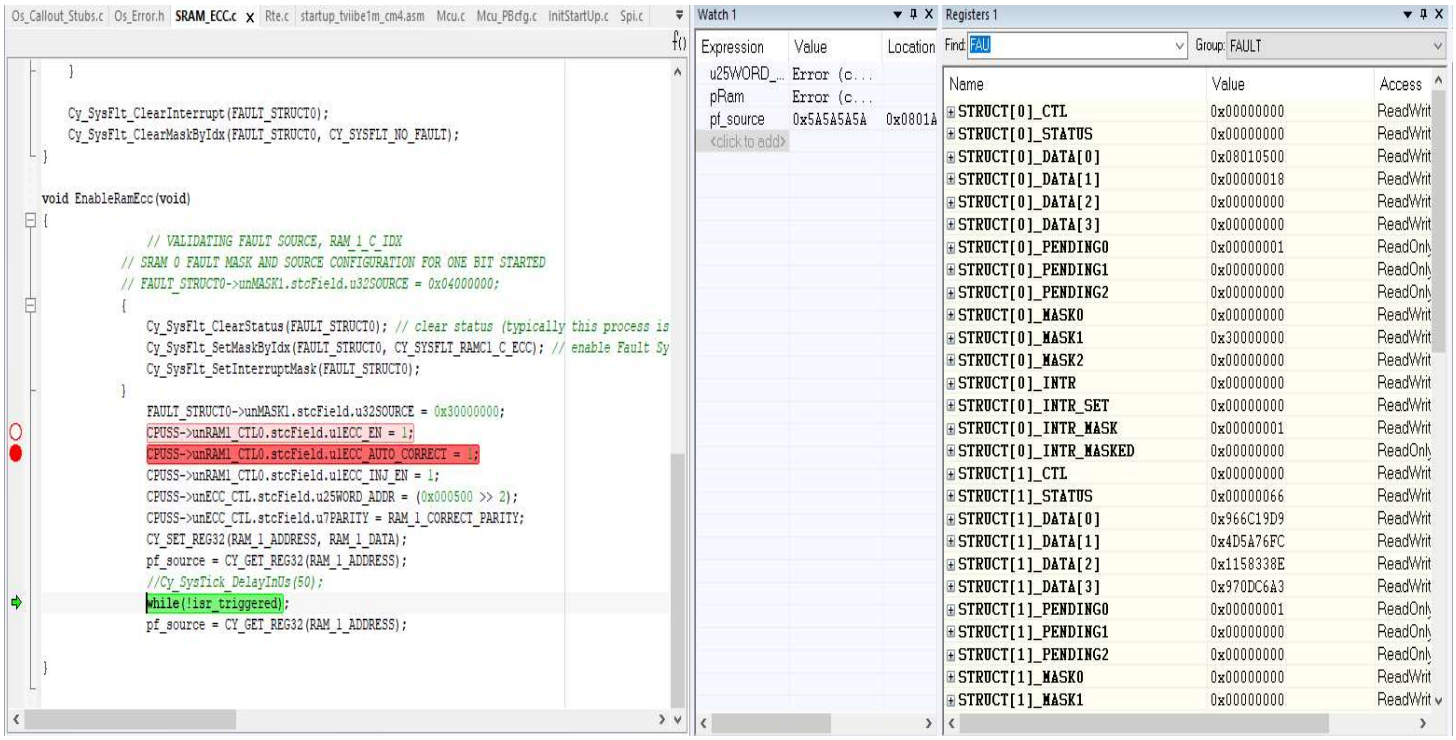
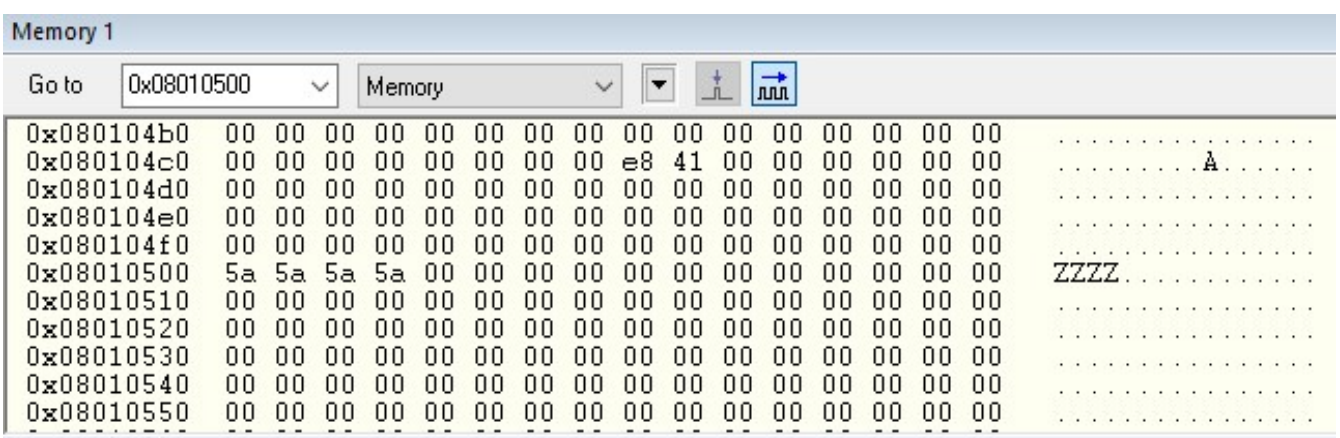
Test Report for ECC

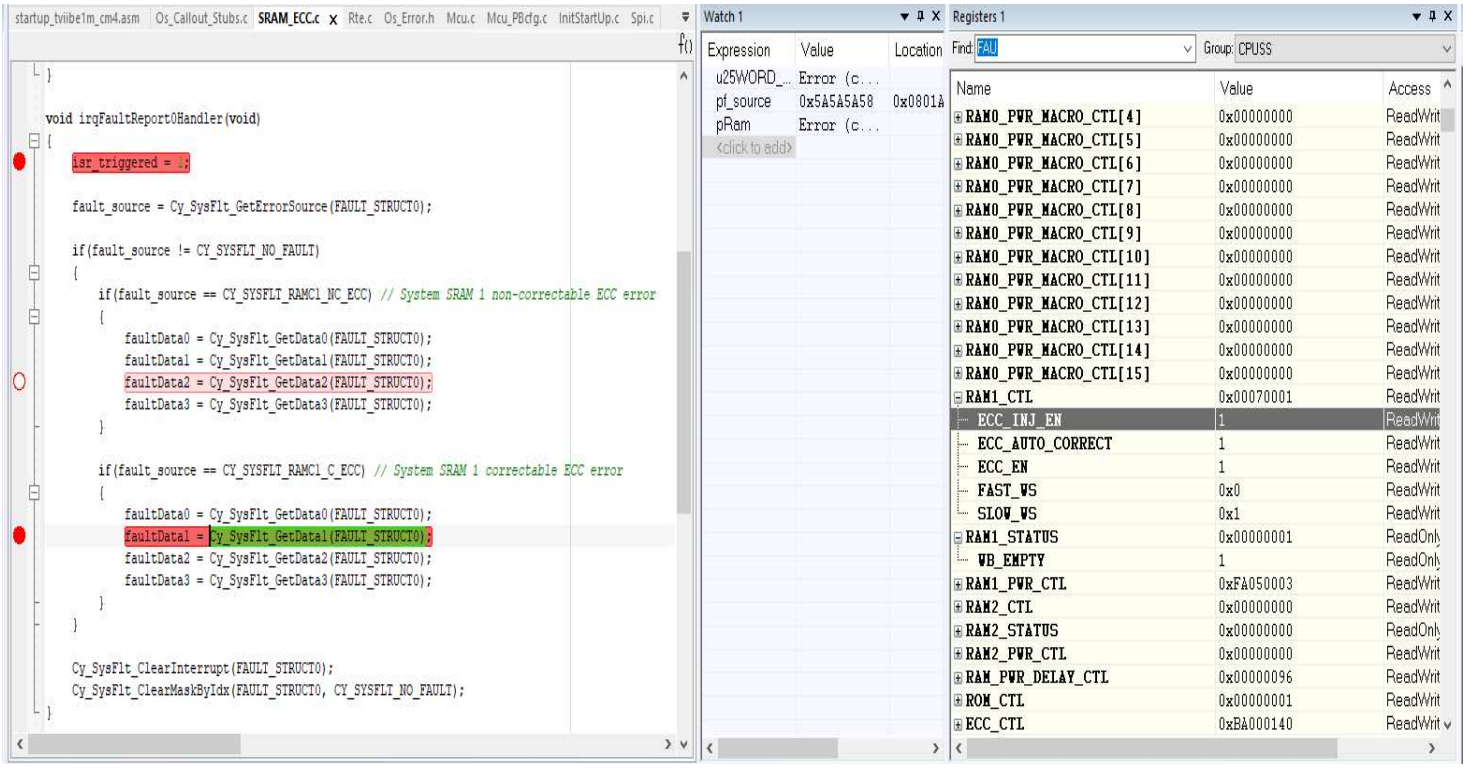
Test case 1: Writing 0x5A5A5A5A into address of SRAM1 0x08010500 with correct parity when ECC is enabled.

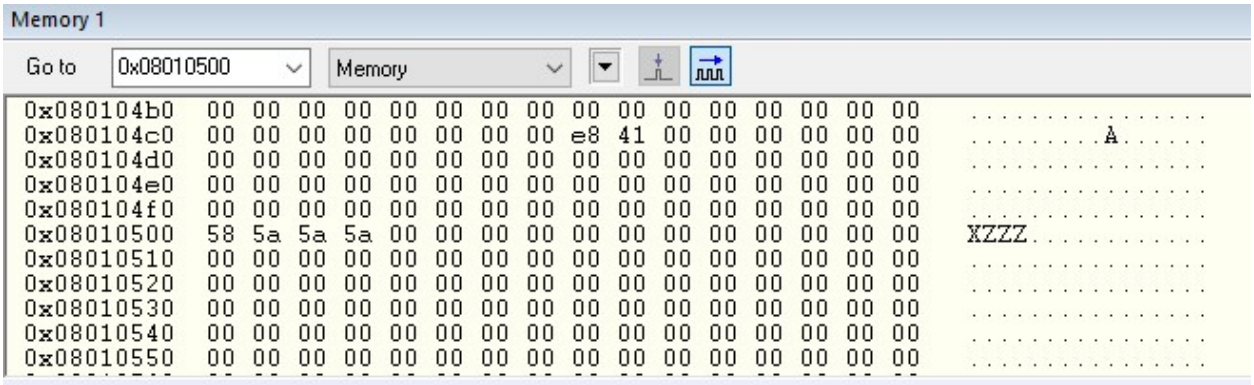


Output: No fault is reported and ISR is not triggered. The above-mentioned address has the value that is written.



Test case 2: Writing 0x5A5A5A5A into address of SRAM1 0x08010500 with one-bit parity error when ECC error injection is enabled.





Auto correct is enabled but memory is still showing error injected data.

Test case 3: Writing 0x5A5A5A5A into address of SRAM1 0x08010500 with Two-bit parity error when ECC error injection is enabled.(non-correctable)

startup_tviibe1m_cm4.asm | SRAM_ECC.c | Rte.c | Os_Callout_Stubs.c | Mcu.c | Mcu_PBcfg.c | InitStartUp.c | Os_Core.c | Spi.c

```
            faultData3 = Cy_SysFlt_GetData3(FAULT_STRUCT0);
        }

        if(fault_source == CY_SYSFLT_RAMC1_C_ECC) // System SRAM 1 correctable ECC error
        {
            faultData0 = Cy_SysFlt_GetData0(FAULT_STRUCT0);
            faultData1 = Cy_SysFlt_GetData1(FAULT_STRUCT0);
            faultData2 = Cy_SysFlt_GetData2(FAULT_STRUCT0);
            faultData3 = Cy_SysFlt_GetData3(FAULT_STRUCT0);
        }
    }

    Cy_SysFlt_ClearInterrupt(FAULT_STRUCT0);
    Cy_SysFlt_ClearMaskByIdx(FAULT_STRUCT0, CY_SYSFLT_NO_FAULT);
}

void EnableRamEcc(void)
{
            // VALIDATING FAULT SOURCE, RAM_1_C_IDX
        // SRAM 0 FAULT MASK AND SOURCE CONFIGURATION FOR ONE BIT STARTED
        // FAULT_STRUCT0->unMASK1.stcField.u32SOURCE = 0x04000000;
        {
            Cy_SysFlt_ClearStatus(FAULT_STRUCT0); // clear status (typically this process is
            Cy_SysFlt_SetMaskByIdx(FAULT_STRUCT0, CY_SYSFLT_RAMC1_C_ECC); // enable Fault Sy
            Cy_SysFlt_SetInterruptMask(FAULT_STRUCT0);
        }
            FAULT_STRUCT0->unMASK1.stcField.u32SOURCE = 0x30000000;
            CPUSS->unRAM1_CTL0.stcField.u1ECC_EN = 1;
```

Watch 1

Expression	Value	Location
u25WORD_...	Error (c...	
pf_source	0x5A5A5A1A	0x0801A
pRam	Error (c...	
<click to add>		

Registers 1

Find: FAU Group: FAULT

Name	Value	Access
STRUCT[0]_CTL	0x00000000	ReadWrit
STRUCT[0]_STATUS	0x8000003D	ReadWrit
VALID	1	ReadWrit
IDX	0x3D	ReadWrit
STRUCT[0]_DATA[0]	0x08010500	ReadWrit
DATA	0x08010500	ReadWrit
STRUCT[0]_DATA[1]	0x0000000C	ReadWrit
STRUCT[0]_DATA[2]	0x00000000	ReadWrit
STRUCT[0]_DATA[3]	0x00000000	ReadWrit
STRUCT[0]_PENDING0	0x00000001	ReadOnly
STRUCT[0]_PENDING1	0x00000000	ReadOnly
STRUCT[0]_PENDING2	0x00000000	ReadOnly
STRUCT[0]_MASK0	0x00000000	ReadWrit
STRUCT[0]_MASK1	0x30000000	ReadWrit
SOURCE	0x30000000	ReadWrit
STRUCT[0]_MASK2	0x00000000	ReadWrit
STRUCT[0]_INTR	0x00000001	ReadWrit
FAULT	1	ReadWrit
STRUCT[0]_INTR_SET	0x00000001	ReadWrit
STRUCT[0]_INTR_MASK	0x00000001	ReadWrit
STRUCT[0]_INTR_MASKED	0x00000001	ReadOnly
STRUCT[1]_CTL	0x00000000	ReadWrit
STRUCT[1]_STATUS	0x00000047	ReadWrit
STRUCT[1]_DATA[0]	0xD66819D8	ReadWrit
STRUCT[1]_DATA[1]	0xD91A76FC	ReadWrit

Memory 1

Go to 0x08010500 Memory

```
0x080104b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x080104c0  00 00 00 00 00 00 00 00 e8 41 00 00 00 00 00 00 00   .........A......
0x080104d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x080104e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x080104f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x08010500  1a 5a 5a 5a 00 00 00 00 00 00 00 00 00 00 00 00 00   .ZZZ............
0x08010510  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x08010520  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x08010530  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x08010540  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x08010550  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

Test case 4: Writing 0x5A5A5A5A into address of SRAM1 0x08010500 with one-bit parity error when ECC error injection is enabled and trying read the data from same memory twice with AUTO_CORRECT disabled.

```
void EnableRamEcc(void)
{
            // VALIDATING FAULT SOURCE, RAM_1_C_IDX
        // SRAM 0 FAULT MASK AND SOURCE CONFIGURATION FOR ONE BIT STARTED
        // FAULT_STRUCT0->unMASK1.stcField.u32SOURCE = 0x04000000;
        {
            Cy_SysFlt_ClearStatus(FAULT_STRUCT0); // clear status (typically this process is done by boot code)
            Cy_SysFlt_SetMaskByIdx(FAULT_STRUCT0, CY_SYSFLT_RAMC1_C_ECC); // enable Fault System SRAM 0 non-correctable ECC er
            Cy_SysFlt_SetInterruptMask(FAULT_STRUCT0);
        }
            FAULT_STRUCT0->unMASK1.stcField.u32SOURCE |= 0x30000000;
            CPUSS->unRAM1_CTL0.stcField.u1ECC_EN = 1;
            CPUSS->unRAM1_CTL0.stcField.u1ECC_AUTO_CORRECT = 0;
            CPUSS->unRAM1_CTL0.stcField.u1ECC_INJ_EN = 1;
            CPUSS->unECC_CTL.stcField.u25WORD_ADDR = (0x000500 >> 2);
            CPUSS->unECC_CTL.stcField.u7PARITY = RAM_1_ONEBIT_PARITY;
            CY_SET_REG32(RAM_1_ADDRESS, RAM_1_DATA);
            CY_GET_REG32(RAM_1_ADDRESS);
            //Cy_SysTick_DelayInUs(50);
            while(!isr_triggered);
            isr_triggered = 0;
            pf_source = CY_GET_REG32(RAM_1_ADDRESS);
            while(!isr_triggered);

}
```

RAM1_CTL	0x00050001
ECC_INJ_EN	1
ECC_AUTO_CORRECT	0
ECC_EN	1
FAST_WS	0x0
SLOW_WS	0x1
RAM1_STATUS	0x00000001
WB_EMPTY	1
RAM1_PWR_CTL	0xFA050003
RAM2_CTL	0x00000000
RAM2_STATUS	0x00000000
RAM2_PWR_CTL	0x00000000
RAM_PWR_DELAY_CTL	0x00000096
ROM_CTL	0x00000001
ECC_CTL	0xBA000140
PARITY	0x5D
WORD_ADDR	0x000140

During the second read operation, the fault is not triggered immediately, but getting updated to Pending Register 1.

Registers 1

Find: FAU Group: FAULT

Name	Value
STRUCT[0]_CTL	0x00000000
STRUCT[0]_STATUS	0x8000003C
VALID	1
IDX	0x3C
STRUCT[0]_DATA[0]	0x08010500
STRUCT[0]_DATA[1]	0x00000045
STRUCT[0]_DATA[2]	0x00000000
STRUCT[0]_DATA[3]	0x00000000
STRUCT[0]_PENDING0	0x00000001
SOURCE	0x00000001
STRUCT[0]_PENDING1	0x10000000
SOURCE	0x10000000
STRUCT[0]_PENDING2	0x00000000
SOURCE	--
STRUCT[0]_MASK0	0x00000000
STRUCT[0]_MASK1	0x30380000
SOURCE	--
STRUCT[0]_MASK2	0x00000000
STRUCT[0]_INTR	0x00000000
STRUCT[0]_INTR_SET	0x00000000
STRUCT[0]_INTR_MASK	0x00000001
STRUCT[0]_INTR_MASKED	0x00000000

Test Case 5: Writing 0x5A5A5A5A into address of SRAM1 0x08010500 with one-bit parity error when ECC error injection is enabled and trying read the data from same memory twice.

Note: During First read AUTO_CORRECT is enabled and second read AUTO_CORRECT is disabled.

Observations: During first time read fault is getting serviced and memory is not updated with correct data.

Os_Callout_Stubs.c | startup_tviibe1m_cm4.asm | FaultHandlerECC.c | Rte.c | SRAM_ECC.c | Mcu.c | Mcu_PBcfg.c | InitStartUp.c | Os_Core.c | StartApplication.c | Spi.c | Det.c

```
        }
    }

    Cy_SysFlt_ClearInterrupt(FAULT_STRUCT0);
    //Cy_SysFlt_ClearMaskByIdx(FAULT_STRUCT0, CY_SYSFLT_NO_FAULT);
}

void EnableRamEcc(void)
{
                // VALIDATING FAULT SOURCE, RAM_1_C_IDX
            // SRAM 0 FAULT MASK AND SOURCE CONFIGURATION FOR ONE BIT STARTED
            // FAULT_STRUCT0->unMASK1.stcField.u32SOURCE = 0x04000000;
            {
                Cy_SysFlt_ClearStatus(FAULT_STRUCT0); // clear status (typically this process is done by boot code)
                Cy_SysFlt_SetMaskByIdx(FAULT_STRUCT0, CY_SYSFLT_RAMC1_C_ECC); // enable Fault System SRAM 0 non-correctable ECC er
                Cy_SysFlt_SetInterruptMask(FAULT_STRUCT0);
            }
                FAULT_STRUCT0->unMASK1.stcField.u32SOURCE |= 0x30000000;
                CPUSS->unRAM1_CTL0.stcField.u1ECC_EN = 1;
                CPUSS->unRAM1_CTL0.stcField.u1ECC_AUTO_CORRECT = 0;
                CPUSS->unRAM1_CTL0.stcField.u1ECC_INJ_EN = 1;
                CPUSS->unECC_CTL.stcField.u25WORD_ADDR = (0x000500 >> 2);
                CPUSS->unECC_CTL.stcField.u7PARITY = RAM_1_ONEBIT_PARITY;
                CY_SET_REG32(RAM_1_ADDRESS, RAM_1_DATA);
                CY_GET_REG32(RAM_1_ADDRESS);
                //Cy_SysTick_DelayInUs(50);
                while(!isr_triggered);
                isr_triggered = 0;
                pf_source = CY_GET_REG32(RAM_1_ADDRESS);
                while(!isr_triggered);

}
```

Watch 1

Expression	Value	Loca
u25WORD_...	Error (c...	
pf_source	0x00000000	0x08
pRam	Error (c...	
<click to add>		

Registers 1

Find: FAU Group: CPUSS

Name	Value
RAM0_PWR_MACRO_CTL[3]	0x00000000
RAM0_PWR_MACRO_CTL[4]	0x00000000
RAM0_PWR_MACRO_CTL[5]	0x00000000
RAM0_PWR_MACRO_CTL[6]	0x00000000
RAM0_PWR_MACRO_CTL[7]	0x00000000
RAM0_PWR_MACRO_CTL[8]	0x00000000
RAM0_PWR_MACRO_CTL[9]	0x00000000
RAM0_PWR_MACRO_CTL[10]	0x00000000
RAM0_PWR_MACRO_CTL[11]	0x00000000
RAM0_PWR_MACRO_CTL[12]	0x00000000
RAM0_PWR_MACRO_CTL[13]	0x00000000
RAM0_PWR_MACRO_CTL[14]	0x00000000
RAM0_PWR_MACRO_CTL[15]	0x00000000
RAM1_CTL	0x00070001
ECC_INJ_EN	1
ECC_AUTO_CORRECT	1
ECC_EN	1
FAST_WS	0x0
SLOW_WS	0x1
RAM1_STATUS	0x00000001
RAM1_PWR_CTL	0xFA050003
RAM2_CTL	0x00000000
RAM2_STATUS	0x00000000
RAM2_PWR_CTL	0x00000000
RAM_PWR_DELAY_CTL	0x00000096
ROM_CTL	0x00000001
ECC_CTL	0xBA000140
DP_STATUS	0x00320101
BUFF_CTL	0x00000001
DDFT_CTL	0x00000000

Memory 1

Go to 0x08010500 Memory

```
0x080104b0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x080104c0  00 00 00 00 00 00 00 00 e8 41 00 00 00 00 00 00  .........A......
0x080104d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x080104e0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x080104f0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x08010500  58 5a 5a 5a 00 00 00 00 00 00 00 00 00 00 00 00  XZZZ............
0x08010510  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x08010520  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x08010530  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x08010540  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x08010550  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

Observations: During second time read, fault is neither serviced nor reported to fault structure as a pending source.