Dialog SDK 5.0.x/6.0.x Tutorial

Debugging

2017 March

...personal
...portable
...connected

BLE device advertising process

Let's build a demo together …

- **Before we start, we recommend you to …**
  - Install the latest Smartsnippets studio from Dialog customer support website
  - Download the SDK as well
  - Link:
    - https://support.dialog-semiconductor.com/connectivity

- **Consideration …**
  - All the changes are applicable in both the SDK 5.0.x (DA14580/1/2/3) and SDK 6.0.x (DA14585/6) if it is not mentioned specifically for a particular application

- **What are you going to learn from this tutorial …**
  - Debugging using serial port
  - Possible ways of analysing hard-fault

Contents

Debugging using serial port

Possible way of analyzing hard-fault

Serial debug Contents

Barebone example

- **This example demonstrates:**
  - How to activate and use serial debug in a DA1458x project step by step

- **Software you need:**
  - Dialog Smartsnippets studio
  - Dialog SDK
  - Project location:
    - ..\projects\target_apps\ble_examples\ble_app_profile

Serial debug

Configuration

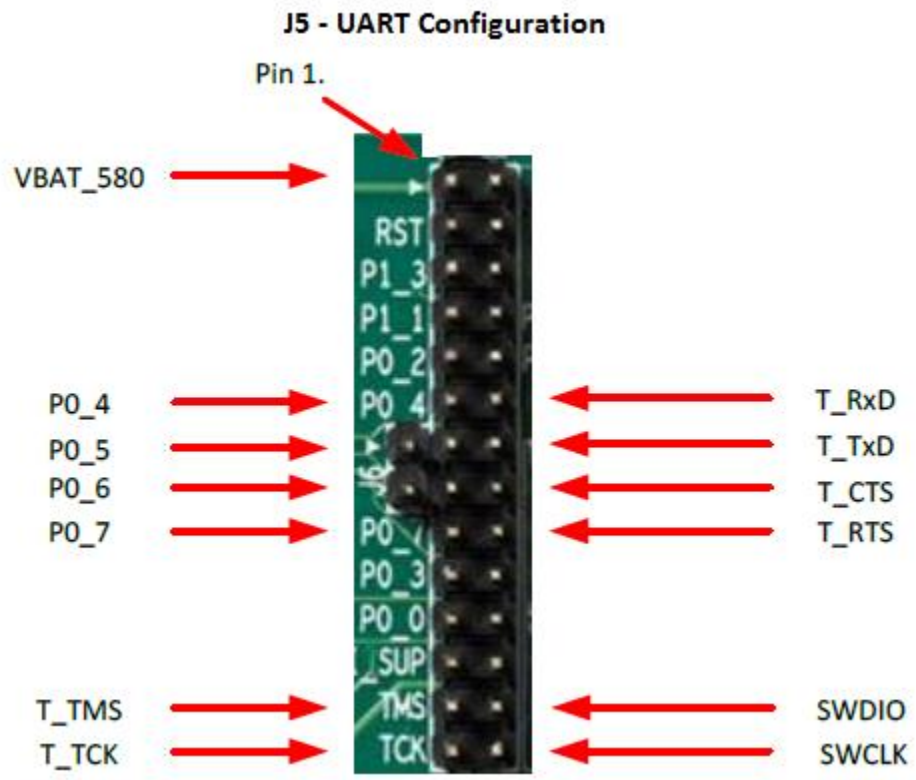
TODO 1 - Define the serial UART print flag, initially it is undefined.

```
/* @file da1458x_config_basic.h */
/* copy and paste */
#define CFG_PRINTF
```

TODO 2 – Check the port configuration

```
/* @file user_periph_setup.h */
```

```
/****************************************************************************************/
/* UART2 pin configuration (debug print console)                                        */
/****************************************************************************************/
#ifdef CFG_PRINTF_UART2
…
   #elif HW_CONFIG_PRO_DK
      #define UART2_TX_GPIO_PORT  GPIO_PORT_0
      #define UART2_TX_GPIO_PIN   GPIO_PIN_4
      #define UART2_RX_GPIO_PORT  GPIO_PORT_0
      #define UART2_RX_GPIO_PIN   GPIO_PIN_5
…
   #endif
#endif
```



Serial debug

Configuration

TODO 3 - Find user_barbone.c under user_app project folder.

```
/* @file user_barebone.c */
/* copy and paste */
#include "arch_console.h"
```

TODO 4 – Check the port configuration

```
/* @file user_barebone.c */
```

```
void user_app_adv_start(void)
{
    struct gapm_start_advertise_cmd* cmd;
    // Schedule the next advertising data update
    app_adv_data_update_timer_used = app_easy_timer(APP_ADV_DATA_UPDATE_TO, adv_data_update_timer_cb);
   // PRINT
    arch_printf("\r\n ADVERTISING TEST STARTED *\r\n");
    cmd = app_easy_gap_undirected_advertise_get_active();
    // add manufacturer specific data dynamically
    mnf_data_update();
    app_add_ad_struct(cmd, &mnf_data, sizeof(struct mnf_specific_data_ad_structure));
    app_easy_gap_undirected_advertise_start();
}
```

Serial debug

Try this

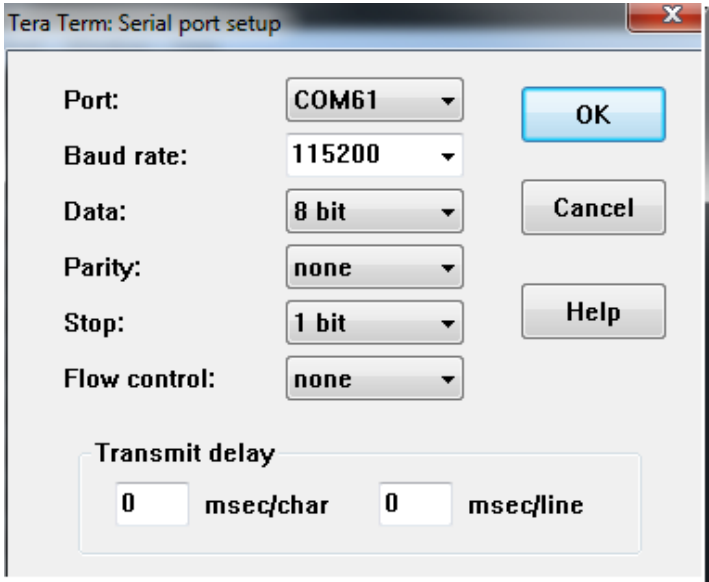
TODO 5 - Try to find out how to send a complete advertising serial message.

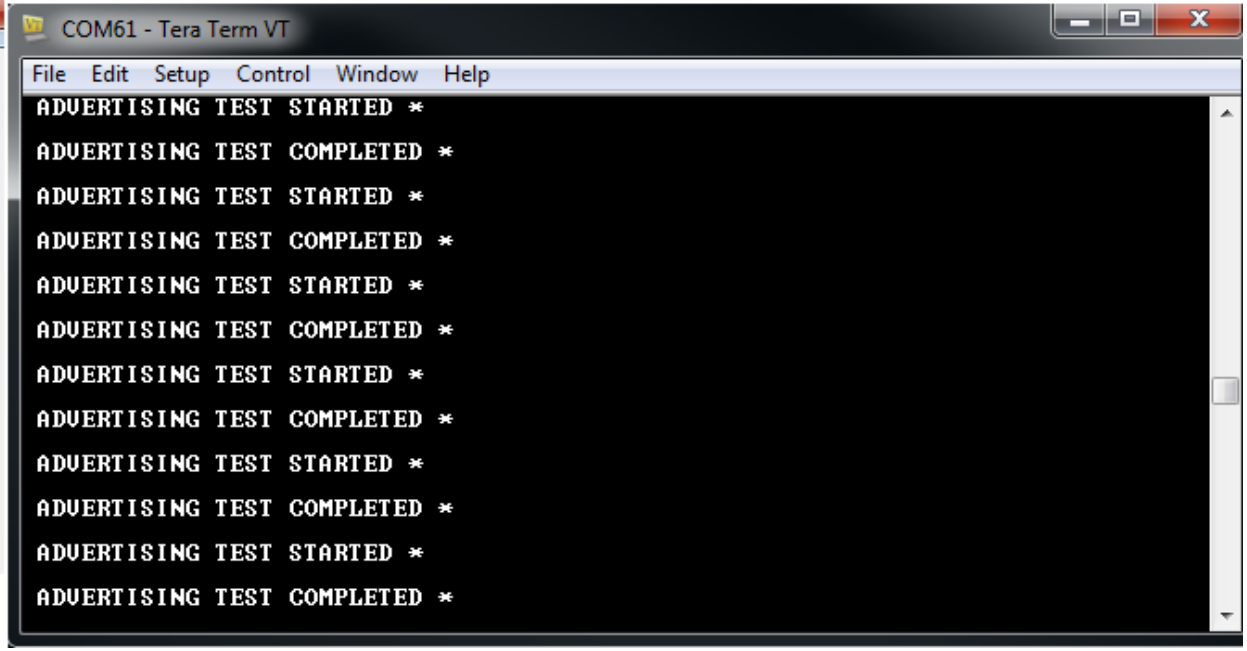
We have used Teraterm as a terminal software and you can see the configuration in the next slide page.

```
/* @file user_barebone.c */
/* copy and paste */
#include "arch_console.h"
```

Serial debug

Output





Contents

Possible way of analyzing hard-fault

Hard-fault analysis Contents

Peripheral custom profile example

- **This example demonstrates:**
  - How to track hard-fault in DA1458x project step by step

- **Software you need:**
  - Dialog Smartsnippets studio
  - Dialog SDK
  - Project location:
    - ..\projects\target_apps\ble_examples\ble_app_peripheral

Hard-fault debug

Configuration

TODO 1 - Insert a jumper in J9 of DA14580 DEV-KIT Pro.

TODO 2 – Open ..\target_apps\ble_examples\ble_app_peripheral\Keil_5\ble_app_peripheral.uvprojx.

TODO 3 – Search for LED write indication handler - **user_custs1_led_wr_ind_handler**

TODO 4 – Replace with the following code in - **user_custs1_led_wr_ind_handler**

```
void user_custs1_led_wr_ind_handler(ke_msg_id_t const msgid,
                                    struct custs1_val_write_ind const *param,
                                    ke_task_id_t const dest_id,
                                    ke_task_id_t const src_id)
{
    uint8_t val = 0;
    memcpy(&val, &param->value[0], param->length);

    if (val == CUSTS1_LED_ON)
        GPIO_SetActive(GPIO_LED_PORT, GPIO_LED_PIN);
    else if (val == CUSTS1_LED_OFF)
        GPIO_SetInactive(GPIO_LED_PORT, GPIO_LED_PIN);

    // test
    *(uint32_t *)0x90 = 0x90;
}
```

Hard-fault debug

Configuration

TODO 5 - You can switch off the sleep mode by assigning app_default_sleep_mode = ARCH_SLEEP_OFF; in user_config.h file and change default BD address and device name (these are optional).

TODO 6 – Compile and download the code in the device (device starts advertising).

TODO 7 – Open iOS Light-Blue - you will find your device is advertising.

TODO 8 – Connect to your device using Light-Blue.

TODO 9 - Find - LED State characteristics and press on it.

TODO 10 – Find - write new value - and press on it.

TODO 11 – Write - 1 and press – Done.

TODO 12 – You will see the Light-Blue shows disconnection alert message.

TODO 13 – Go to Keil IDE you will see a hard-fault is trapped.

Hard-fault debug

Configuration

TODO 13 – Go to Keil IDE you will see a hard-fault is trapped.

```
 98
 99  void HardFault_HandlerC(unsigned long *hardfault_args)
100 {
101
102     if (DEVELOPMENT_DEBUG)
103     {
104         SetBits16(SYS_CTRL_REG, DEBUGGER_ENABLE, 1);    // enable
105         *(volatile unsigned long *)(STATUS_BASE       ) = hardfau
106         *(volatile unsigned long *)(STATUS_BASE + 0x04) = hardfau
107         *(volatile unsigned long *)(STATUS_BASE + 0x08) = hardfau
108         *(volatile unsigned long *)(STATUS_BASE + 0x0C) = hardfau
109         *(volatile unsigned long *)(STATUS_BASE + 0x10) = hardfau
110         *(volatile unsigned long *)(STATUS_BASE + 0x14) = hardfau
111         *(volatile unsigned long *)(STATUS_BASE + 0x18) = hardfau
112         *(volatile unsigned long *)(STATUS_BASE + 0x1C) = hardfau
113         *(volatile unsigned long *)(STATUS_BASE + 0x20) = (unsign
114
115         *(volatile unsigned long *)(STATUS_BASE + 0x24) = (*((vol
116         *(volatile unsigned long *)(STATUS_BASE + 0x28) = (*((vol
117         *(volatile unsigned long *)(STATUS_BASE + 0x2C) = (*((vol
118         *(volatile unsigned long *)(STATUS_BASE + 0x30) = (*((vol
119         *(volatile unsigned long *)(STATUS_BASE + 0x34) = (*((vol
120         *(volatile unsigned long *)(STATUS_BASE + 0x38) = (*((vol
121         if (USE_WDOG)
122             wdg_freeze();            // Stop WDOG
123
124         if ((GetWord16(SYS_STAT_REG) & DBG_IS_UP) == DBG_IS_UP)
125             __asm("BKPT #0\n");
126         else
127             while(1);
128     }
129     else // DEVELOPMENT_DEBUG
```

1. Hardfault Trapped

Hard-fault debug

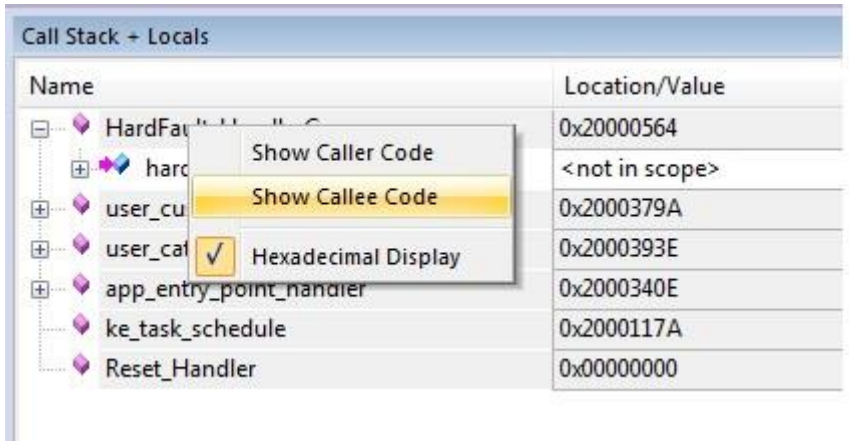
Configuration 1

TODO 14 – Now there are 2 options to debug hard-fault.

Option 1 is the easiest one:

TODO 15 – Open KEIL IDE call stack from - [View->Call stack window]

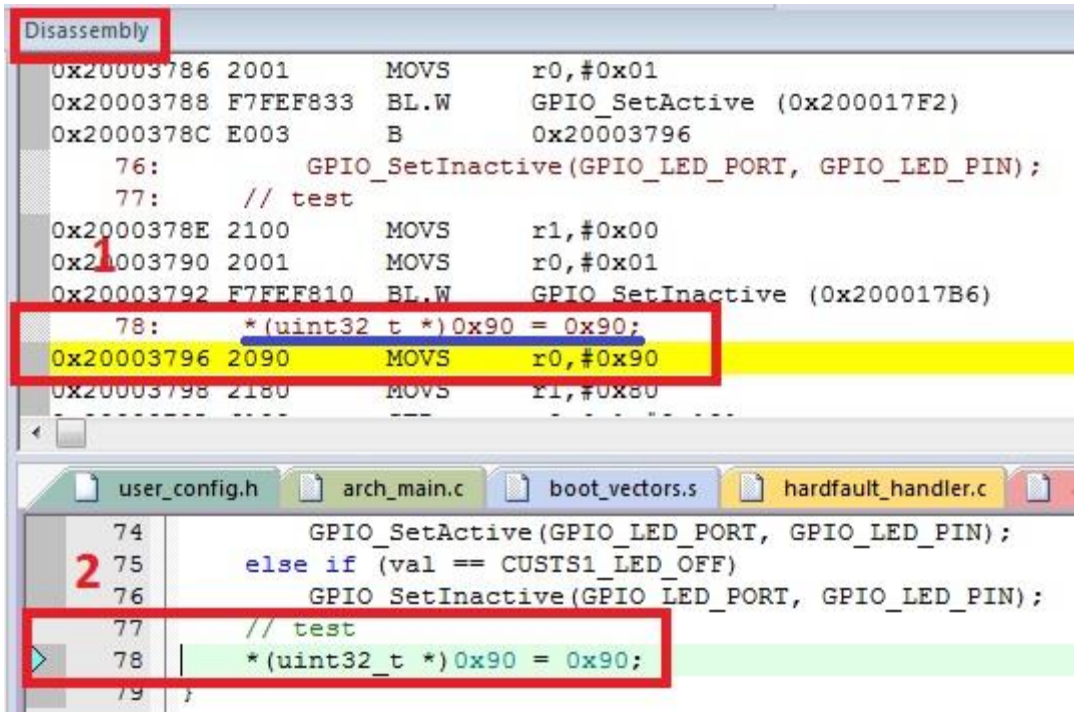
TODO 16 – Press right button mouse on [HardFault_HandlerC] and press [Show Callee Code] option



Hard-fault debug

Configuration

TODO 17 – You will see in disassembly window that where exactly the hardfault has occurred. You can see the hard-fault in code window too.



Hard-fault debug

Configuration 2

Option 2 when there is no call stack generated:

TODO 15 – Open KEIL IDE call stack from - [View->Register window]

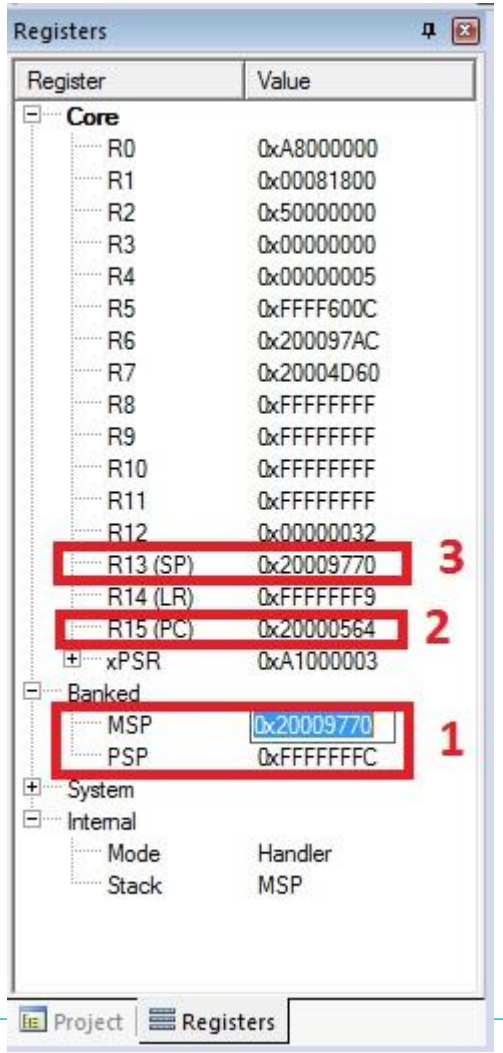
TODO 16 – Open KEIL IDE call stack from - [View->Memory window->Memory 1]

TODO 17 – In Registers window find out

Main stack pointer (MSP) for this study the address is **0x20009770**

Stack pointer (SP) for this study the address is **0x20009770**

Program counter (PC) for this study the address is **0x20000564**

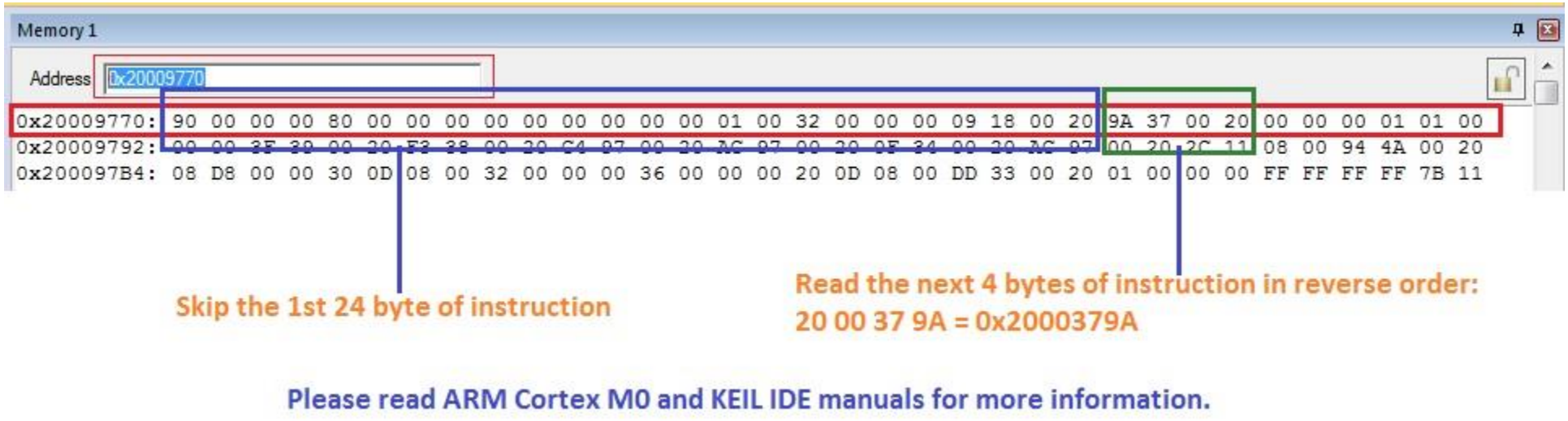


Hard-fault debug

Configuration 2

TODO 18 – Copy Main stack pointer (MSP) in this study the address is **0x20009770**
Paste in **Address of Memory 1 window.** Then create the memory address as instructed below:

In this study it is **0x2000379A**

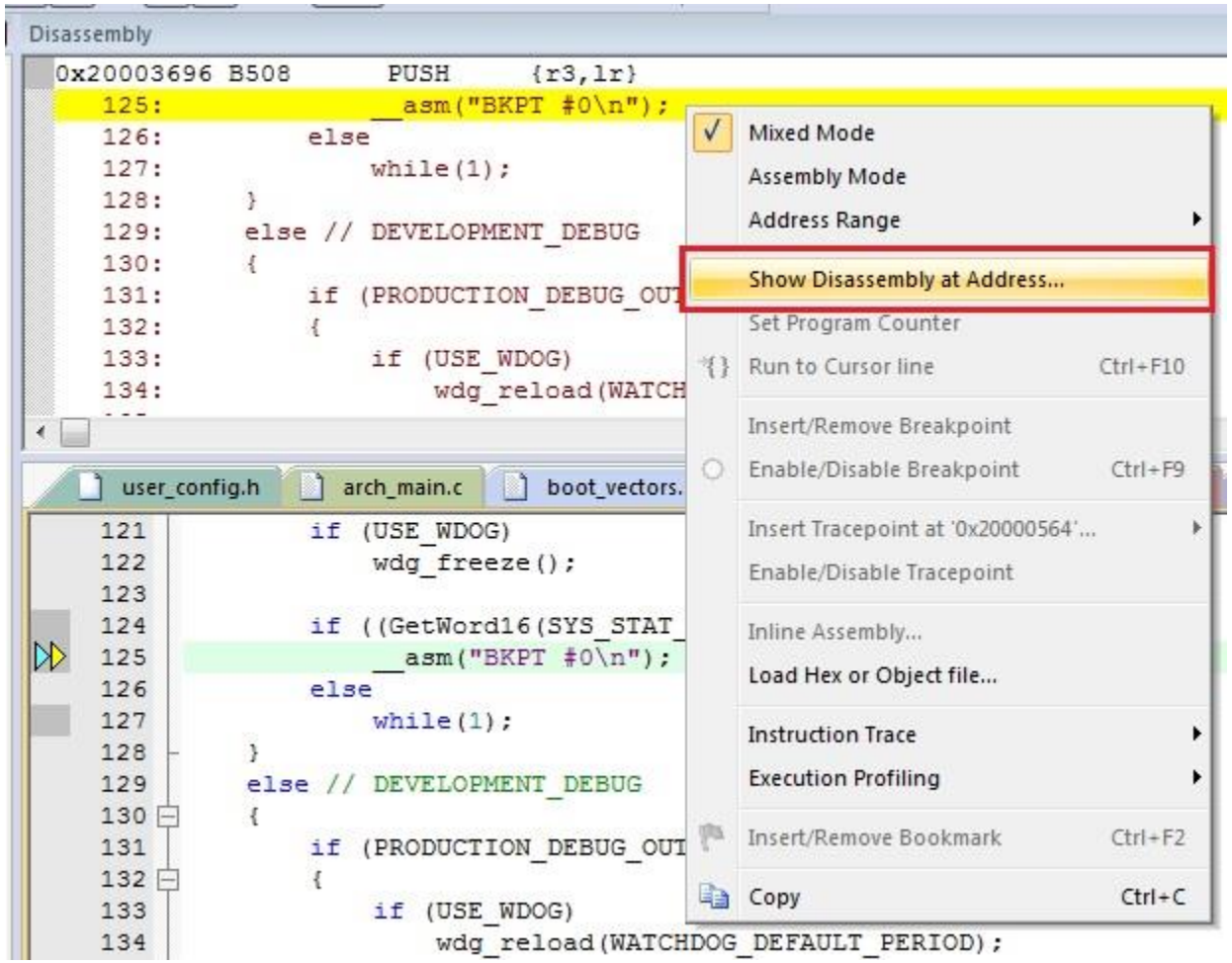


Please read ARM Cortex M0 and KEIL IDE manuals for more information.

Hard-fault debug

Configuration 2

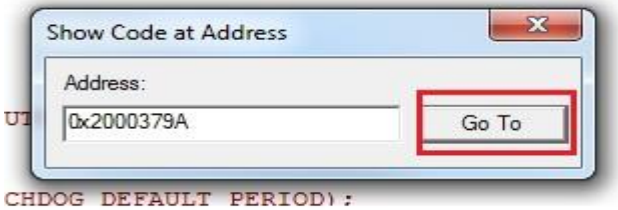
TODO 19 – Right click on - [Disassembly window] and Select [Show Disassembly at Address...]



Hard-fault debug

Configuration 2

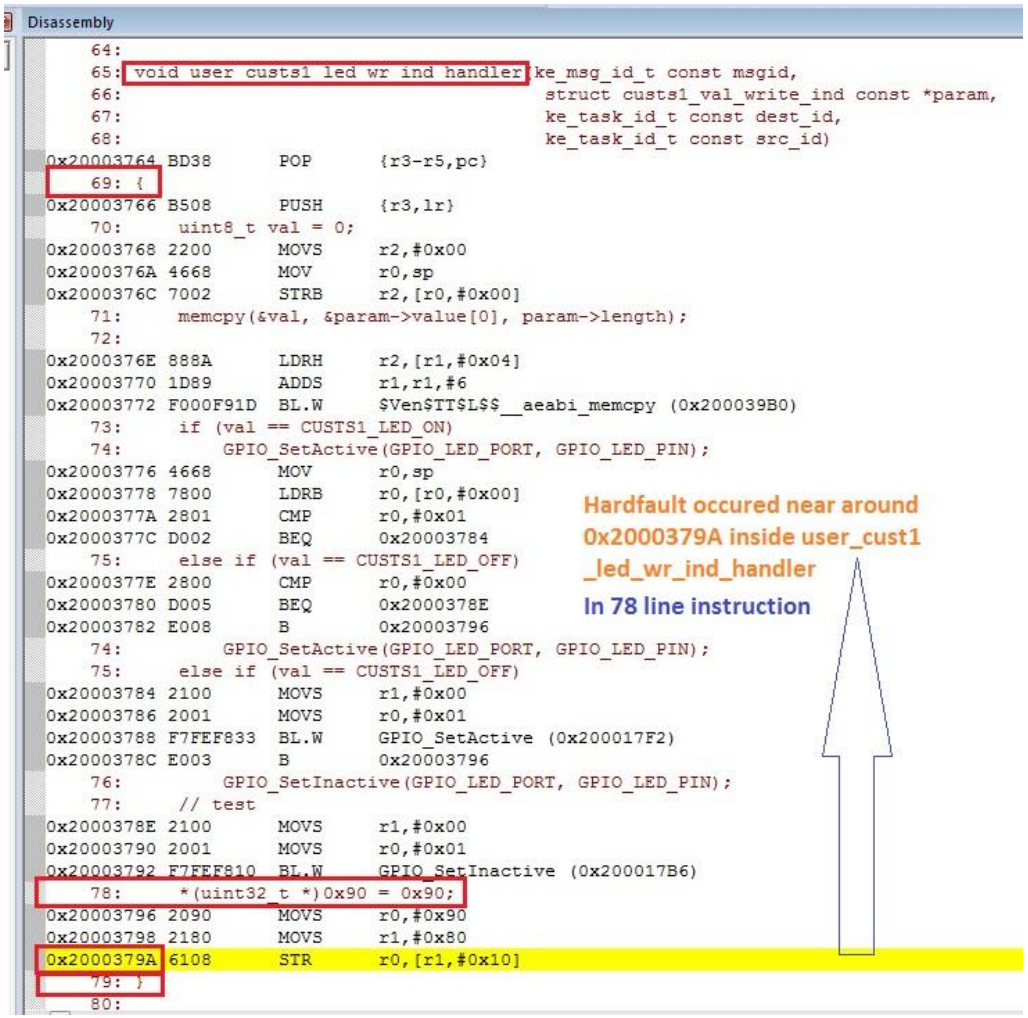
TODO 20 – Paste the address **0x2000379A** in [Show code at address window] and press [Go To] button.



Hard-fault debug

Configuration 2

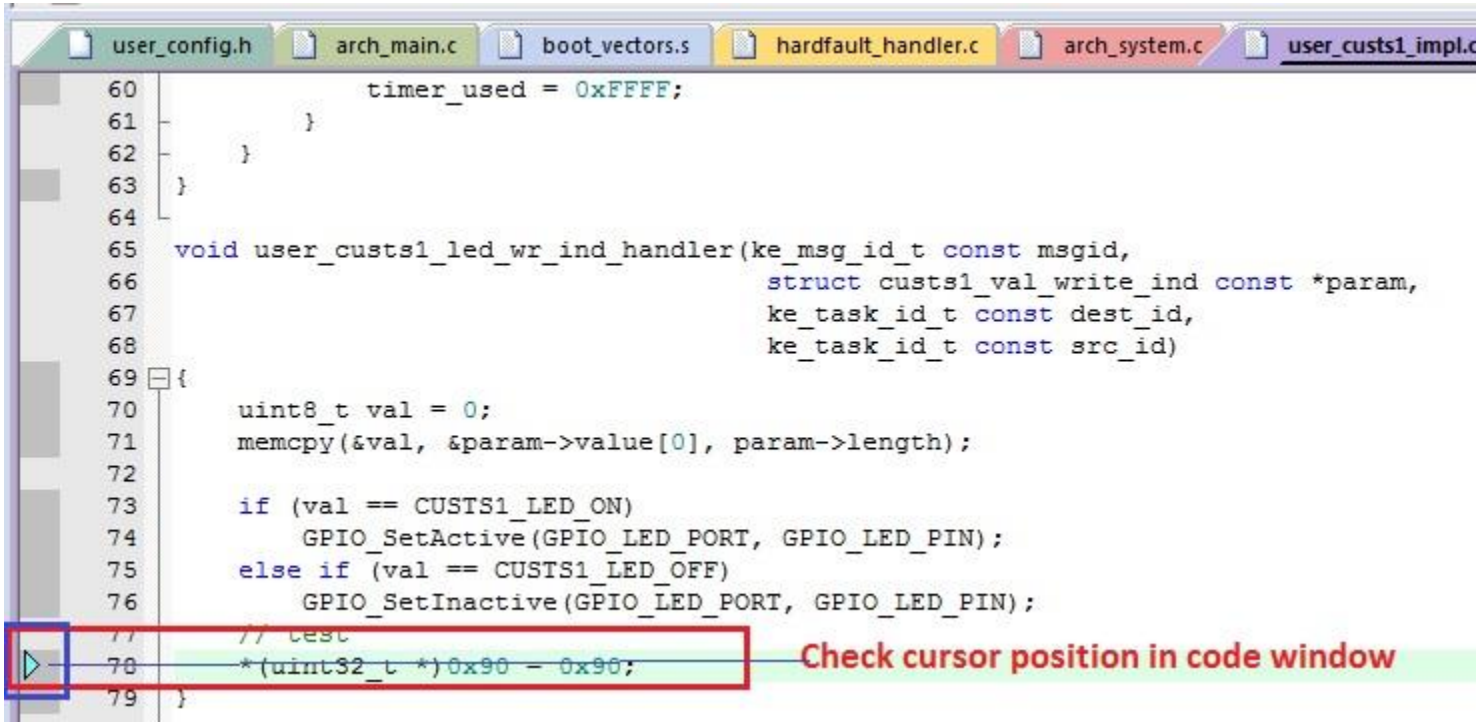
TODO 21 – Check the disassembly window.



Hard-fault debug

Configuration 2

TODO 22 – Check the cursor position in [code window]



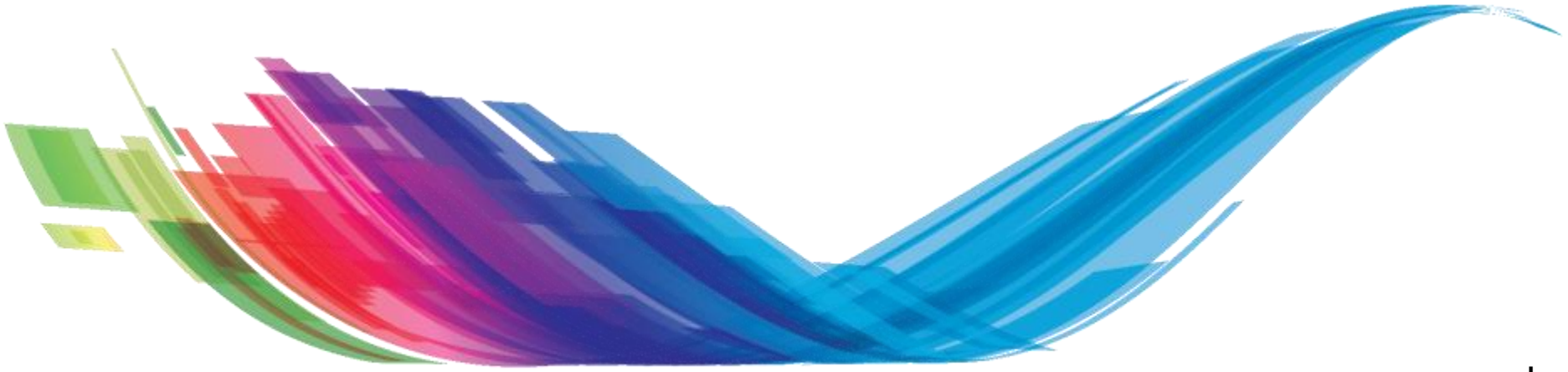
TODO 23 – PRACTISE, PRACTISE, PRACTISE

What's next

For more …

- **What's next …**
  - Please follow the other Dialog tutorials based on –
    - **SDK 5.0.x** for **DA14580/1/2/3** development OR
    - **SDK 6.0.x** for **DA14585/6** development
  - See **Reference** section of this training slide
  - Learn about Dialog BLE chip **differences** at a glance from –
    https://support.dialog-semiconductor.com/connectivity/products

The Power To Be…



…personal
…portable
…connected