Read PC Value by SVC

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This code reads the current PC, stack, R0, R1 and R12 registers value by SVC.
BSP Version	NUC230_240 Series BSP CMSIS V3.01.001
Hardware	NuTiny-EVB-NUC240 V1.2

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

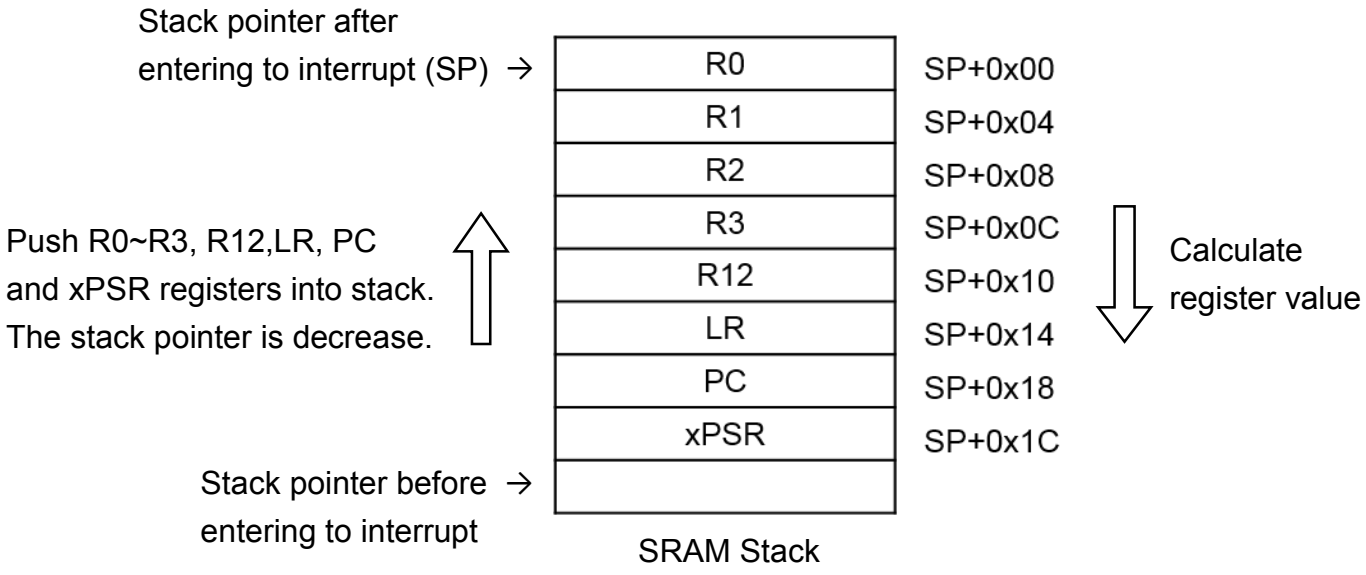
1 Function Description

1.1 Introduction

SVC interrupt is generated through executing SVC instruction which can provide service in OS operation system. After SVC interrupt is generated, system will push current registers status into stack. This code uses this information to get the PC value where the SVC instruction is executed to generate SVC interrupt, and then print the PC value. Others information can be obtain including stack, R0~R3, R12 and xPSR.

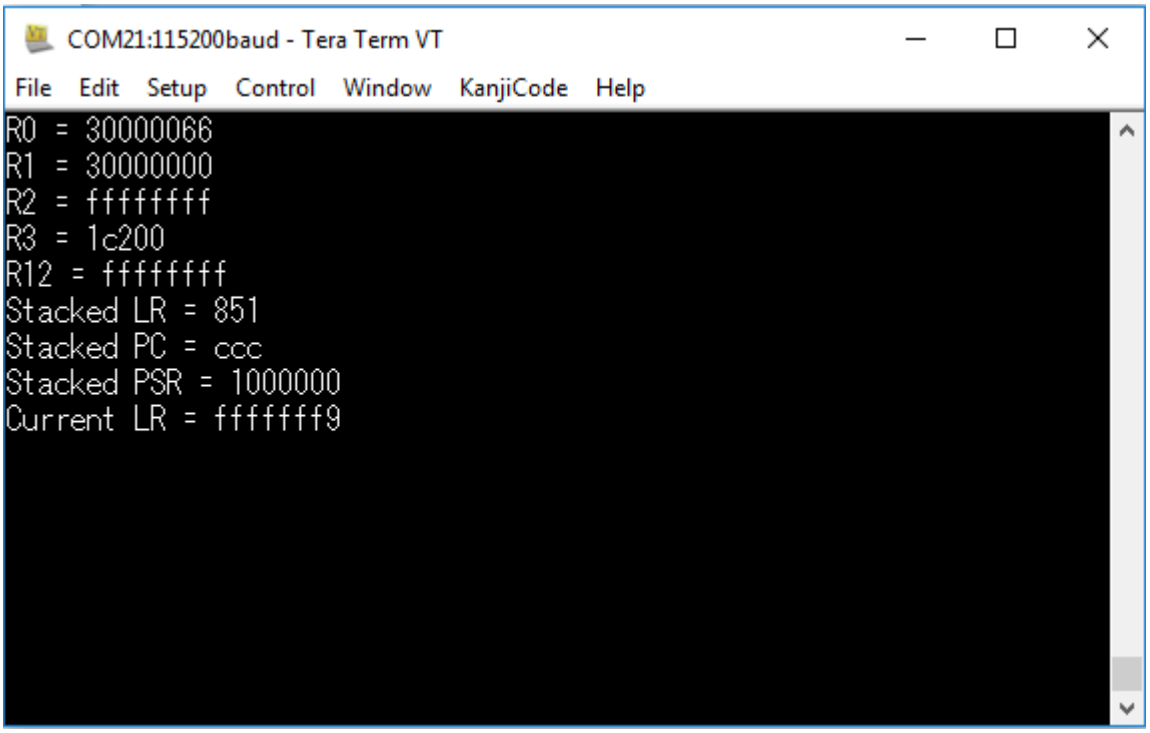
1.2 Principle

When interrupt is generated, system will push the following 8 registers value into stack: R0, R1, R2, R3, R12, LR, PC and xPSR. After entering to SVC interrupt, reading current stack pointer address can get these 8 registers value, too. Including the program address where execute the SVC instruction (PC).



1.3 Demo Result

Connect NUC240 development board with NuBridge through PB.0(UART RX) and PB.1(UART TX) to print message from UART0. The print message shows the PC value along with others information when calling SVC.



2 Code Description

SVC Interrupt Generation

Execute SVC instruction to generate SVC interrupt. Use the keyword __svc provided by Keil MDK to define the SVC function. Define a SVC function with the number is 0 and it does not have parameter and return value.

```
    /* Define SVC function */
    void __svc(0x00) svc_PC(void);
```

Call the defined SVC function directly to generate SVC interrupt.

```
    /* Supervisor Call */
    svc_PC();
```

SVC Interrupt Handler

Read the second bit value of LR register to check which stack is used, PSP or MSP. Then call the print message function with parameter: stack pointer address and return address.

```
__asm void SVC_Handler(void)
{
    MOVS    r0, #4
    MOV     r1, LR
    TST     r0, r1
    BEQ     stacking_used_MSP
    MRS     R0, PSP         ; first parameter - stacking was using PSP
    B       get_LR_and_branch
stacking_used_MSP
    MRS     R0, MSP         ; first parameter - stacking was using MSP
get_LR_and_branch
    MOV     R1, LR          ; second parameter is LR current value
    LDR     R2,=__cpp(dump_handler_c)
    BX      R2
    NOP
    BX      LR
}
```

Print the PC value

Get the register value: R0~R3, R12, LR, PC and xPSR from stack and print their value.

```
void dump_handler_c(unsigned int * hardfault_args, unsigned lr_value)
{
    static volatile unsigned int stacked_r0;
    static volatile unsigned int stacked_r1;
    static volatile unsigned int stacked_r2;
    static volatile unsigned int stacked_r3;
    static volatile unsigned int stacked_r12;
    static volatile unsigned int stacked_lr;
    static volatile unsigned int stacked_pc;
    static volatile unsigned int stacked_psr;

    stacked_r0 = ((unsigned long) hardfault_args[0]);
    stacked_r1 = ((unsigned long) hardfault_args[1]);
    stacked_r2 = ((unsigned long) hardfault_args[2]);
    stacked_r3 = ((unsigned long) hardfault_args[3]);
    stacked_r12 = ((unsigned long) hardfault_args[4]);
    stacked_lr = ((unsigned long) hardfault_args[5]);
    stacked_pc = ((unsigned long) hardfault_args[6]);
    stacked_psr = ((unsigned long) hardfault_args[7]);

    printf ("R0 = %x\n", stacked_r0);
    printf ("R1 = %x\n", stacked_r1);
    printf ("R2 = %x\n", stacked_r2);
    printf ("R3 = %x\n", stacked_r3);
    printf ("R12 = %x\n", stacked_r12);
    printf ("Stacked LR = %x\n", stacked_lr);
    printf ("Stacked PC = %x\n", stacked_pc);
    printf ("Stacked PSR = %x\n", stacked_psr);
    printf ("Current LR = %x\n", lr_value);
}
```

3 Software and Hardware Environment

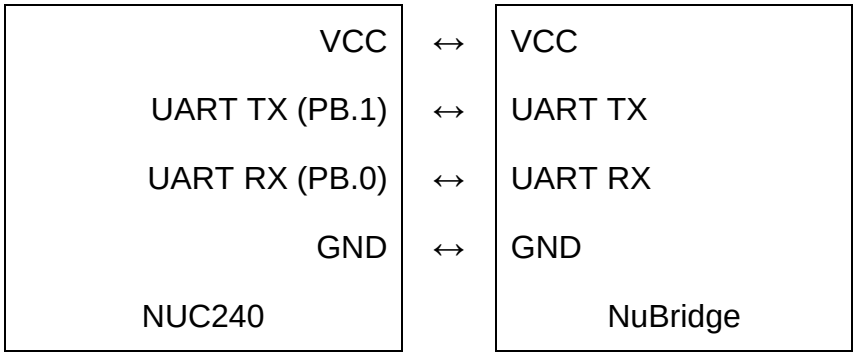
Software Environment

- BSP version
  - NUC230_240 Series BSP CMSIS V3.01.001
- IDE version
  - Keil uVersion 5.28

Hardware Environment

- Circuit components
  - NuTiny-EVB-NUC240 V1.2
  - NuBridge
- Diagram

Connect NUC240 development board with NuBridge to print message.



4 Directory Information

- EC_NUC240_READ_PC_BY_SVC_V1.00
  - Library — Sample code header and source files
    - CMSIS — Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
    - Device — CMSIS compliant device header file
    - StdDriver — All peripheral driver header and source files
  - SampleCode
    - ExampleCode — Source file of example code

5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click NUC240_READ_PC_BY_SVC.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Jul. 31, 2019	1.00	1. Initially issued.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.