

フォールト発生の原因を調査したい (1/3)

CPUとして、想定外の状態になるとフォールトが発生する

種類により3つのローカルフォールトが発生
明示的に有効にしないと、ハードフォールトに昇格する

Memory Management Fault

・・・MPUの設定への違反

Bus Fault

・・・アドレスやメモリ関連のフォールト

Usage Fault

・・・不正な命令指定などの違反



Hard Fault

ローカルフォールトの一覧については下記を参照

<http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0337gj/ch05s12s02.html>

フォールト発生の原因を調査したい (2/3)

フォールト発生時の情報はSystem Control レジスタに表示される

| レジスタ | |
|--------------------------|-----------------------|
| System Control Block | |
| <find reg> | |
| AFSR = 0x00000000 | ID_ISAR0 = 0x01101110 |
| AIACPR = 0x5A050000 | ID_ISAR1 = 0x02112000 |
| BFAR = 0x00000000 | ID_ISAR2 = 0x20232231 |
| BCCR = 0x00000020 | ID_ISAR3 = 0x01111131 |
| CFSR = 0x00000000 | ID_ISAR4 = 0x01310132 |
| CFAR = 0x00000000 | ID_ISAR5 = 0x00000000 |
| DFSR = 0x00000008 | MMFAR = 0x00000000 |
| HFSR = 0x00000000 | SCR = 0x00000000 |
| ICSR = 0x00000000 | VTOR = 0x08000000 |
| CPUID = 0x410FC271 | SHPR1 = 0x00000000 |
| ID_PFR0 = 0x00000030 | SHPR2 = 0x00000000 |
| ID_PFR1 = 0x00000200 | SHPR3 = 0x00000000 |
| ID_DFR0 = 0x00100000 | SHCSR = 0x00000000 |
| ID_AFR0 = 0x00000000 | |
| ID_MMFR0 = 0x00110030 | |
| ID_MMFR1 = 0x00000000 | |
| ID_MMFR2 = 0x01000000 | |
| ID_MMFR3 = 0x00000000 | |

```
例：存在しないアドレスにReadアクセス
int data=*(int*)(0x30000000);
```

CFSR: Configurable Fault Status Register

| |
|-------------------|
| CFSR = 0x00008200 |
| IACCVIOL = 0 |
| DACCVIOL = 0 |
| MUNSTKERR = 0 |
| MSTKERR = 0 |
| MLSPERR = 0 |
| MMARVALID = 0 |
| IBUSERR = 0 |
| PRECISERR = 1 |
| IMPRECISERR = 0 |
| UNSTKERR = 0 |
| STKERR = 0 |
| LSPERR = 0 |
| BFARVALID = 1 |
| UNDEFINSTR = 0 |
| INVSTATE = 0 |
| INVPC = 0 |
| NOCP = 0 |
| UNALIGNED = 0 |
| DIVBYZERO = 0 |

何がエラーになっているかを示す
1が立っていたら要確認

PRECISERR : データバスエラー

BFARVALID : BFARが有効

BFAR: Bus Fault Address Register

| |
|----------------------|
| BFAR = 0x30000000 |
| ADDRESS = 0x30000000 |

0x30000000へのアクセスでエラーを示す

レジスタ値を見ることでエラー原因の推測ができる

フォールト発生の原因を調査したい (3/3)

フォールトに限らず、例外発生時はスタックにレジスタが退避

```

main.c
/* Add your application code here
*/
SysTick->LOAD = 0x0000FFFF;
SysTick->VAL = 0;
SysTick->CTRL = SysTick_CTRL_CLKSOURCE_Msk |
                SysTick_CTRL_TICKINT_Msk |
                SysTick_CTRL_ENABLE_Msk;

int data=*(int*)(0x30000000);

printf("Hello!World");
/* Infinite loop */
while (1)
{
    for(volatile int i = 0; i < 10000;i++);
    main_count++;
}

/**
 * @brief System Clock Configuration
 * The system Clock is configured as follow
 * System Clock source = PLL (
 * SYSCLK (Hz) = 84000
 * HCLK (Hz) = 84000
 */

```

| レジスタ | 値 |
|---------|------------|
| R0 | 0x30000000 |
| R1 | 0xE000E010 |
| R2 | 0xE000ED18 |
| R3 | 0x00000001 |
| R4 | 0x00000000 |
| R5 | 0x00000000 |
| R6 | 0x00000000 |
| R7 | 0x00000000 |
| R8 | 0x00000000 |
| R9 | 0x00000000 |
| R10 | 0x00000000 |
| R11 | 0x00000000 |
| R12 | 0x00000000 |
| SP | 0x20000398 |
| LR | 0x080014C5 |
| PSR | 0x01000000 |
| APSR | 0x00000000 |
| IPSR | 0x00000000 |
| EPSR | 0x01000000 |
| PC | 0x08001512 |
| SP_main | 0x20000398 |

| 位置 | データ | 変換 |
|-----|------------|------------|
| R0 | 0x20000330 | 0x30000000 |
| R1 | 0x20000334 | 0xE000E010 |
| R2 | 0x20000338 | 0xE000ED18 |
| R3 | 0x2000033C | 0x00000001 |
| R12 | 0x20000340 | 0x00000000 |
| LR | 0x20000344 | 0x080014C5 |
| PC | 0x20000348 | 0x08001512 |
| | 0x2000034C | 0x01000000 |
| | 0x20000350 | 0x00000002 |
| | 0x20000354 | 0x00000000 |

バスフォールト発生

```

/**
 * @brief This function handles Hard Fault exception
 * @param None
 * @retval None
 */
void HardFault_Handler(void)
{
    /* Go to infinite loop when Hard Fault exception occurs
    while (1)
    {
    }
}

```

| | |
|------------|------------|
| R11 | 0x00000000 |
| R12 | 0x00000000 |
| SP | 0x20000330 |
| LR | 0xFFFFFFFF |
| PSR | 0x01000003 |
| APSR | 0x00000000 |
| IPSR | 0x00000003 |
| EPSR | 0x01000000 |
| PC | 0x08001E16 |
| SP_main | 0x20000330 |
| SP_process | 0x00000000 |

PC : どこを実行していたのか
LR : どこにReturnする予定だったか
判断可能