1/2 AHB Memory/Peripheral (Fault #67, CEM _0)

The ECC logic detects data correctable error bits. This error signal is collected by CEM_0, before arriving to the FCCU failure input #67.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details see the Section 3.1: CEM module.

Table 6. CEM reg bit # for DMA correctable error

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
5	e2eECC data correctable error DMA1 AHB memory	0x5
6	e2eECC data correctable error DMA1 AHB peripheral	0x6
7	e2eECC data correctable error DMA2 AHB memory	0x7
8	e2eECC data correctable error DMA2 AHB peripheral	0x8

4.4.3 e2eECC data uncorrectable error DMA1/2 AHB memory/peripheral (Fault #68, CEM_1)

The ECC logic detects data uncorrectable error bits. This error signal is collected by CEM_1, before arriving to the FCCU failure input #68.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details see the Section 3.1: CEM module.

Table 7. CEM reg bit # for DMA uncorrectable error

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
5	e2eECC data uncorrectable error DMA1 AHB memory	0x5
6	e2eECC data uncorrectable error DMA1 AHB peripheral	0x6
7	e2eECC data uncorrectable error DMA2 AHB memory	0x7
8	e2eECC data uncorrectable error DMA2 AHB peripheral	0x8

4.4.4 e2eECC protocol error DMA1/2 AHB memory/peripheral (Fault #69, CEM_3)

The ECC logic detects protocol error bits. This error signal is collected by CEM_3, before arriving to the FCCU failure input #69.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details see the Section 3.1: CEM module.

Table 8. CEM reg bit # for DMA protocol error

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
5	e2eECC protocol error DMA1 AHB memory	0x5
6	e2eECC protocol error DMA1 AHB peripheral	0x6
7	e2eECC protocol error DMA2 AHB memory	0x7
8	e2eECC protocol error DMA2 AHB peripheral	0x8

4.4.5 e2eECC upsizer error DMA1/2 AHB memory/peripheral (Fault #81, CEM_14)

When a decode error response is seen on the bus DMA1 AHB peripheral, DMA1 AHB memory, DMA2 AHB peripheral, DMA2 AHB memory a fault is detected. This error signals is collected by CEM_14, before arriving to the FCCU failure input #81.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details see the section Section 3.1: CEM module.

Table 9. CEM reg bit # for DMA upsizer

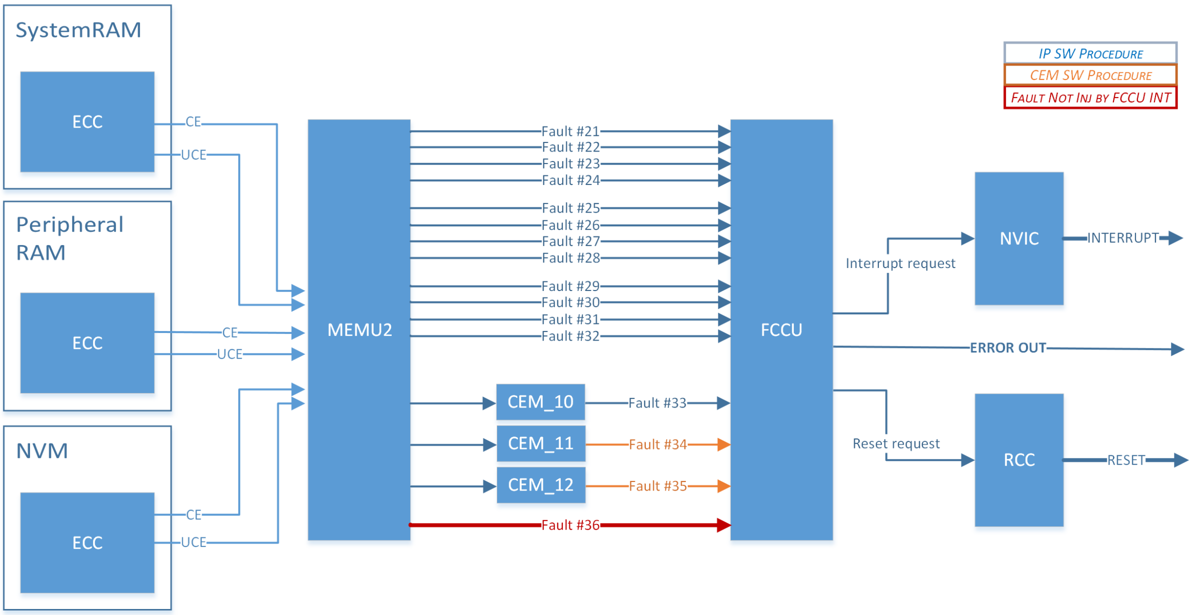
CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
3	Upsizer error - DMA1 AHBP	0x3
4	Upsizer error - DMA1 AHBM	0x4
5	Upsizer error - DMA1 AHBP	0x5
6	Upsizer error - DMA1 AHBM	0x6

4.5 MEMU2 faults

The MEMU2 is responsible for collecting and reporting error events to the fault collection and control unit (FCCU) associated with faults detected by memory BISTs as well as ECC (error correction code) logic, used on system-accessible RAM, peripheral local RAM, non-volatile memory (NVM).

When any of the following events occurs, the MEMU2 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 MEMU2 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. Refer to device SR5E1 reference manual for further details on the MEMU2.

Figure 9. MEMU2 faults



4.5.1 MEMU2 SYS Trigger fault

The user must address the correctable and uncorrectable ECC error trigger.

The user can choose the SYS_RAM_TRIG_0, SYS_RAM_TRIG_1, SYS_RAM_TRIG_2, SYS_RAM_TRIG_3.

The system RAM output trigger control register is used to check the connections between MEMU2 and FCCU via fake fault injection procedure, but the system RAM memory - MEMU2 - FCCU path is fully tested.

4.5.1.1 *SYS_RAM_TRIG_0 (Fault #21)*

The user can inject this fault by:

1. Forcing trigger SYS_RAM_TRIG_0 (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG0] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS21] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG0] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS21] bit

4.5.1.2 *SYS_RAM_TRIG_1 (Fault #22)*

The user can inject this fault by:

1. Forcing trigger SYS_RAM_TRIG_1 (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG1] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS22] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG1] = 0)

2. Clearing the relevant FCCU_RF_S0[RFS22] bit

4.5.1.3 *SYS_RAM_TRIG_2 (Fault #23)*

The user can inject this fault by:

1. Forcing trigger SYS_RAM_TRIG_2 (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG2] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS23] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG2] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS23] bit

4.5.1.4 *SYS_RAM_TRIG_3 (Fault #24)*

The user can inject this fault by:

1. Forcing trigger SYS_RAM_TRIG_3 (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG3] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS24] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (SYS_RAM_OUT_TRIG_CTRL[FR_SR_FCCU_TRIG3] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS24] bit

4.5.2 MEMU2 PERIPH Trigger fault

The user must address the correctable and uncorrectable ECC error trigger.

The user can choose the PERIPH _RAM_TRIG_0, PERIPH _RAM_TRIG_1, PERIPH _RAM_TRIG_2, PERIPH _RAM_TRIG_3.

This peripheral RAM output trigger control register is used to check the connections between MEMU2 and FCCU via fake fault injection procedure, but the peripheral RAM - MEMU2 - FCCU path is fully tested.

4.5.2.1 *PERIPH_RAM_TRIG_0 (Fault #25)*

The user can inject this fault by:

1. Forcing trigger PERIPH_RAM_TRIG_0 (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG0] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS25] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG3] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS25] bit

4.5.2.2 *PERIPH_RAM_TRIG_1 (Fault #26)*

The user can inject this fault by:

1. Forcing trigger PERIPH_RAM_TRIG_1 (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG1] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS26] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG1] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS26] bit

4.5.2.3 *PERIPH_RAM_TRIG_2 (Fault #27)*

The user can inject this fault by:

1. Forcing trigger PERIPH_RAM_TRIG_2 (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG2] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS27] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG2] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS27] bit

4.5.2.4 *PERIPH_RAM_TRIG_3 (Fault #28)*

The user can inject this fault by:

1. Forcing trigger PERIPH_RAM_TRIG_3 (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG3] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS28] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (PERIPH_RAM_OUT_TRIG_CTRL [FR_PR_FCCU_TRIG3] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS28] bit

4.5.3 MEMU2 NVM Trigger fault

The user must address the correctable and uncorrectable ECC error trigger.

The user can choose the NVM _TRIG_0, NVM _TRIG_1, NVM _TRIG_2 NVM _TRIG_3.

This NVM output trigger control register is used to check the connections between MEMU2 and FCCU via fake fault injection procedure, but the non-volatile memory (NVM) - MEMU2 - FCCU path is fully tested.

4.5.3.1 *NVM_TRIG_0 (Fault #29)*

The user can inject this fault by:

1. Forcing trigger NVM_TRIG_0 (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG0] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS29] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG0] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS29] bit

4.5.3.2 *NVM_TRIG_1 (Fault #30)*

The user can inject this fault by:

1. Forcing trigger NVM_TRIG_1 (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG1] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS30] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG1] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS30] bit

4.5.3.3 *NVM_TRIG_2 (Fault #31)*

The user can inject this fault by:

1. Forcing trigger NVM_TRIG_2 (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG2] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS31] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG2] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS31] bit

4.5.3.4 *NVM_TRIG_3 (Fault #32)*

The user can inject this fault by:

1. Forcing trigger NVM_TRIG_3 (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG3] = 1)

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS32] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_F_FCCU_TRIG3] = 0)
2. Clearing the relevant FCCU_RF_S0[RFS32] bit

4.5.3.5 *Sys/periph/NVM RAM uncorrectable/correctable error table overflow (Fault #33, CEM_10)*

Find below a resume table:

Table 10. **CEM reg bit # for memories error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	Sys RAM single bit error table overflow	0x0

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
1	Sys RAM uncorrectable error table overflow	0x1
2	Periph RAM single bit error table overflow	0x2
3	Periph RAM uncorrectable error table overflow	0x3
4	NVM single bit error table overflow	0x4
5	NVM uncorrectable error table overflow	0x5
6	NVM double bit error table overflow	0x6

4.5.3.5.1 CEM_REG_BIT#0 (Sys RAM single bit error table overflow)

The user can inject this fault by:

1. Forcing system RAM correctable error overflow flag (SYS_RAM_OUT_TRIG_CTRL[FR_SR_CEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (SYS_RAM_OUT_TRIG_CTRL[FR_SR_CEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 0)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.3.5.2 CEM_REG_BIT#1 (Sys RAM uncorrectable error table overflow)

The user can inject this fault by:

1. Forcing system RAM uncorrectable error overflow flag (SYS_RAM_OUT_TRIG_CTRL[FR_SR_UCEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (SYS_RAM_OUT_TRIG_CTRL[FR_SR_UCEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 1)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.3.5.3 CEM_REG_BIT#2 (Periph RAM single bit error table overflow)

The user can inject this fault by:

1. Forcing peripheral RAM correctable error overflow flag. (PERIPH_RAM_OUT_TRIG_CTRL[FR_PR_CEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (PERIPH_RAM_OUT_TRIG_CTRL[FR_PR_CEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 2)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.3.5.4 CEM_REG_BIT#3 (Periph uncorrectable error table overflow)

The user can inject this fault by:

1. Forcing peripheral RAM uncorrectable error overflow flag. PERIPH_RAM_OUT_TRIG_CTRL[FR_PR_UCEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (PERIPH_RAM_OUT_TRIG_CTRL[FR_PR_UCEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 3)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.3.5.5 CEM_REG_BIT#4 (NVM single bit error table overflow)

The user can inject this fault by:

1. Forcing NVM single correctable error overflow flag (NVM_OUT_TRIG_CTRL [FR_F_UCEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_F_UCEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 4)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.3.5.6 CEM_REG_BIT#5 (NVM uncorrectable error table overflow)

The user can inject this fault by:

1. Forcing NVM single correctable error overflow flag (NVM_OUT_TRIG_CTRL [FR_F_DCEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_F_DCEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 5)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.3.5.7 CEM_REG_BIT#6 (NVM double correctable table overflow)

The user can inject this fault by:

1. Forcing NVM double correctable error overflow flag. (NVM_OUT_TRIG_CTRL [FR_PR_SCEO] = 1 of module MEMU2)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS1] status bit is set.

The user can clear the fault by:

1. Clearing the fault source error (NVM_OUT_TRIG_CTRL [FR_PR_SCEO] = 0)
2. Clearing the status of CEM (CMD[KEY] = 0xA5A5, CMD[CMD] = 0x6, CMD[FAULT_OR_GRP_NUM] = 6)
3. Clearing the relevant FCCU_RF_S1[RFS1] bit

4.5.4 System RAM FIF0 overflow (Fault #34, CEM_11)

The system RAM FIFOs to MEMU2 overflow occurs and forwards this fault to the CEM_11 and FCCU. The FIF0 is n-deep and is shared among m-safety monitors. The maximum number of input sources that can be handled by a single FIFO is four. Whenever there is an overflow in the synchronous FIFO, it is denoted by sending a FIFO_overflow flag to FCCU via CEM.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module. Each CEM reg bit is associated with the FIFO overflow (from 0 to 8 and from 32 to 44).

4.5.5 Peripheral RAM FIF0 overflow (Fault #35, CEM_12)

The peripheral RAM FIFOs to MEMU2 overflow occurs and forwards this fault to the CEM_12 and FCCU. The FIF0 is n-deep and is shared among m-safety monitors. The maximum number of input sources that can be handled by a single FIFO is 4. Whenever there is an overflow in the synchronous FIFO, it is denoted by sending a FIFO_overflow flag to FCCU via CEM.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module. Each CEM reg bit is associated with the FIFO overflow (from 0 to 2, from 32 to 37).

4.5.6 Flash FIF0 overflow (Fault #36)

MEMU2 FLASH FIFO overflow occurs and forwards this fault to FCCU. Whenever there is an overflow in the synchronous FIFO, it is denoted by sending a FIFO_overflow flag to FCCU.

The FIFOs allocated for NVM safety mechanism also sends a backpressure (FIFO_FULL) signal to the NVM safety mechanism module.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

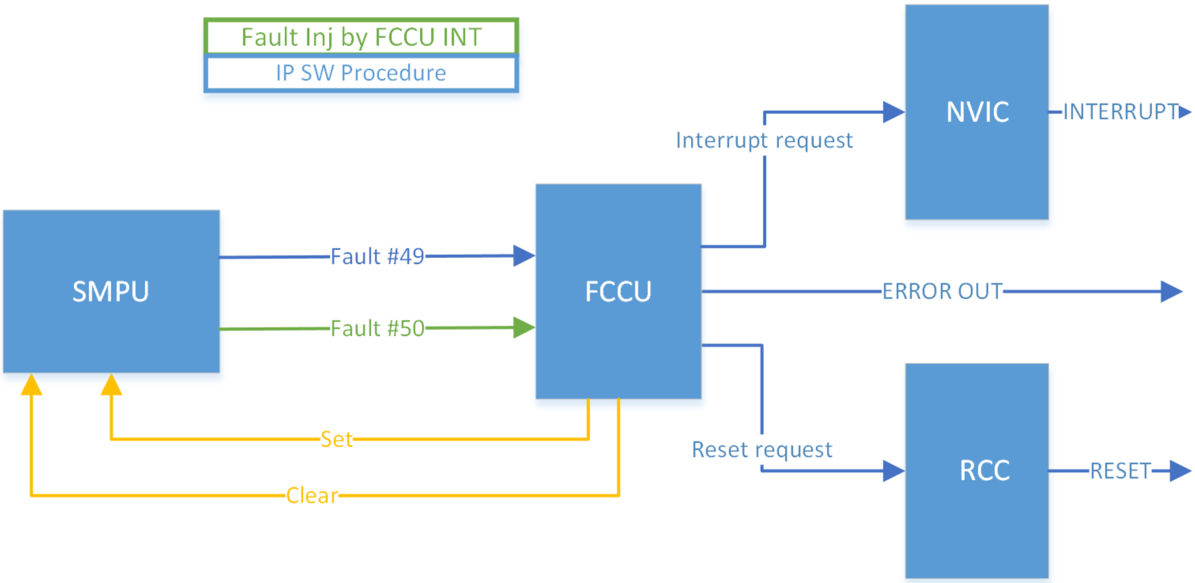
The user injects the fake fault by setting the error code 0x24 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S1[RFS4] is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS4] bit be reset.

4.6 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 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. Refer to SR5E1x reference manual for further details on the SMPU.

Figure 10. SMPU faults



4.6.1 SMPU region violation (Fault #49)

In case the SMPU denies access to a mapped memory location with insufficient rights, the hardware monitors inside the SMPU detect this event and forward this fault to the FCCU.

The user can inject this fault by a software procedure that accesses and writes a read-only memory address.

A hard fault interrupt must be handled. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS17] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS17] bit be reset.

4.6.2 SMPU monitors that no signal is altered by the SMPU logic (Fault #50)

When no access violation is detected, the SMPU shall act transparently with respect to control signals to and from the targeted target port.

The SMPU monitors that the SMPU logic does not alter any signal by comparing in vs out signals.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x32. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS18] is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS18] bit be reset.

4.7 Cores (Core1/2, HSM) faults

The SR5E1Ex has three cores in two distinct modules Cortex® M0+ (the hardware security module (HSM)), two Cortex® M7 in the main platform.

The two Cortex® M7, Core1 and Core2, can be configured:

- In decoupled mode, offering two processing units.
- In lock-step mode, offering one processing unit.

Refer to the SR5E1x reference manual for further details on the three cores.

Figure 11. **Cores faults**

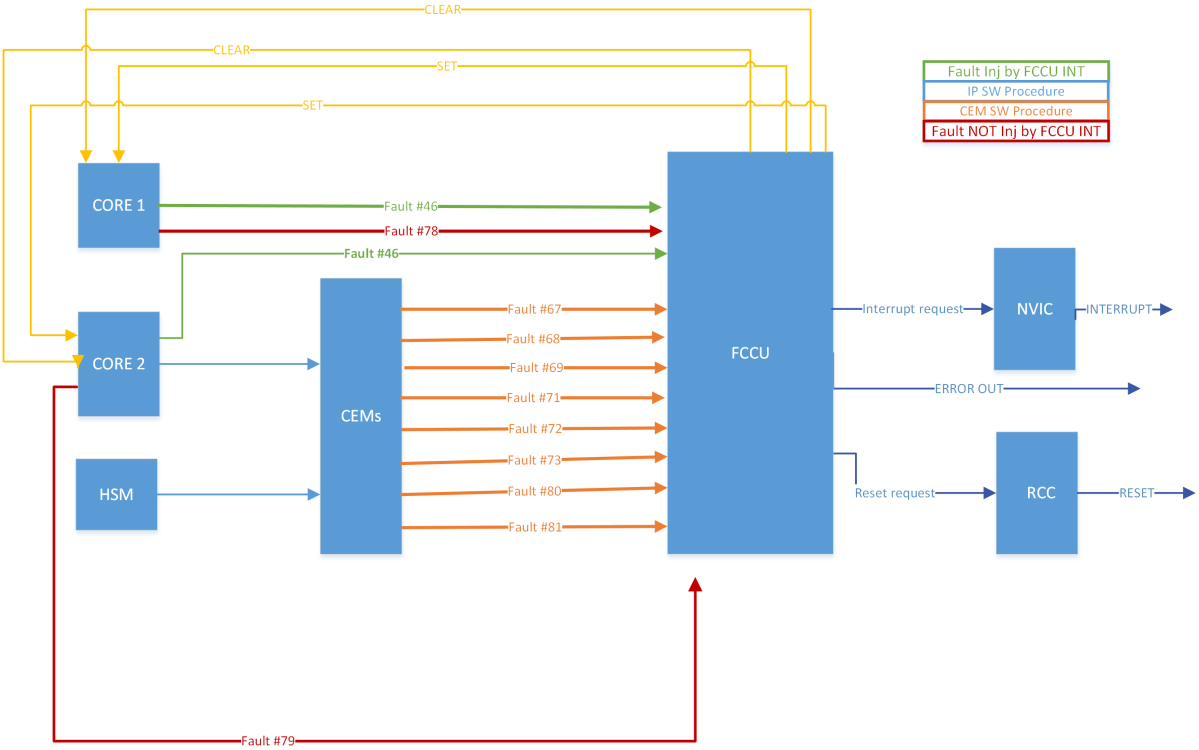


Figure 12. **Cores e2eECC schematic**

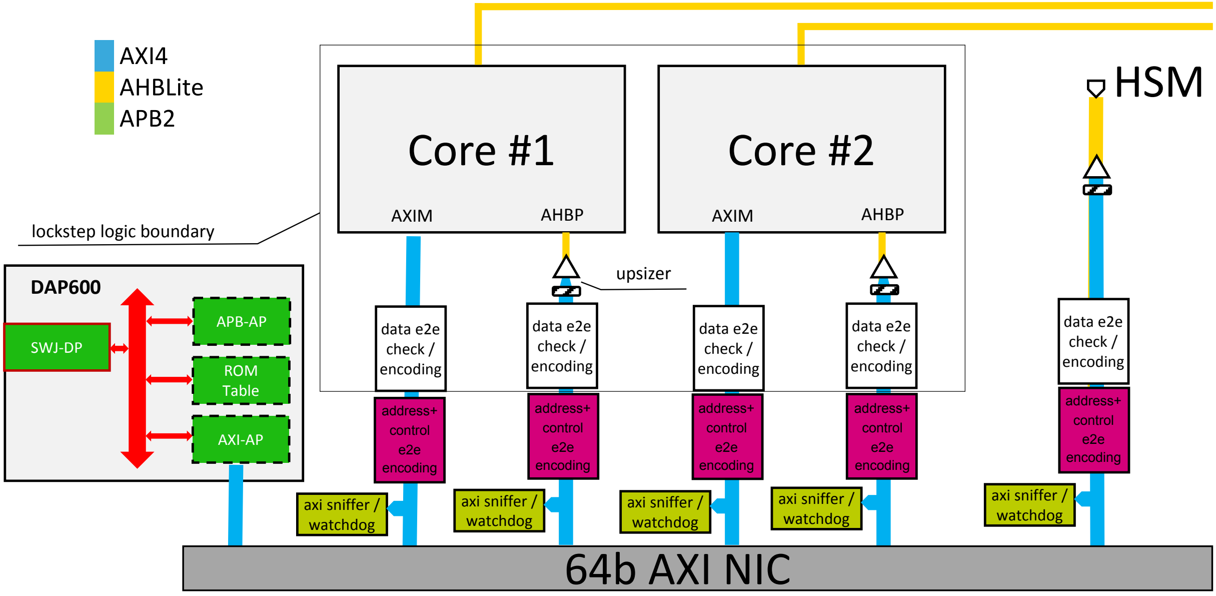
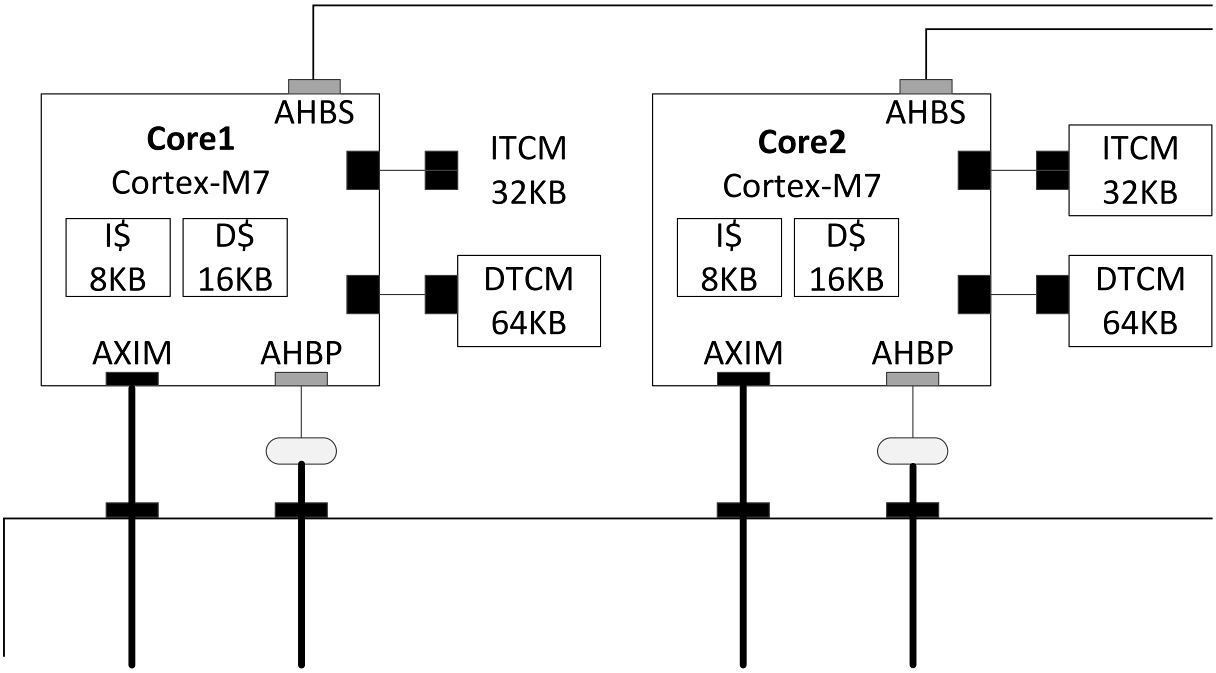


Figure 13. Cores of system architecture



4.7.1 Core lock/split change state alarm (Fault #46)

The two Cortex® M7, Core1 and Core2, can be configured: in decoupled mode, offering two processing units or in lock-step mode, offering one processing unit.

The Cores can configure in lock or split mode, an involuntary change in mode triggers to FCCU an error.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault by setting the FCCU_RFF[FRFC] field to the value 0x2E.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS14] bit be reset.

4.7.2 e2eECC data correctable error Core1/2 AXIM/AHBM (Fault #67, CEM_0)

The ECC logic, through the AXIM/AHBM/AHB Cores bus (see the Figure 12), detects data correctable error bits and forwards this fault to FCCU by CEM_0.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 11. CEM reg bit # for Cores correctable error

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	e2eECC data correctable error Core1 AXIM	0x0
1	e2eECC data correctable error Core1 AHBM	0x1
2	e2eECC data correctable error Core2 AXIM	0x2
3	e2eECC data correctable error Core2 AHBM	0x3
4	e2eECC data correctable error HSM AHBM	0x4

4.7.3 e2eECC data uncorrectable error Core1/2 - AXIM/AHBM and HSM - AHB (Fault #68, CEM_1)

The ECC logic, through AXIM/AHBM/AHB Cores bus (see the Figure 12), detects data uncorrectable error bits and forwards this fault to FCCU by CEM_1.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 12. **CEM reg bit # for Cores uncorrectable error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	e2eECC data uncorrectable error data correctable error Core1 AXIM	0x0
1	e2eECC uncorrectable error Core1 AHBM	0x1
2	e2eECC uncorrectable error Core2 AXIM	0x2
3	e2eECC data uncorrectable error Core2 AHBM	0x3
4	e2eECC data uncorrectable error HSM AHB	0x4

4.7.4 e2eECC protocol error Core1/2, HSM (Fault #69, CEM_2)

The ECC logic, through AXIM/AHBM/AHB Cores bus (see the Figure 12) detects protocol bits error and forwards this fault to FCCU by CEM_2.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 13. **CEM reg bit # for Cores protocol error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	e2eECC protocol error Core1 AXIM	0x0
1	e2eECC protocol error Core1 AHBM	0x1
2	e2eECC protocol error Core2 AXIM	0x2
3	e2eECC protocol error Core2 AHBM	0x3
4	e2eECC protocol error HSM AHB	0x4

4.7.5 Core1 lockup error (Fault #78)

Lockup is broadly defined as the symptom of a function or task using Core1 and not releasing it for a period. The lockup behavior is more often caused by an application use case and occurs during firmware code development.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x4E in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S2[RFS14] is set.

4.7.6 Core2 lockup error (Fault #79)

Lockup is broadly defined as the symptom of a function or task using Core2 and not releasing it for a period. The lockup behavior is more often caused by an application use case and occurs during firmware code development.

The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x4F in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S2[RFS15] is set.

4.7.7 Core1/2 address feedback err and EDC after ECC (Fault #80, CEM_8)

Malfunction of ECC logic may result in corruption of error event reporting. Thus, EDC after ECC check is performed on all read-modify-write transactions. If a mismatch is detected, indicating a failure in the ECC logic, the event is reported to FCCU.

An address feedback error reports a mismatch in the transmission path between the cores and the DTCM /ITCM ram array. This event is also reported to FCCU. For more schematic details, see the Figure 13.

Refer to SR5E1 reference manual for further details about the DTCM and ITCM.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 14. CEM reg bit # for Cores address feedback error

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	I-TCM Core1 address feedback err	0x0
1	D0-TCM Core1 address feedback err	0x1
2	D1-TCM Core1 address feedback err	0x2
3	I-TCM Core1 EDC after ECC	0x3
4	D0-TCM Core1 EDC after ECC	0x4
5	D1-TCM Core1 EDC after ECC	0x5
6	I-TCM Core2 address feedback err	0x6
7	D0-TCM Core2 address feedback err	0x7
8	D1-TCM Core2 address feedback err	0x8
9	I-TCM Core2 EDC after ECC	0x9
10	D0-TCM Core2 EDC after ECC	0xA
11	D1-TCM Core2 EDC after ECC	0xB

4.7.8 Upsizer error Core1/2, HSM (Fault #81, CEM_14)

When a decode error response is seen on the bus Core1/2 - AHB, HSM a fault is detected and forwards this fault to FCCU by CEM_14 (see the Figure 12. Cores e2eECC schematic).

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

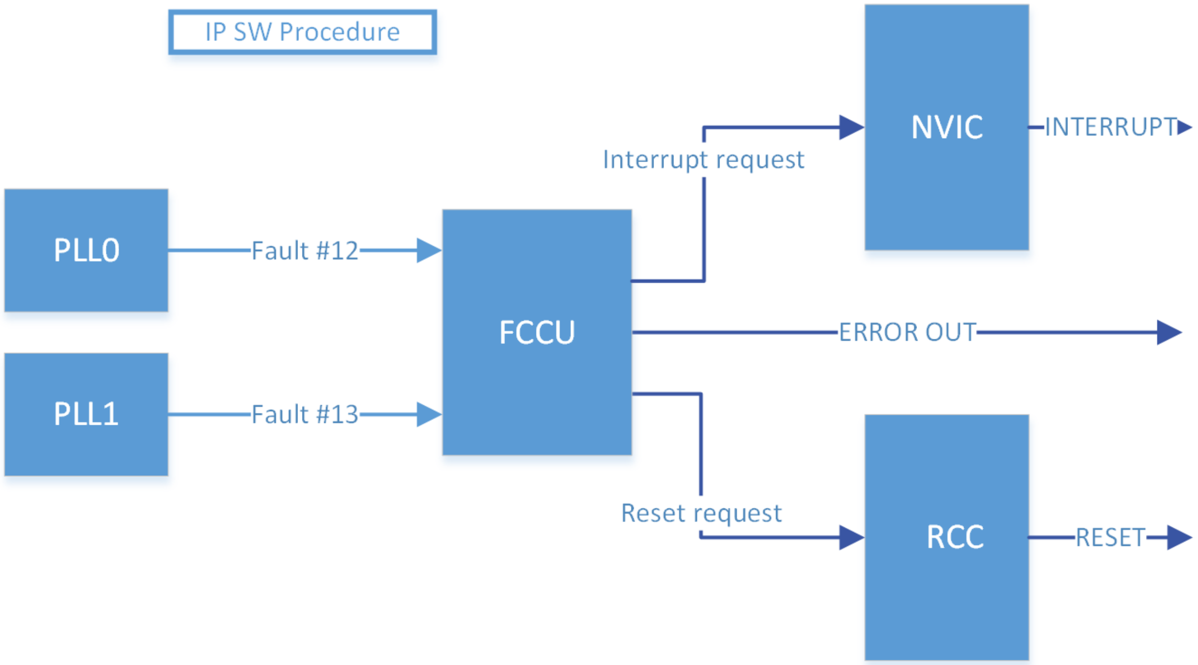
Table 15. CEM reg bit # for upsizer Cores

CEM reg bit #	Failure	Injection mechanism
0	Upsizer error - Core1 AHB	0x0
1	Upsizer error - Core2 AHB	0x1
2	Upsizer error - HSM	0x2

4.8 PLLDIG Faults

The SR5E1x embeds a dual PLL system which provides separate system and peripheral clocks. Refer to SR5E1x reference manual for further details on dual PLL.

Figure 14. PLL DIG faults



4.8.1 PLL0 loss of lock error (Fault #12)

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 software 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_S0[RFS12] 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] (writing 1) status bit before clearing the relevant FCCU_RF_S0[RFS12] bit.

4.8.2 PLL1 loss of lock error (Fault #13)

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 software 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.

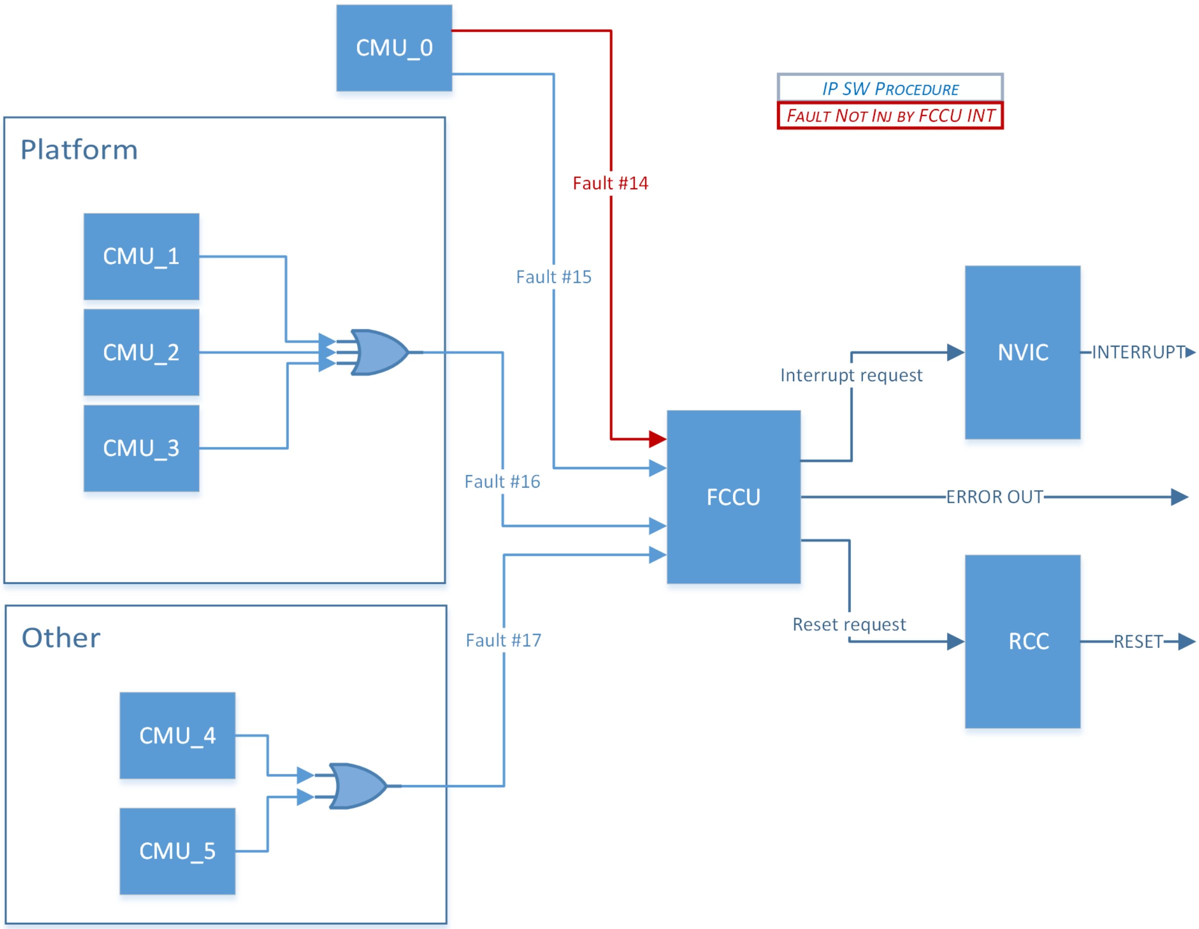
The FCCU error reaction path is verified if the FCCU_RF_S0[RFS13] and the PLLDIG_PLL1SR[LOLF] status bits are set.

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_S0[RFS13] bit.

4.9 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 these faults to the FCCU (the user must enable the CMU that is disabled by default). Refer to the SRE51x reference manual for further details on the CMUs.

Figure 15. CMU faults



4.9.1 CMU_0 error (XOSC less than IRC (Fault #14))

The CMU_0 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 can inject this fault by the FCCU fake fault interface. Only the FCCU interface is tested, the fault path between monitor and FCCU is not stimulated.

The user must set the FCCU_RFF[FRFC] field to the value 0x0E. The FCCU error reaction is verified if the FCCU_RF_S0[RFS14] is set.

The clear mechanism of the fault requires that the status FCCU_RF_S0[RFS14] bit is reset.

Note: The crystal oscillator frequency depends on applications. In case the frequency of the XOSC is less than IRC/2 it is possible to trigger a fake fault by a software procedure that configures the appropriate threshold.

4.9.2 Frequency out of range (Fault #15)

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 CMU_0 implements the XOSC monitor to calibrate the IRCOSC frequency**.**

The user can inject this fault by a software procedure that sets a misconfigured value for one of the monitoring thresholds, for example, the user can set the CMU_HFREFR[HFREF] field, of CMU_0 module, to a value lower than the correct one.

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS15] and the CMU_ISR[FHHI] status bits are set.

The user must clear the CMU_ISR[FHHI] status bit before clearing the relevant FCCU_RF_S0[RFS15] bit.

4.9.3 Sysclk frequency out of range (Fault #16)

Using the IRCOSC frequency as monitor references, the CMU_1 monitors the clock frequency used by COREs, DMAs, HRTIM, HSM, the CMU_2 monitors the clock frequency used by APB1 peripherals and TIMx (with x = 2, 6, 7, TS), the CMU_3 monitors the clock frequency used by the by APB2 peripherals and TIMx (with x = 1, 8, 4, 5, 16). 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 OR'ed signal from these CMU modules).

The user can inject this fault by a software procedure that sets a misconfigured value for one of the monitoring thresholds, for example, the user can set the CMU_ HFREFR[HFREF] field, of CMU_x module (with x = 1, 2, 3), to a value lower than the correct one. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS16] and the CMU _ISR[FHHI] status is set.

The user must clear the CMU_ ISR[FHHI] status bit before clearing the relevant FCCU_RF_S0[RFS16] bit.

4.9.4 Monitoring other internal clocks (Fault #17)

Using the IRCOSC frequency as monitor references, the CMU_4 monitors the clock frequency used by the SARADCs and the CMU_5 monitors the clock frequency used by the SDADCs. 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 OR'ed signal from these CMU modules).

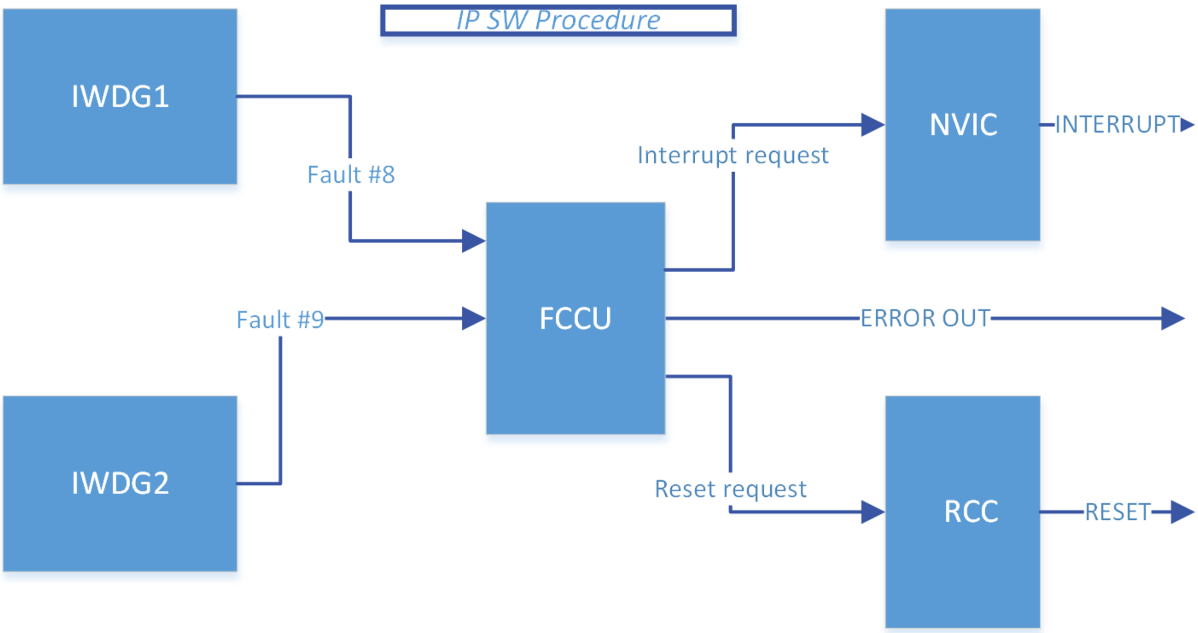
The user can inject this fault by a software procedure that sets a misconfigured value for one of the monitoring thresholds, for example, the user can set the CMU_HFREFR[HFREF] field, of CMU_y module (with y = 4, 5) to a value lower than the correct one. The FCCU error reaction path is verified if the FCCU_RF_S0[RFS17] and the CMU_ISR[FHHI] status bits are set.

The user must clear the CMU_ISR[FHHI] status bit before clearing the relevant FCCU_RF_S0[RFS17] bit.

4.10 IWDG faults

The IWDG (two instance IWDG1, IWDG2) is a watchdog peripheral that offers a combination of high safety level, timing accuracy and flexibility of use. The independent watchdog peripheral detects and solves malfunctions due to software failure, and triggers system reset when the counter reaches a given timeout value. Refer to device SR5E1x reference manual for further details on IWDG.

Figure 16. **IWDG faults**



4.10.1 Independent IWDG1 reset request (Fault #8)

If the IWDG1 reaches a timeout, the IWDG1 forwards this fault to the FCCU.

The user can inject this fault by a software procedure:

1. Enable the IWDG1
2. It is not serviced or the service routine is generated outside the allowed window

In both cases, an error is forwarded to FCCU.

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS8] status bit is set after IWDG1 timeout.

Note: *The user can clear this fault only by triggering a reset.*

4.10.2 Independent IWDG2 reset request (Fault #9)

If the IWDG2 reaches a timeout, the IWDG2 forwards this fault to the FCCU.

The user can inject this fault by a software procedure:

1. Enable the IWDG2
2. It is not serviced or the service routine is generated outside the allowed window

In both cases, an error is forwarded to FCCU.

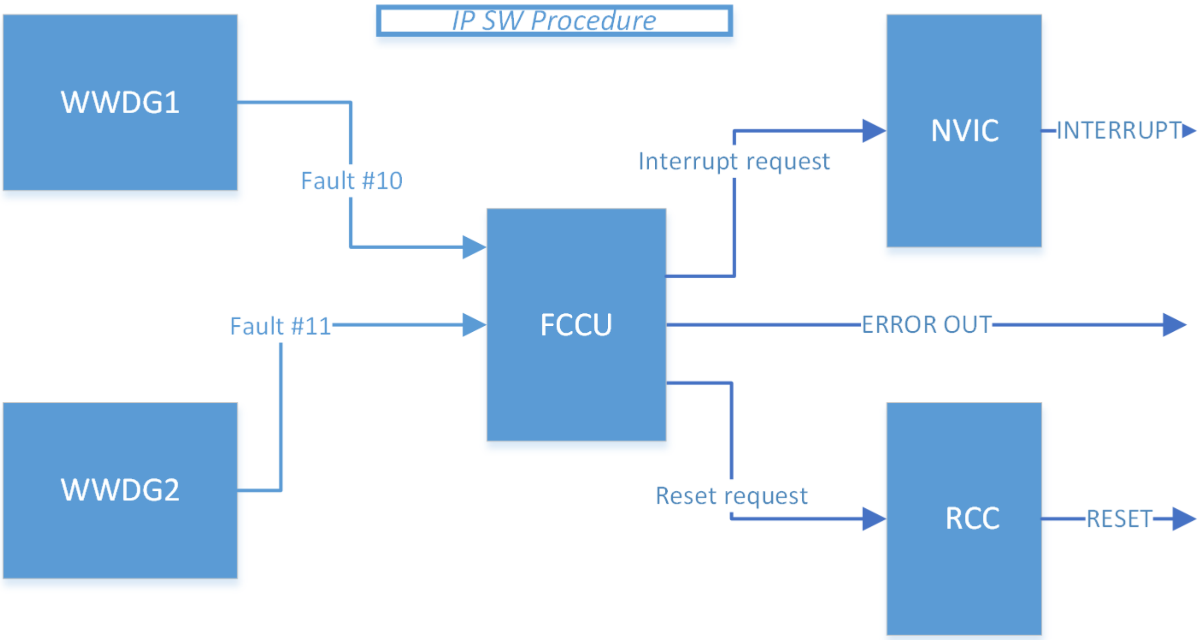
The FCCU error reaction path is verified if the FCCU_RF_S0[RFS9] status bit is set after IWDG2 timeout.

Note: *The user can clear this fault only by triggering a reset.*

4.11 WWDG faults

The WWDG is a module used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. Refer to device SR5E1x reference manual for further details on WWDG.

Figure 17. **WWDG faults**



4.11.1 Independent WWDG1 reset request (Fault #10)

If the WWDG1 reaches a timeout, the WWDG1 forwards this fault to the FCCU.

The user can inject this fault by a software procedure:

1. Enable the WWDG1
2. It is not serviced or the service routine is generated outside the allowed window

In both cases, an error is forwarded to FCCU.

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS10] status bit is set after WWDG1 timeout.

Note: *The user can clear this fault only by triggering a reset.*

4.11.2 Independent WWDG2 reset request (Fault #11)

If the WWDG2 reaches a timeout, the WWDG2 forwards this fault to the FCCU.

The user can inject this fault by a software procedure:

1. Enable the WWDG2
2. It is not serviced or the service routine is generated outside the allowed window

In both cases, an error is forwarded to FCCU.

The FCCU error reaction path is verified if the FCCU_RF_S0[RFS11] status bit is set after WWDG2 timeout.

Note: The user can clear this fault only by triggering a reset.

The Window Watchdog 1 (WWDG1) interrupt is connected to the CPU1 NVIC (NVIC1) position 0, while Window Watchdog 2 (WWDG2) interrupt is connected to the CPU2 NVIC (NVIC2) position 0.

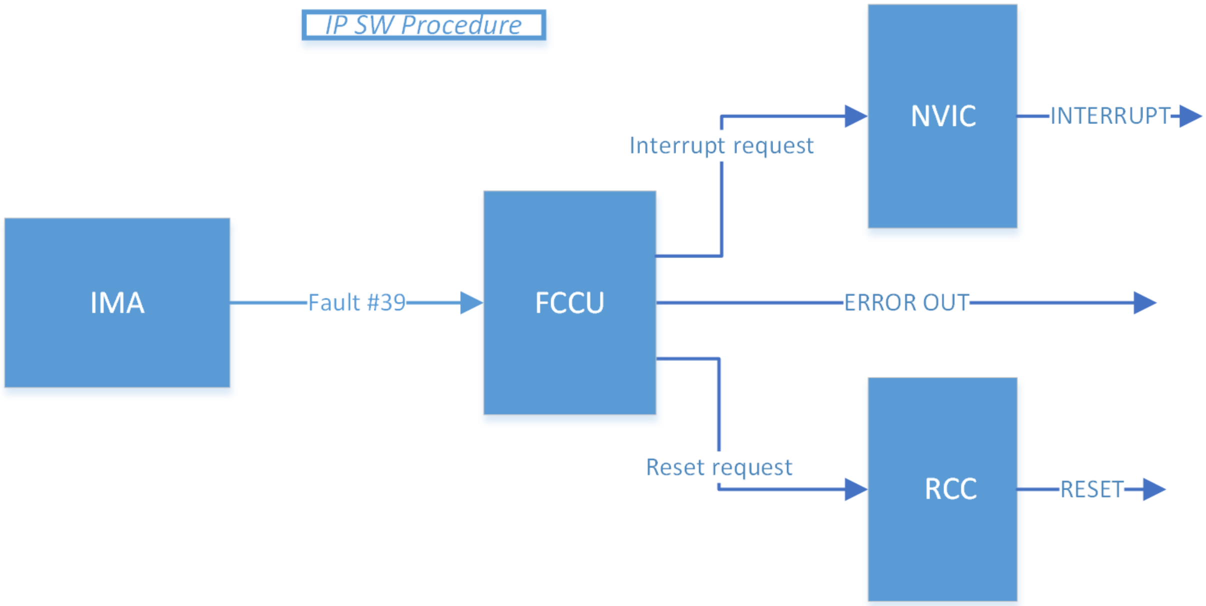
The Core1 uses the WWDG1.

The Core2 uses the WWDG2.

4.12 IMA faults

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. Refer to the SR5E1x reference manual for further details on the IMA.

Figure 18. IMA faults



4.12.1 IMA SOC active (Fault #39)

Since unwanted activation of the IMA can interfere with execution of the application. The IMA signals to the FCCU when it is wrongly activated. 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 software 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_S1[RFS7] status bit is set. The user must deactivate the IMA (IMA_SLCT[ARRAY_SLCT] = 0x00) before clearing the relevant FCCU_RF_S1[RFS7] bit.

4.13 AHBP bridge faults

The Cortex® M7 CPU uses the 32-bit AHBP bus to access AHB1, AHB2, APB1 and APB2 peripherals.

Refer to device SR5E1x reference manual for further details on the AHBP bridges.

Figure 19. **AHBP faults**

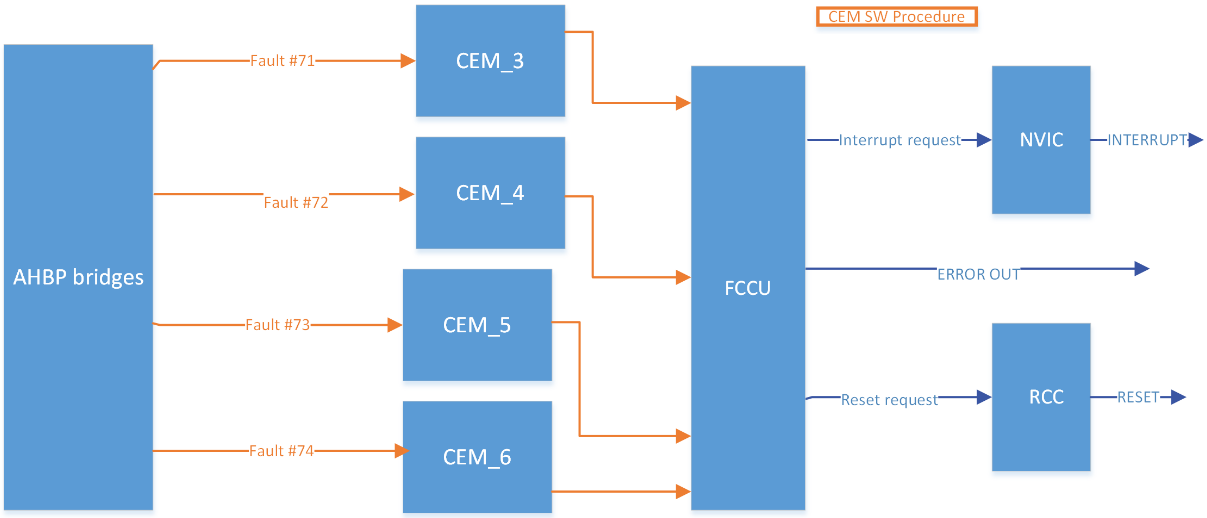
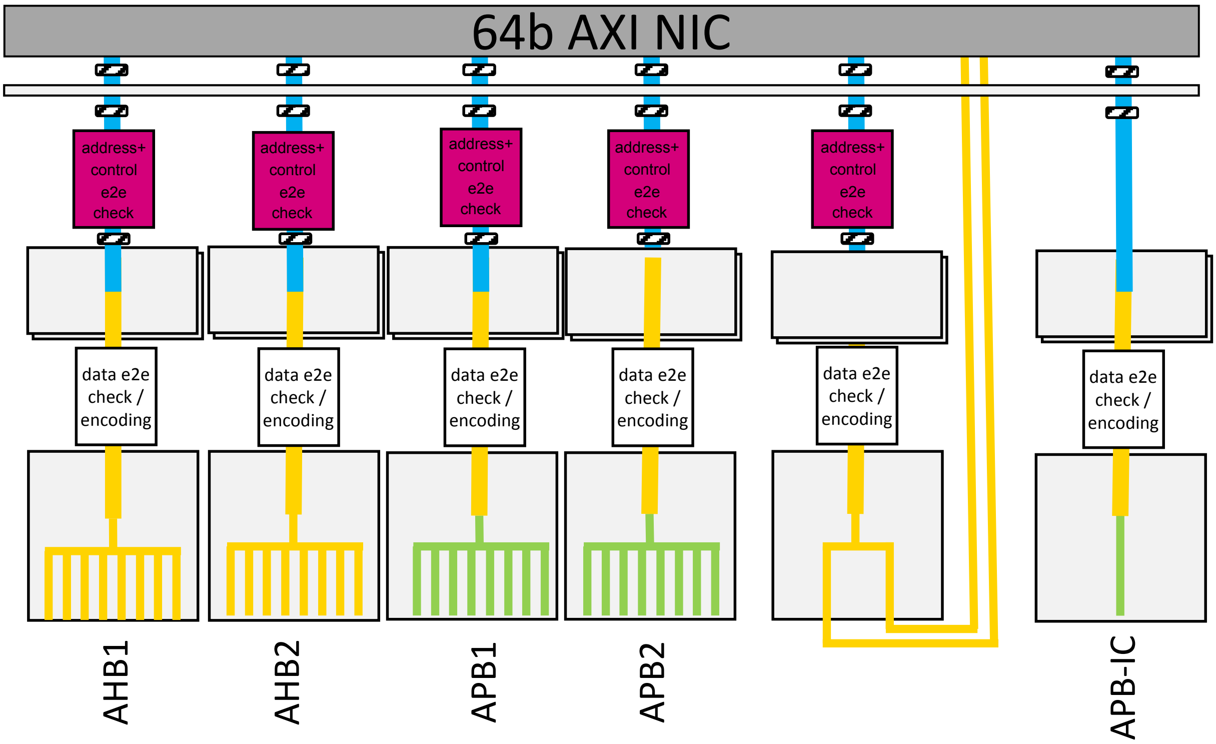


Figure 20. **AHBP e2eECC schematic**



4.13.1 e2eECC data correctable error AHBP (Fault #71, CEM_3)

The ECC logic from target port to Cores AHBP bridges (see the Figure 20), detects data correctable error. This error signal is collected by CEM_3, before arriving to the FCCU failure input #71.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 16. **CEM reg bit # for AHBP bridges correctable error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	e2eECC data correctable error Cores AHBP	0x0

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
1	e2eECC data correctable error AHB1	0x1
2	e2eECC data correctable error AHB2	0x2
3	e2eECC data correctable error APB1	0x3
4	e2eECC data correctable error APB2	0x4

4.13.2 e2eECC data uncorrectable error AHBP (Fault #72, CEM #4)

The ECC logic from target port to Cores AHBP bridges (see the Figure 20), detects data uncorrectable error. This error signal is collected by CEM_4, before arriving to the FCCU failure input #72.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 17. **CEM reg bit # for AHBP bridges uncorrectable error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	e2eECC data uncorrectable error Cores AHBP	0x0
1	e2eECC data uncorrectable error AHB1	0x1
2	e2eECC data uncorrectable error AHB2	0x2
3	e2eECC data uncorrectable error APB1	0x3
4	e2eECC data uncorrectable error APB2	0x4

4.13.3 e2eECC protocol error AHBP (Fault #73, CEM_5)

The ECC logic from target port to Cores AHBP bridges (see the Figure 20), detects protocol error cores (protocol error Cores AHBP) bits. This error signal is collected by CEM_5, before arriving to the FCCU failure input #73.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 18. **CEM reg bit # for AHBP bridges protocol error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	e2eECC data protocol error Cores AHBP	0x0
1	e2eECC protocol error AHB1	0x1
2	e2eECC protocol error AHB2	0x2
3	e2eECC protocol error APB1	0x3
4	e2eECC protocol error APB2	0x4

4.13.4 Protection violation AHPB (Fault #74, CEM_6)

The violations AHB1/AHB2/APB1/APB2 bridges occur. This error signal is collected by CEM_6, before arriving to the FCCU failure input #74.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM write CMD. For more details, see the Section 3.1: CEM module

Table 19. **CEM reg bit # for AHBP protection violation**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	Protection violation AHB1	0x0
1	Protection violation AHB2	0x1
2	Protection violation APB1	0x2

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
3	Protection violation APB2	0x3

4.14 GLUE logic faults

Figure 21. Glue logic fault #38, #45

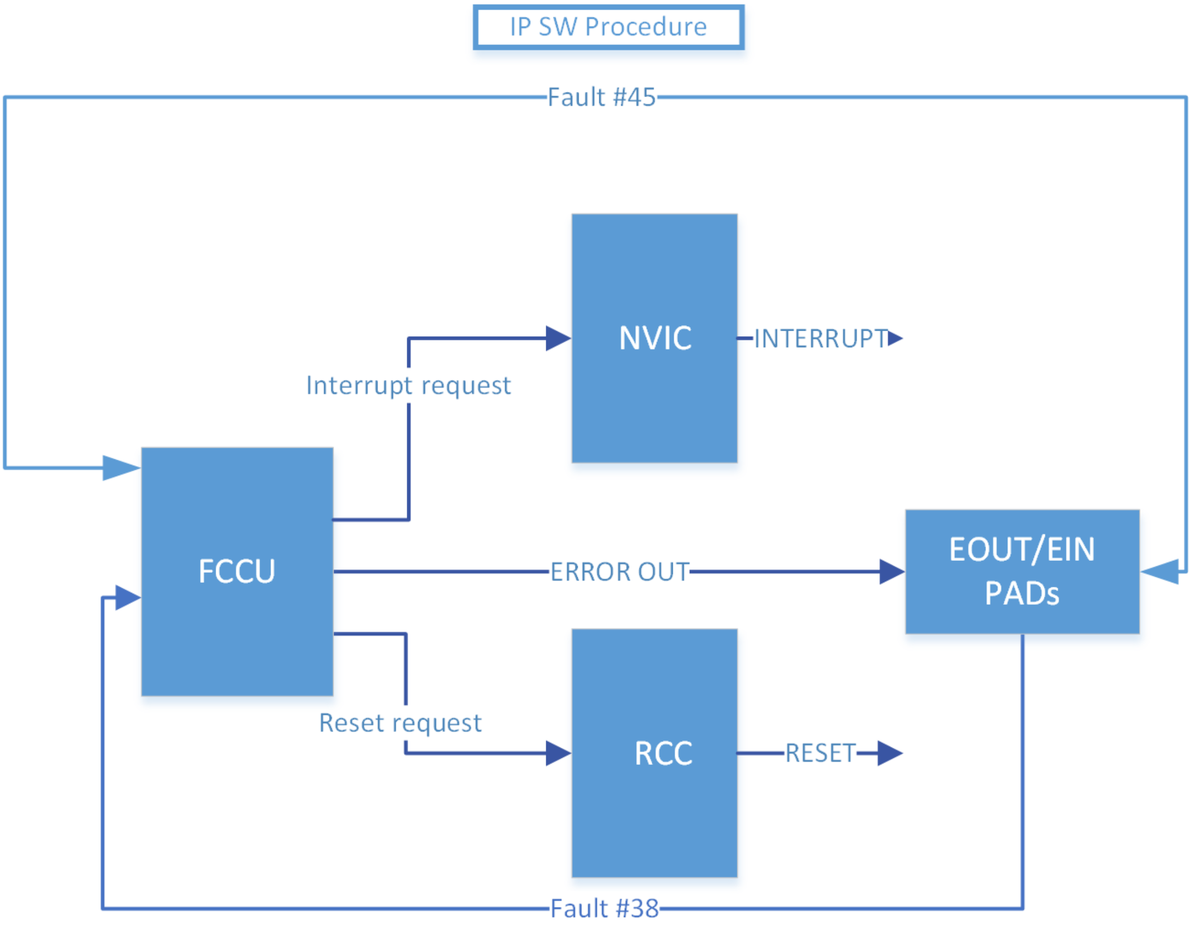
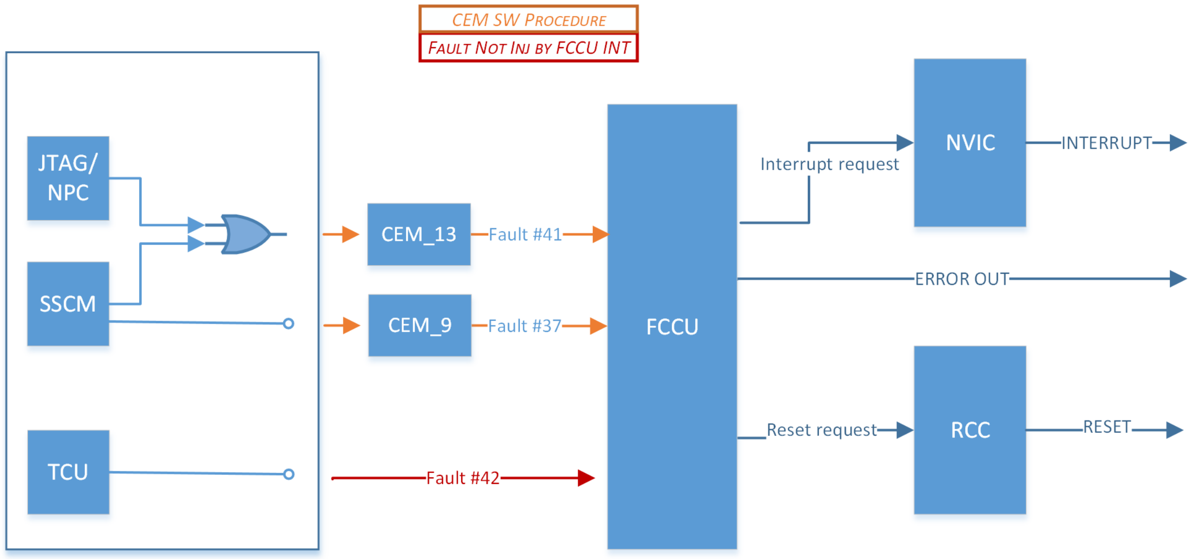


Figure 22. Glue logic faults #37, #41, #42



There are 2 sets of FCCU pin:

- Set_1 (ERRIN0): PA[9] and PA[10] pins are dedicated to FCCU for bidirectional signal
- Set_2 (ERRIN1): PD[9] and PH[13] used only for FCCU unidirectional signal

The input function FCCU_EIN0 on PA[9] is configured at startup by default.

4.14.1 Error from unidirectional input error signal (External failure to MCU (Fault #38))

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 user can inject this fault by:

1. Enabling EOUT control by FCCU (FCCU_CFG[FCCU_SET_AFTER_RESET] = 0x1)
2. Selecting a pin in the set_2
3. Configuring pin as input, alternate function and pull-down configuration

The user can clear the fault by:

1. Re-configuring the chosen pin as an analog GPIO
2. The fault clear mechanism requires that the status FCCU_RF_S1[RFS6] bit be reset

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS6] status bit is set after the step (3).

4.14.2 Error from bidirectional input error signal (External or internal failure to MCU (Fault #45))

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 user can inject this fault by:

1. Enabling EOUT control by FCCU (FCCU_CFG[FCCU_SET_AFTER_RESET] = 0x1)
2. Selecting a pin in the set_1
3. Configuring pin as input, alternate function and pull-down configuration

The user can clear the fault by:

1. Re-configuring the chosen pin as an analog GPIO
2. The fault clear mechanism requires that the status FCCU_RF_S3[RFS13] bit be reset

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS13] status bit is set after the step (3).

Note: For set_1, the error can be reported outside the MCU. For further details on the output protocol, refer to the SR5E1x RM.

4.14.3 JTAG or debug functionality out of reset, SSCM activation (Fault #41, CEM_13)

Unexpected activation of JTAG 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 monitors:

- JTAG or debug functionality during the execution of the application can interfere with the application.
- The unwanted activation of the SSCM and, if this event occurs, a dedicated glue logic forwards this fault to the FCCU.

These two error signals are collected by CEM_13, before arriving to the FCCU failure input #41.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM write CMD. For more details, see the Section 3.1: CEM module.

Table 20. **FCCU details fault**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	SPURIOUS_DEBUG_ACTIVATION	0x0
1	SPURIOUS_SSCM_ACTIVATION	0x1

4.14.4 SPURIOUS DFT (design for testability) signals ACTIVATION (Fault #42)

Unexpected test circuitry group spurious activation 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 error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x2A in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S1[RFS10] is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS10] bit be reset.

4.14.5 DCF errors at boot time error (Fault #37, CEM_9)

Faults can occur during the SSCM transfer, boot and the CEM forwards this fault to the FCCU.

RCC DCF records global system configuration used to configure:

- Cores to be started by hardware upon reset
- Lockstep configuration of the cores and DMAs
- Flash OTA mode

The user can inject a fake fault by a software procedure that sets CEM functionality: CEM channel must be enabled, the fault must be injected through the CEM write CMD. For more details, see the Section 3.1: CEM module.

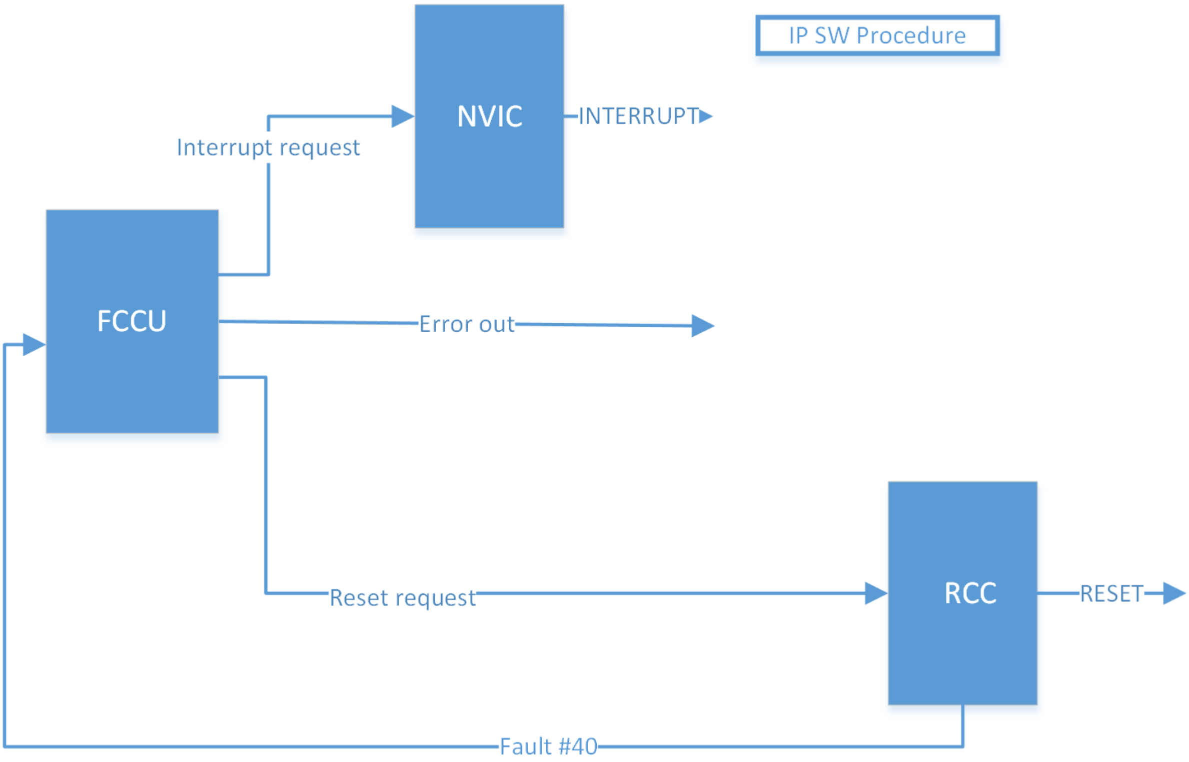
Table 21. **CEM reg bit # for SSCM transfer error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
0	SSCM_XFER_ERR	0x0
1	MEMORY REPAIR DCF SAFETY ERROR	0x1
2	TDM_DCF_SAFETY_ERR	0x2
3	RCC_DCF_SAFETY_ERR	0x3

4.15 RCC faults

The reset and clock control module (RCC) is a complex state machine that begins sequencing the SR5E1x through the initial steps of the reset process. The RCC does not execute program code, it is a state machine that centralizes the different reset sources and manages the reset sequence. Refer to the SR5E1x reference manual for further details on the RCC.

Figure 23. RCC faults



4.15.1 Transition to RCOSC in case of critical faults on clock sources (Fault #40)

The RCC can request a transition to SAFE mode, forcing SysClock to IRCOSC. In a 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 can inject this fault by a software procedure:

1. Enabling the safe mode FCCU fault enable (CIER[RCC_CIER_SAFEMODE_FE] = 1)
2. Forcing the transition of SysClock to IRCOSC (CFGR[SW] = 0x4)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS8] status bit is set.

The user can clear the fault by:

1. Clearing safe mode fault (CICR [RCC_CICR_SAFEMODE_FC] = 1)
2. The fault clear mechanism requires that the status FCCU_RF_S1[RFS18] bit be reset

4.16 HRTIM1/2 AXI bridge faults

The main system consists of an AXI bus matrix that interconnects up to subordinates, HRTIM1 and HRTIM2.

Figure 24. **HRTIM1/2 e2eECC schematic**

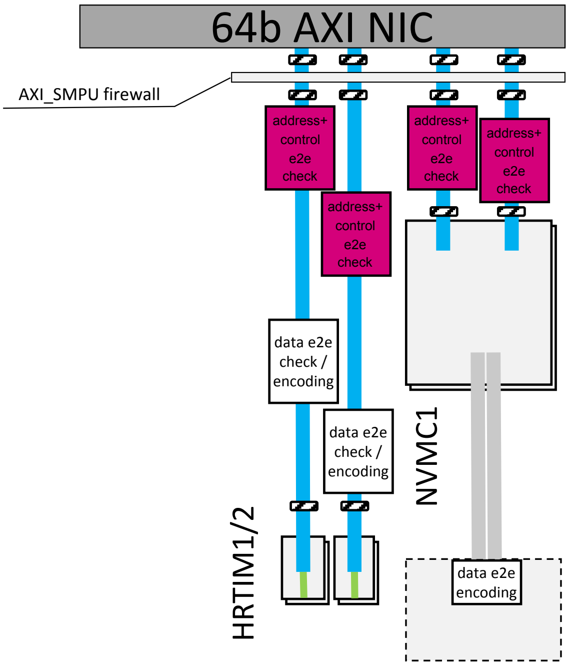
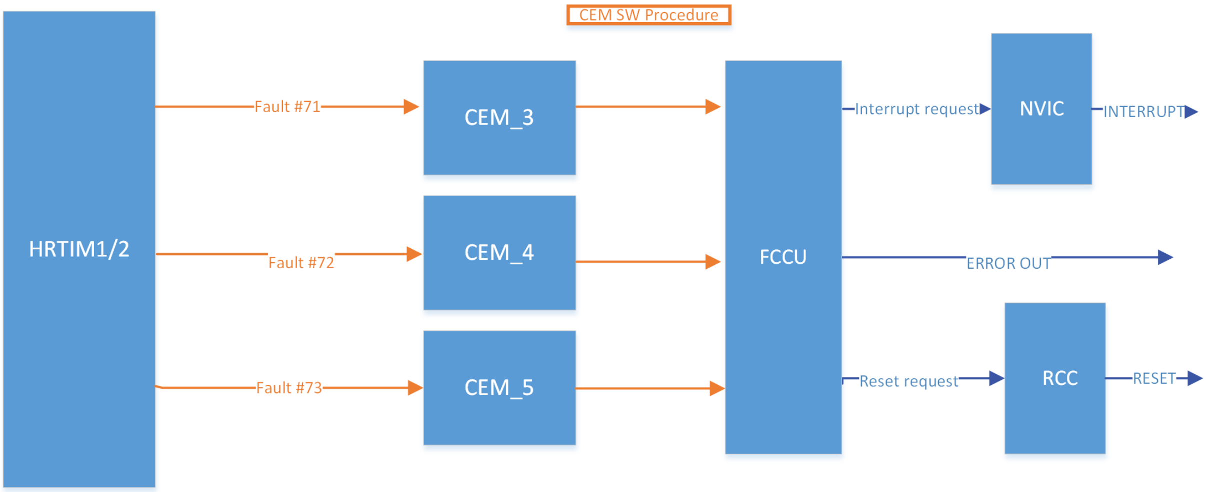

64b AXI NIC
AXI_SMPU firewall
address+ control e2e check
address+ control e2e check
address+ control e2e check
address+ control e2e check
data e2e check / encoding
data e2e check / encoding
NVMC1
HRTIM1/2
data e2e encoding

Figure 25. **HRTIM1/2 faults**



4.16.1 e2eECC data correctable error HRTIM1/2 AXI (Fault #71, CEM_3)

The ECC logic through HRTIM 1/2 bridges, detects data correctable error bits. This error signal is collected by CEM_3, before arriving to the FCCU failure input #71.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 22. **CEM reg bit # for HRTIMs correctable error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
5	e2eECC data correctable error HRTIM1 AXI	0x5
6	e2eECC data correctable error HRTIM2 AXI	0x6

4.16.2 e2eECC data uncorrectable error HRTIM1/2 AXI (Fault #72, CEM_4)

The ECC logic, through HRTIM 1/2 bridges, detects data uncorrectable error bits. This error signal is collected by CEM_4, before arriving to the FCCU failure input #72.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

Table 23. **CEM reg bit # for HRTIMs uncorrectable error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
5	e2eECC data uncorrectable error HRTIM1 AXI	0x5
6	e2eECC data uncorrectable error HRTIM2 AXI	0x6

4.16.3 e2eECC protocol error AHBP (Fault #73, CEM_5)

The ECC logic through HRTIM 1/2 bridges, detects protocol error cores (protocol error Cores AHBP) bits. This error signal is collected by CEM_5, before arriving to the FCCU failure input #73.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

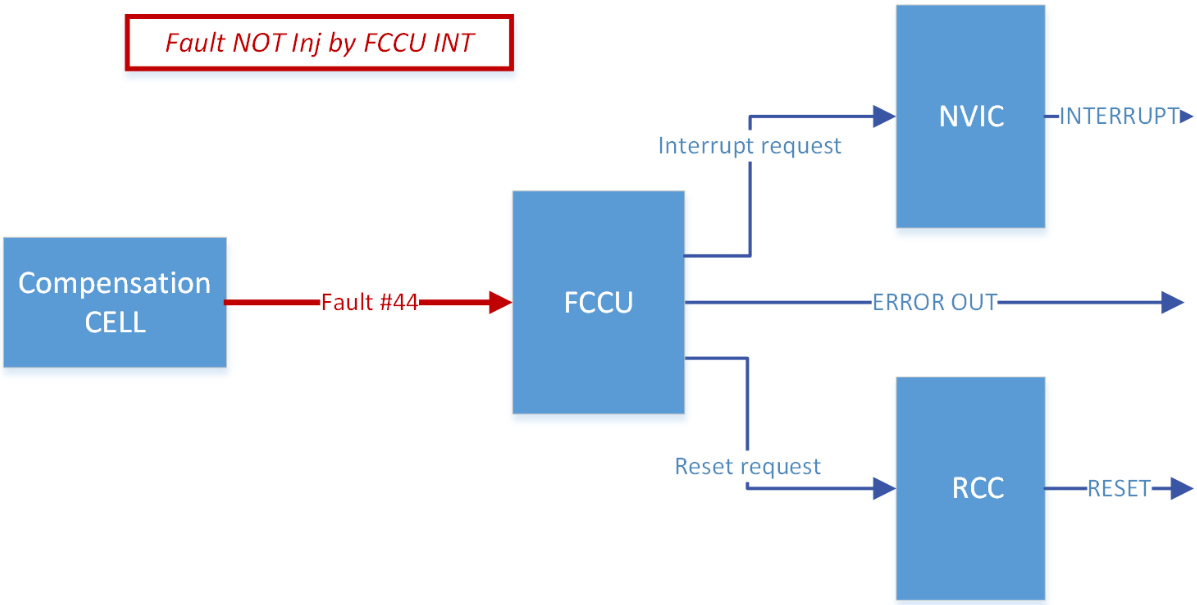
Table 24. **CEM reg bit # for HRTIMs protocol error**

CEM reg bit #	Failure	CMD[FAULT_OR_GRP_NUM] value
5	e2eECC protocol error HRTIM1 AXI	0x5
6	e2eECC protocol error HRTIM2 AXI	0x6

4.17 Compensation cells faults

Compensation cells generate 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 in the I/O buffers over temperature, pressure and voltage.

Figure 26. **Compensation cell fault**



4.17.1 Compensation disable (Fault #44)

When the compensation cell is in normal mode, it generates the compensation codes. If it exits the normal mode, the hardware detects this event and forwards it to the FCCU.

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

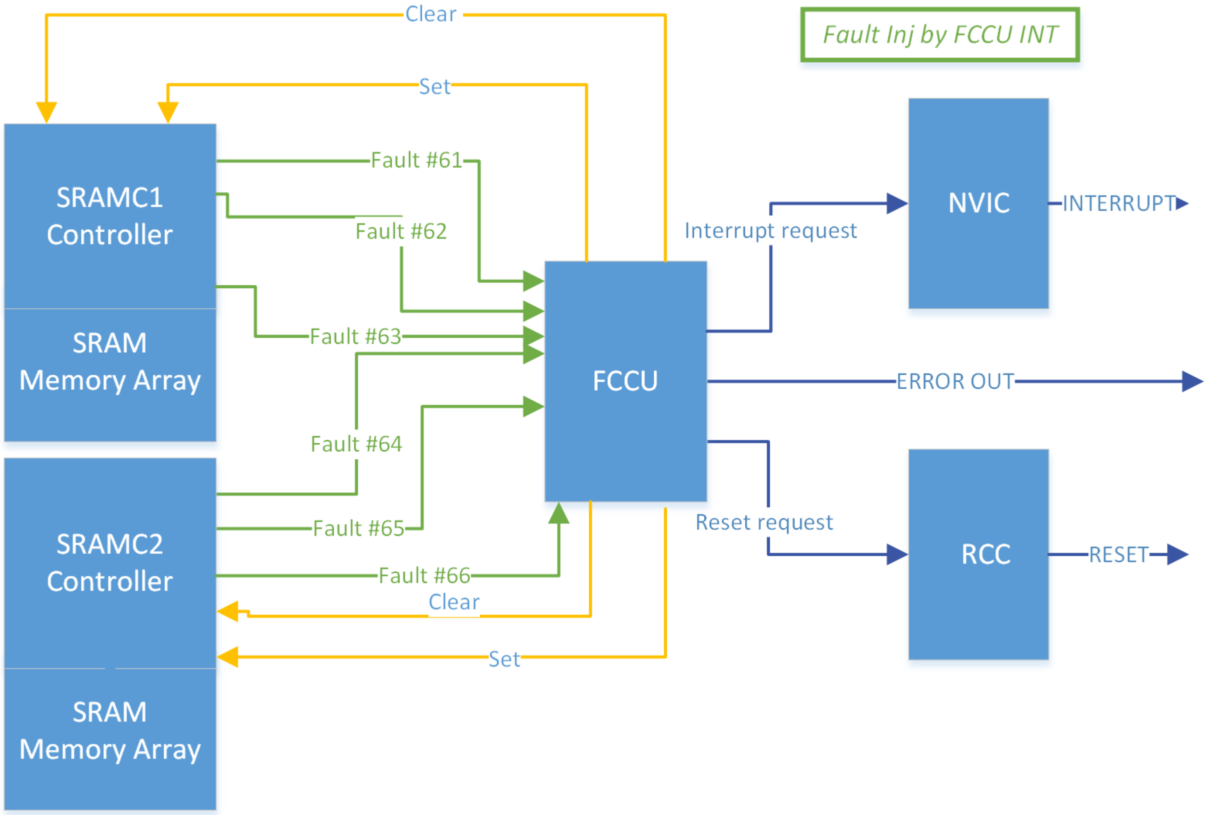
The user must set the FCCU_RFF[FRFC] field to the value 0x2C. The FCCU error reaction is verified if the FCCU_RF_S1[RFS12] is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS12] bit be reset.

4.18 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, the SRAMC1 and SRAMC2. Refer to SR5E1x reference manual for further details on the PRAMC_AX controller.

Figure 27. **PRAM faults**



4.18.1 SRAMC1 EDC after ECC error (Fault #61)

The EDC after ECC logic inside the SRAMC1 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction and forwards this fault to the FCCU.

The user can inject this fault using the FCCU fake fault injection interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault setting the FCCU_RFF[FRFC] field to the value 0x3D. The FCCU error reaction is verified if the FCCU_RF_S1[RFS29] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS29] bit be reset.

4.18.2 FCCU RAM alarm (Fault #62)

This fault signals that the PRAMC memory feedback error–alarm, indicating the SRAMC1 controller, detected a mismatch in the transmission path between the PRAM controller and the RAM array.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault setting the FCCU_RFF[FRFC] field to the value 0x3E. The FCCU error reaction is verified if the FCCU_RF_S1[RFS30] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS30] bit be reset.

4.18.3 Address/control EDC/Parity check FCCU alarm (Fault #63)

The EDC after ECC logic inside the SRAMC1 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for address/control.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault setting the FCCU_RFF[FRFC] field to the value 0x3F. The FCCU error reaction path is verified if the FCCU_RF_S1[RFS31] status bi is set.

The fault clear mechanism requires that the status FCCU_RF_S1[RFS31] bit be reset.

4.18.4 SRAMC2 EDC after ECC error (Fault #64)

The EDC after ECC logic inside the SRAMC2 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction and forwards this fault to the FCCU.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault setting the FCCU_RFF[FRFC] field to the value 0x40. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS0] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S2[RFS0] bit be reset.

4.18.5 FCCU RAM alarm (Fault #65)

PRAMC memory feedback error–alarm indicating the SRAMC2 controller detected a mismatch in the transmission path between the PRAM controller and the RAM array.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault setting the FCCU_RFF[FRFC] field to the value 0x41. The FCCU error reaction is verified if the FCCU_RF_S2[RFS1] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S2[RFS1] bit be reset.

4.18.6 Address/control EDC/Parity check FCCU alarm (Fault #66)

The EDC after ECC logic inside the SRAMC2 detects a hardware fault in the ECC logic resulting in a corrupted ECC correction for Address/Control.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fake fault setting the FCCU_RFF[FRFC] field to the value 0x42. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS3] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S2[RFS3] bit be reset.

4.19 AXI watchdog

ECC is also applied to control signals and address decoding, to verify the data reaches all the intended clients, from all possible connections to these clients and the intended operation is performed on the target address.

To avoid system stalls, the transactions are monitored by an AXI sniffer and watchdog.

Refer to SR5E1x reference manual for further details on the AXI sniffer and watchdog.

4.19.1 AXI sniffer watchdog - OR all (Fault #70)

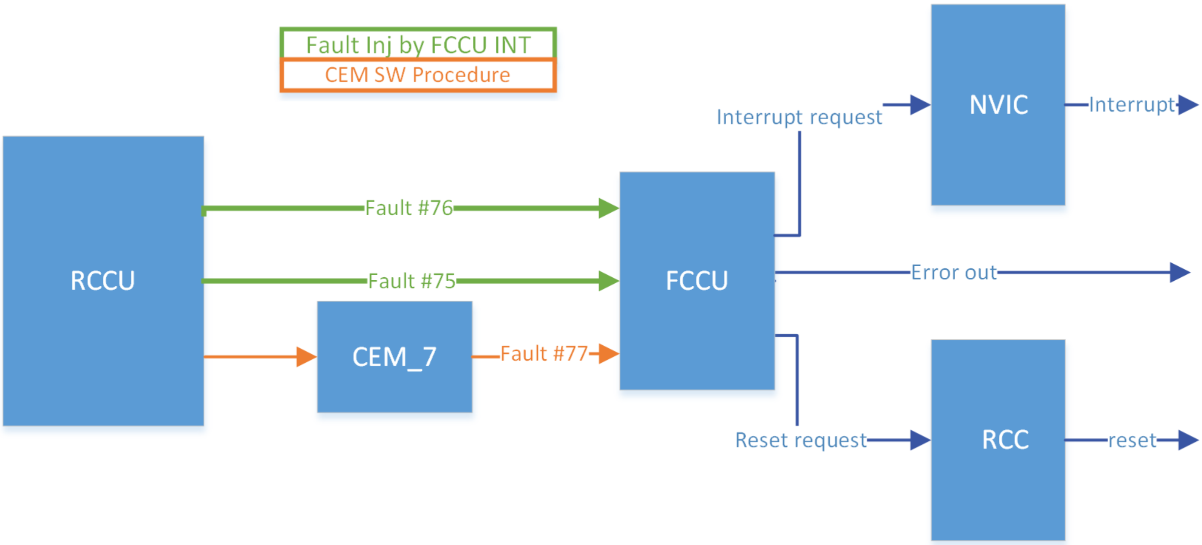
The error injection mechanism is only available within the FCCU fake fault interface (MON3). The error path between safety monitor and FCCU is not stimulated.

The user injects the fake fault by setting the error code 0x46 in the FCCU_RFF[FRFC] field. The FCCU error reaction is verified if the FCCU_RF_S2[RFS6] is set.

4.20 RCCU faults

The RCCU structure compares a set of equivalent input signals from two different sources (a primary set of inputs from main core, and a secondary set of inputs from the checker core). In case of a mismatch in a compared signal, a fault is forwarded to the FCCU.

Figure 28. RCCU faults



4.20.1 RCCUS for Cores lockstep (Fault #75)

This FCCU fault channel receives the fault indication from the Core Lockstep RCCU.

Since unwanted deactivation of the lockstep configuration cores can interfere with the execution of the application, the RCCUs signal to the FCCU when this event occurs.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user can inject a fault by setting the FCCU_RFF[FRFC] field to the value 0x4b. The FCCU error reaction path is verified if the FCCU_RF_S2[RFS11] status bit is set. The fault clear mechanism requires that the status FCCU_RF_S2[RFS11] bit be reset.

4.20.2 RCCUS for DMA lockstep (Fault #76)

This FCCU fault channel receives the fault indication from the DMA Lockstep RCCU.

Since unwanted deactivation of the lockstep configuration DMAs can interfere with the execution of the application, the RCCUs signal to the FCCU when this event occurs.

The user can inject this fault by the FCCU fake fault interface. The error path between safety monitor and FCCU is stimulated.

The user must enable the clock for DMA1 and DMA2 controller (AHB1LENR[DMA2] = 0x1 AHB1LENR[DMA]) and set the FCCU_RFF[FRFC] field to the value 0x4C.

The FCCU error reaction path is verified if the FCCU_RF_S2[RFS12] status bit is set.

The fault clear mechanism requires that the status FCCU_RF_S2[RFS12] bit be reset.

4.20.3 RCCU others - from duplication of AXI targets (Fault #77, CEM_7)

This FCCU fault channel receives the fault indication from the checker of the targets dataless duplication in lockstep.

The user can inject a fake fault by a software procedure that sets CEM functionality: the fault must be injected through the CEM CMD register. For more details, see the Section 3.1: CEM module.

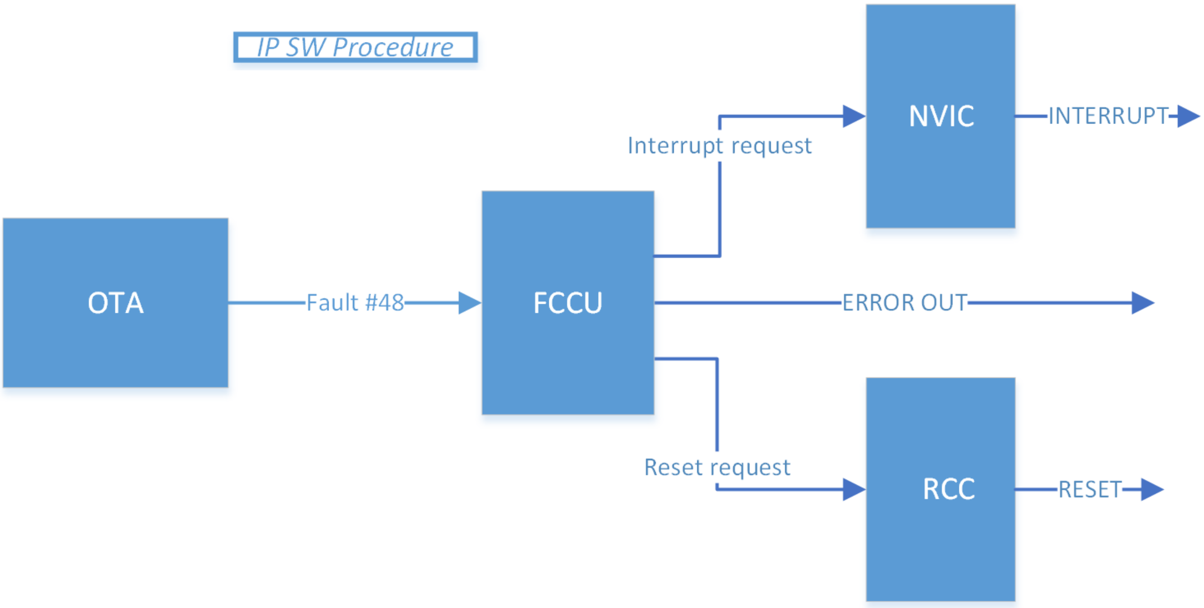
Table 25. CEM reg bit # for RCCU vs AXI targets

CEM reg bit #	Failure	Injection mechanism
0	AHB1_Bridge_Alarm	0x0
1	AHB2_Bridge_Alarm	0x1
2	APB1_Bridge_Alarm	0x2
3	APB2_Bridge_Alarm	0x3
4	AHBS_Bridge_Alarm	0x4
5	HRTIM1_Bridge_Alarm	0x5
6	HRTIM2_Bridge_Alarm	0x6
7	NVMC1_Bridge_Alarm	0x7
8	NVMC2_Bridge_Alarm	0x8
9	RAMC1_Bridge_Alarm	0x9
10	RAMC2_Bridge_Alarm	0xA

4.21 OTA faults

4.21.1 OTA change state alarm (Fault #48)

Figure 29. OTA fault



The NVMPC detects the OTA-X1 swapping using two swap signals along with ota_enable. If there is ota_enabled asserted while the swap bits are not equal, the NVMPC generates the OTA-X1 SWAP error to FCCU. There is no specific response to controller for this error. The FLT* registers have dedicated bits for OTA-X1 SWAP error. Additionally, OTA-X1 SWAP alarm is asserted on access to NVM system address which is not remapped for OTA (for example, mirrored NVM system address).

The user can inject this fault by:

1. Enabling the error forwarding to FCCU (FLTENA[NVM_FLTENA_NVMPCENSWAP] = 0x1, of NVM1/2 module)
2. Forcing the error latching to check the error reporting path (FLTFRC[NVM_FLTFRC_NVMPCENSWAP] = 0x1)

The FCCU error reaction path is verified if the FCCU_RF_S1[RFS16] status bit is set after step.

The user can clear the fault by:

1. Clearing the fault source error (FLTSCR[NVM_FLTSCR_NVMPCENSWAP] = 0x1)
2. Clearing the relevant FCCU_RF_S1[RFS16] bit

5 Example application

An example application 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 application:

- Initialize the MCU (clocks and monitors):
    1. Reset the RCC and clear its registers
    2. Initialize the FCCU for all the testable faults:
        a. Enabled the FCCU input
        b. Set the FCCU input as software recoverable
        c. No reset action
        d. Enable interrupt with timeout (FCCU state machine goes to alarm state in case of fault)
        e. Enable output pins (only for external pin test)
- For each fault identified as "Testable" the software:
    1. Verifies the FCCU status before injection:
        a. If it is in normal state, proceed, otherwise, if it is in alarm or fault state, stop
        b. The FCCU error reaction path is verified if the FCCU_RF_Sy [RFS x] (with y = 0, 1, 2; with from x = 0 to 81) is reset. (The value of x and y depends on the error stimulated)
    2. Injects it (using the monitor's registers or using fake fault injection or a software procedure or CEM software procedure, if possible)
    3. Verifies the FCCU status after injection
        a. If it is in alarm state proceed, otherwise, if it is in normal or fault state, stop
        b. The FCCU error reaction path is verified if the FCCU_RF_Sy [RFS x] (with y = 0, 1, 2; with from x = 0 to 81) is set. (The value of x and y depends on the error stimulated)
    4. Checks the FCCU reaction (IRQ and relevant FCCU fault flag)
    5. Clears the monitor fault and the FCCU alarm state
    6. Verifies the FCCU status after recovering from alarm
        a. If it is in normal state, proceed, otherwise, if it is in alarm or fault state, stop

6 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 SR5E1x devices - describes the FCCU faults inputs and how to verify their reaction path.

Appendix A Reference documents

Table 26. Reference documents

Document name	ID	Document title
DS13808	035656	SR5 E1 line of Stellar electrification MCUs — 32-bit Arm® Cortex®-M7 automotive MCU 2x cores, 300 MHz, 2 MB flash, rich analog, 104 ps 24-ch high-resolution timer, HSM, and ASIL D
RM0483	034781	SR5E1x 32-bit Arm® Cortex®-M7 architecture microcontroller for electrical vehicle applications
TN1404	036137	SR5E1x IO definition (signal description and input multiplexing tables) and device identification registers
AN5862	036664	SR5E1x Safety Manual

Revision history

Table 27. Document revision history

Date	Version	Changes
05-Feb-2024	1	Initial release.

Contents

1 General information 2
1.1 Acronyms 2
2 Overview 4
3 FCCU fault injection, clearing and fake fault interface 13
3.1 CEM module 14
4 Faults description 16
4.1 PMC_DIG faults 16
4.1.1 Temperature detector out of range (Fault #0) 16
4.1.2 Voltage out of range from LVDs (Fault #1) 17
4.1.3 Voltage out of range from HVDs (Fault #2) 17
4.1.4 Digital PMC initialization error during DCF data load (Fault #3). 17
4.1.5 Digital PMC voltage detector BIST (Fault #4) 17
4.2 FLASH/PFLASHC faults. 17
4.2.1 Flash fatal error (Fault #5). 18
4.2.2 Flash reset error (Fault #6) 18
4.2.3 Flash read reference error (Fault #7) 18
4.2.4 NVMC1 EDC after ECC for code FLASH (Fault #51) 19
4.2.5 NVMC1 EDC after ECC for data FLASH (Fault #52). 19
4.2.6 NVMC1 FLASH memory access fault (Fault #53) 19
4.2.7 NVMC1 address feedback error (Fault #54) 19
4.2.8 NVMC2 EDC after ECC for code FLASH (Fault #55) 19
4.2.9 NVMC2 FLASH memory access fault (Fault #57) 20
4.2.10 NVMC2 address feedback error (Fault #58) 20
4.2.11 e2eECC NVMC1 protocol error (Fault #59). 20
4.2.12 e2eECC NVMC2 protocol error (Fault #60). 20
4.3 STCU3 faults 20
4.3.1 BIST result - wrong signature (STCU3 recoverable fault), (Fault #19). 21
4.3.2 SPURIOUS STCU3 activation (Fault #41, CEM_13, bit#2) 21
4.4 DMA faults 21
4.4.1 DMA lock/split change state alarm (Fault #47) 23
4.4.2 e2eECC data correctable error DMA1/2 AHB Memory/Peripheral (Fault #67, CEM _0) . . 23
4.4.3 e2eECC data uncorrectable error DMA1/2 AHB memory/peripheral (Fault #68, CEM_1) . 24
4.4.4 e2eECC protocol error DMA1/2 AHB memory/peripheral (Fault #69, CEM_3) 24
4.4.5 e2eECC upsizer error DMA1/2 AHB memory/peripheral (Fault #81, CEM_14) 24
4.5 MEMU2 faults 25
4.5.1 MEMU2 SYS Trigger fault 25

4.5.2 MEMU2 PERIPH Trigger fault 26
4.5.3 MEMU2 NVM Trigger fault 27
4.5.4 System RAM FIF0 overflow (Fault #34, CEM_11) 29
4.5.5 Peripheral RAM FIF0 overflow (Fault #35, CEM_12) 29
4.5.6 Flash FIF0 overflow (Fault #36) 29
4.6 SMPU faults 30
4.6.1 SMPU region violation (Fault #49) 30
4.6.2 SMPU monitors that no signal is altered by the SMPU logic (Fault #50) 30
4.7 Cores (Core1/2, HSM) faults 30
4.7.1 Core lock/split change state alarm (Fault #46) 32
4.7.2 e2eECC data correctable error Core1/2 AXIM/AHBM (Fault #67, CEM_0) 32
4.7.3 e2eECC data uncorrectable error Core1/2 - AXIM/AHBM and HSM - AHB (Fault #68, CEM_1) 32
4.7.4 e2eECC protocol error Core1/2, HSM (Fault #69, CEM_2) 33
4.7.5 Core1 lockup error (Fault #78) 33
4.7.6 Core2 lockup error (Fault #79) 33
4.7.7 Core1/2 address feedback err and EDC after ECC (Fault #80, CEM_8) 33
4.7.8 Upsizer error Core1/2, HSM (Fault #81, CEM_14) 34
4.8 PLLDIG Faults 35
4.8.1 PLL0 loss of lock error (Fault #12) 35
4.8.2 PLL1 loss of lock error (Fault #13) 35
4.9 CMU faults 35
4.9.1 CMU_0 error (XOSC less than IRC (Fault #14)) 36
4.9.2 Frequency out of range (Fault #15) 36
4.9.3 Sysclk frequency out of range (Fault #16) 37
4.9.4 Monitoring other internal clocks (Fault #17) 37
4.10 IWDG faults 37
4.10.1 Independent IWDG1 reset request (Fault #8) 37
4.10.2 Independent IWDG2 reset request (Fault #9) 38
4.11 WWDG faults 38
4.11.1 Independent WWDG1 reset request (Fault #10) 38
4.11.2 Independent WWDG2 reset request (Fault #11) 39
4.12 IMA faults 39
4.12.1 IMA SOC active (Fault #39) 39
4.13 AHBP bridge faults 39
4.13.1 e2eECC data correctable error AHBP (Fault #71, CEM_3) 40
4.13.2 e2eECC data uncorrectable error AHBP (Fault #72, CEM #4) 41
4.13.3 e2eECC protocol error AHBP (Fault #73, CEM_5) 41

4.13.4 Protection violation AHPB (Fault #74, CEM_6) 41
4.14 GLUE logic faults. 42
4.14.1 Error from unidirectional input error signal (External failure to MCU (Fault #38)). 43
4.14.2 Error from bidirectional input error signal (External or internal failure to MCU (Fault #45)) 43
4.14.3 JTAG or debug functionality out of reset, SSCM activation (Fault #41, CEM_13) 44
4.14.4 SPURIOUS DFT (design for testability) signals ACTIVATION (Fault #42). 44
4.14.5 DCF errors at boot time error (Fault #37, CEM_9) 44
4.15 RCC faults 44
4.15.1 Transition to RCOSC in case of critical faults on clock sources (Fault #40). 45
4.16 HRTIM1/2 AXI bridge faults 45
4.16.1 e2eECC data correctable error HRTIM1/2 AXI (Fault #71, CEM_3) 46
4.16.2 e2eECC data uncorrectable error HRTIM1/2 AXI (Fault #72, CEM_4) 47
4.16.3 e2eECC protocol error AHBP (Fault #73, CEM_5) 47
4.17 Compensation cells faults 47
4.17.1 Compensation disable (Fault #44). 48
4.18 PRAM faults 48
4.18.1 SRAMC1 EDC after ECC error (Fault #61) 48
4.18.2 FCCU RAM alarm (Fault #62). 48
4.18.3 Address/control EDC/Parity check FCCU alarm (Fault #63) 49
4.18.4 SRAMC2 EDC after ECC error (Fault #64) 49
4.18.5 FCCU RAM alarm (Fault #65). 49
4.18.6 Address/control EDC/Parity check FCCU alarm (Fault #66) 49
4.19 AXI watchdog. 49
4.19.1 AXI sniffer watchdog - OR all (Fault #70) 49
4.20 RCCU faults 50
4.20.1 RCCUS for Cores lockstep (Fault #75) 50
4.20.2 RCCUS for DMA lockstep (Fault #76). 50
4.20.3 RCCU others - from duplication of AXI targets (Fault #77, CEM_7) 50
4.21 OTA faults 51
4.21.1 OTA change state alarm (Fault #48) 51
5 Example application 53
6 Summary 54
Appendix A Reference documents. 55
Revision history 56

List of tables

Table 1. Device summary . . . 1
Table 2. Acronyms . . . 2
Table 3. FCCU failure inputs . . . 5
Table 4. FCCU error path and monitors . . . 13
Table 5. CEM reg bit # for spurious STCU3 activation . . . 21
Table 6. CEM reg bit # for DMA correctable error . . . 24
Table 7. CEM reg bit # for DMA uncorrectable error . . . 24
Table 8. CEM reg bit # for DMA protocol error . . . 24
Table 9. CEM reg bit # for DMA upsizer . . . 24
Table 10. CEM reg bit # for memories error . . . 27
Table 11. CEM reg bit # for Cores correctable error . . . 32
Table 12. CEM reg bit # for Cores uncorrectable error . . . 33
Table 13. CEM reg bit # for Cores protocol error . . . 33
Table 14. CEM reg bit # for Cores address feedback error . . . 34
Table 15. CEM reg bit # for upsizer Cores . . . 34
Table 16. CEM reg bit # for AHBP bridges correctable error . . . 40
Table 17. CEM reg bit # for AHBP bridges uncorrectable error . . . 41
Table 18. CEM reg bit # for AHBP bridges protocol error . . . 41
Table 19. CEM reg bit # for AHBP protection violation . . . 41
Table 20. FCCU details fault . . . 44
Table 21. CEM reg bit # for SSCM transfer error . . . 44
Table 22. CEM reg bit # for HRTIMs correctable error . . . 46
Table 23. CEM reg bit # for HRTIMs uncorrectable error . . . 47
Table 24. CEM reg bit # for HRTIMs protocol error . . . 47
Table 25. CEM reg bit # for RCCU vs AXI targets . . . 51
Table 26. Reference documents . . . 55
Table 27. Document revision history . . . 56

List of figures

Figure 1. FCCU monitor to reaction path . . . 4
Figure 2. FCCU inner . . . 13
Figure 3. CEM error reaction path. . . . 14
Figure 4. PMC_DIG faults . . . 16
Figure 5. FLASH/PFLASHC faults . . . 18
Figure 6. STCU3 faults . . . 21
Figure 7. DMA faults . . . 22
Figure 8. DMA e2eECC schematic . . . 23
Figure 9. MEMU2 faults. . . . 25
Figure 10. SMPU faults . . . 30
Figure 11. Cores faults . . . 31
Figure 12. Cores e2eECC schematic . . . 31
Figure 13. Cores of system architecture . . . 32
Figure 14. PLL DIG faults . . . 35
Figure 15. CMU faults . . . 36
Figure 16. IWDG faults . . . 37
Figure 17. WWDG faults . . . 38
Figure 18. IMA faults . . . 39
Figure 19. AHBP faults . . . 40
Figure 20. AHBP e2eECC schematic . . . 40
Figure 21. Glue logic fault #38, #45 . . . 42
Figure 22. Glue logic faults #37, #41, #42 . . . 43
Figure 23. RCC faults . . . 45
Figure 24. HRTIM1/2 e2eECC schematic . . . 46
Figure 25. HRTIM1/2 faults . . . 46
Figure 26. Compensation cell fault . . . 47
Figure 27. PRAM faults . . . 48
Figure 28. RCCU faults . . . 50
Figure 29. OTA fault . . . 51