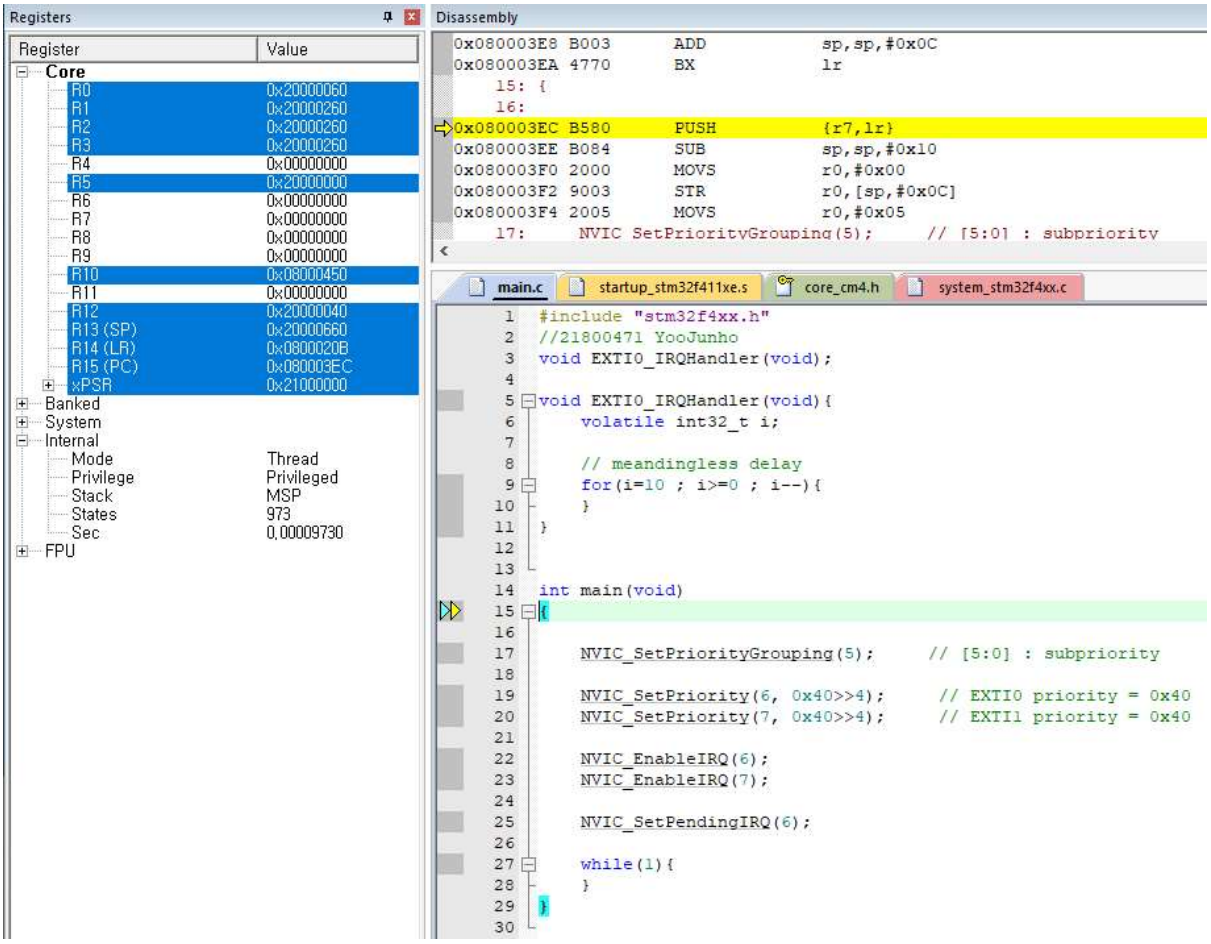
# 임베디드 프로세서 응용 HW3

21800471

유준호



(a) 17번, 19번, 22번과 25번 라인의 코드의 동작을 각각 설명하시오. (**8점**)

[ NVIC_SetPriorityGrouping(5) ]

Cortex-M 프로세서에서는 우선순위는 그룹 우선순위와 하위 우선순위로 나뉜다. 그 둘을 구분하는 것은 SCB에 위치한 AIRCB의 비트 [10:8]에 위치한 PRIGROUP값이다. PRIGROUP에 설정한 값이 5이면 우선순위 8비트 중에서 하위 6비트([5:0])가 하위 우선순위에 속하게 된다. NVIC_SetPriorityGrouping(5)에서는 우선순위 그룹을 5로 설정하여 하위 우선순위에 8비트 중 [5:0]비트가 속하게 된다.

[ NVIC_SetPriority() ]

IRQn=6인 인터럽트의 우선순위 값을 0x40, IRQn=7인 인터럽트의 우선순위 값을 0x00으로 설정하는 모습입니다. 이 때 >>4 비트 연산을 하는 이유는 STM32f4xx프로세서의 __NVIC_PRIO_BITS의 값이 4이므로 priority인자값을 상위 4비트값만 주어야 하기 때문이다.

[ NVIC_EnableIRQ() ]

IRQn=6 , IRQn=7의 인터럽트를 활성화합니다. 이 인터럽트가 발생할 때 인터럽트 핸들러 함수가 실행되도록 한다.

[ NVIC_SetPendingIRQ() ]

이 함수는 지정된 IRQn의 pending bit을 1로 설정한다. 펜딩이 설정된 인터럽트는 처리될 때까지 기다리는 대신 다른 작업을 수행할 수 있게됨.

(b) 디버거를 시작한 다음, 25번 라인과 5번 라인에 breakpoint를 설정하고, Peripherals -> Core Peripherals -> NVIC를 선택 다음, Idx 22번과 23번의 상태를 설명하시오. (E, P, A, Priority) (**4점**) 그리고 Interrupt Control & State register (scb->ICSR)의 주요 필드들의 상태를 설명하시오 (VECTACTIVE, RETTOBASE, VECTPENDING, ISRPREEMPT, ISRPENDING) (**5점**)

[실행 결과]

| Idx | Source | Name | E | P | A | Priority |
|---|---|---|---|---|---|---|
| 22 | EXTI Line0 interrupt | EXTI0 | 0 | 0 | 0 | 0 = 0 s0 |
| 23 | EXTI Line1 interrupt | EXTI1 | 0 | 0 | 0 | 0 = 0 s0 |

현재 EXTI0, EXTI1의 E, P, A, Priority의 필드 모두 0값이므로

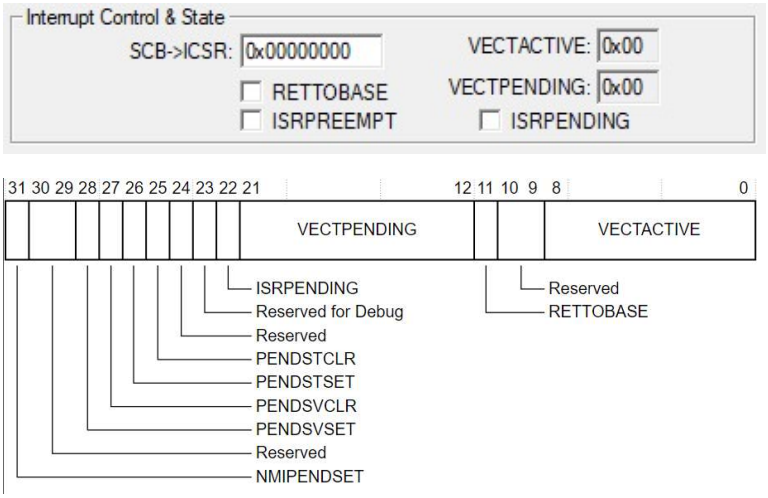
E=0 -> 두 IRQ 모두 비활성 상태이다

P=0 -> 처리할 IRQ가 없다

A=0 -> IRQ가 현재 활성화 되어 실행 중이지 않음

Priority=0 -> 우선순위가 0x00임

을 알 수있다.




VECTACTIVE : 0x00 -> IRQ가 실행 중이지 않아 현재 0값을 나타내고 있다.

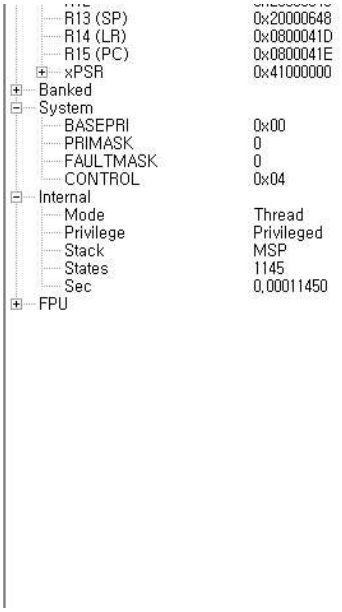
RETTOBASE : 0x0 -> 현재 실행중인 ISR에서 다른 ISR실행될 수 있음을 의미함. 현재 ISR이 끝나면 이전 ISR의 실행 위치로 돌아감(반환주소가 이전 ISR의 실행 위치로)

VECTPENDING : 0x00 -> 처리되지 않은 IRQ의 번호를 나타냄. 즉 대기중인 IRQ의 번호임. 0인 경우에는 대기중인 IRQ가 없음. 있을 때는 처리해야 할 IRQ의 번호가 저장됨.

ISRPREEMPT : 0x0 -> 현재 처리중인 IRQ에 더 높은 우선순위의 IRQ가 발생했을 때, 현재 IRQ가 중단 가능한지. 1인 경우에는 Preemption이 가능함을 의미. 지금처럼 0인 경우에는 불가능.

ISPENDING : 0x0 -> 처리되지 않은 IRQ가 있는지 여부를 나타냄. 이 필드가 1인경우, 처리되지 않은 IRQ가 있음. 0인 경우는 처리되지 않은 IRQ가 없음.

(c) Run 버튼을 클릭하시오. 그러면 25번 breakpoint에서 실행이 일단 멈출 것이다. 이때 SP, 값은 무엇인가? (**1점**) 그리고 CONTROL register 값이 무엇인지 기록하고, 그 값의 의미를 설명하시오. (**3점**) 그리고 Peripherals -> Core Peripherals -> NVIC에서 Idx 22 번과 23번의 바뀐 상태를 설명하시오. (E, P, A, Priority) (**4점**)





R13(SP) = 0x20000648

CONTROL = 0x04

= 0000 0100

Cortex-M3/M4에서는 프로세서의 동작 모드가 쓰레드, 핸들러 모드로 구성되어있음. 쓰레드 모드는 일반 소프트웨어를 실행하는 모드이고, 핸들러 모드는 익셉션을 처리하는 모드임. 핸들러 모드에서는 항상 특권을 가지며, 쓰레드 모드에서는 특권이 없는 사용자모드 또는 특권모드로 동작할 수 있음.

| CONTROL[0] : | CONTROL[1] | CONTROL[2] |
|---|---|---|
| 0 = privileged thread mode | 0 = 디폴트 스택(MSP) | 0 = 부동 소수점 관련 명령어가 사용되지 않았다. |
| 1 unprivileged(user) thread mode | 1 = 프로세스 스택(PSP) | 1 = 부동소수점 관련 명령어가 사용되었다. |

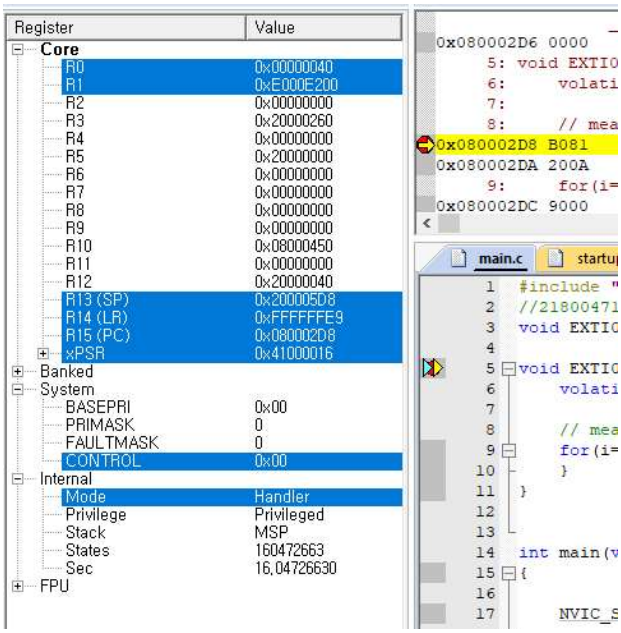
CONTROL = 0x04 = 0000 0100(2진수)

여기서 0비트는 0, 1비트는 0, 2비트는 1이다. 따라서 부동소수점 관련 명령어가 사용됨을 알 수 있다.

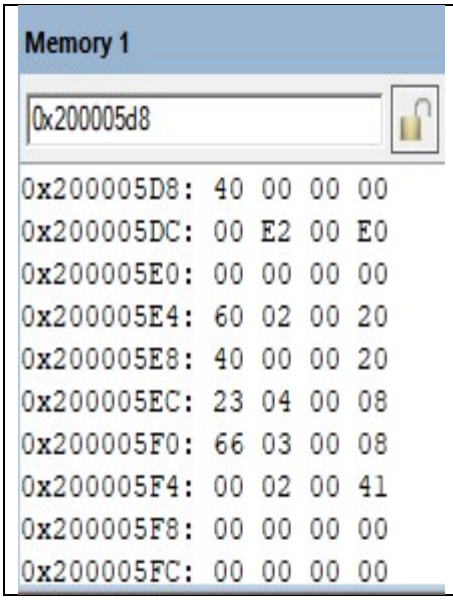
| Idx | Source | Name | E | P | A | Priority |
|---|---|---|---|---|---|---|
| 22 | EXTI Line0 interrupt | EXTI0 | 1 | 0 | 0 | 4 = 1 s0 |
| 23 | EXTI Line1 interrupt | EXTI1 | 1 | 0 | 0 | 0 = 0 s0 |

EXTI0과 EXTI1이 Enable되어있음을 알 수 있다. 또한 EXTI0의 Priority값이 0x40의 상위 4비트인 4로 바뀌어 있음을 알 수 있다.

(d) Run 버튼을 클릭하시오. 그러면 5번 breakpoint에서 실행이 일단 멈출 것이다. 이때 SP 값은 무엇인가? Stack에 저장된 내용들을 설명하시오. (**7점**) 그리고 Peripherals -> Core Peripherals -> NVIC와 Interrupt Control & State register (scb->ICSR)의 주요 필드들에서 바뀐 상태 상태와 (**4점**) LR에 나타난 EXC_RETURN 값의 의미를 설명하시오. (**4점**)

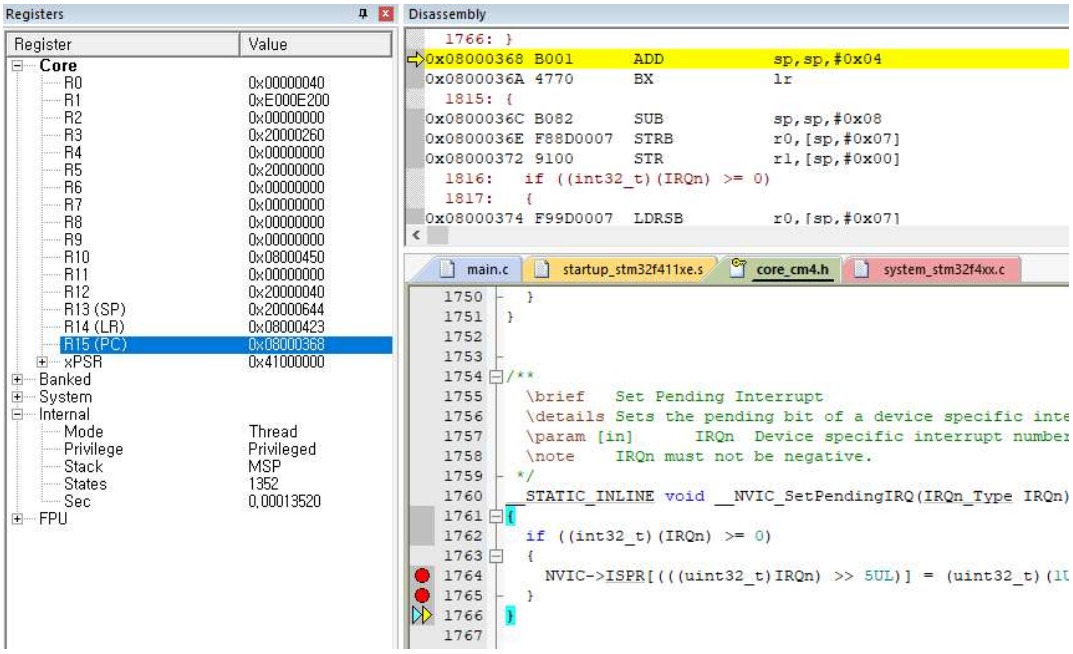


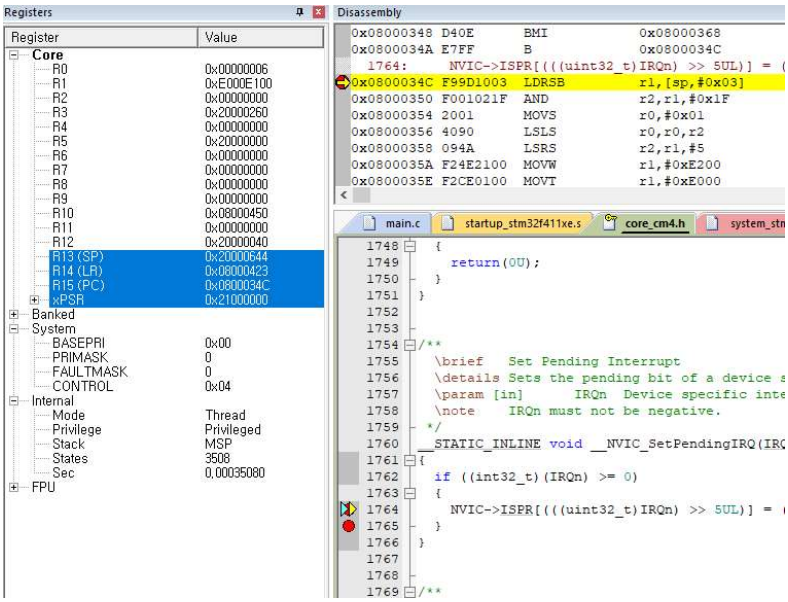

R13(SP) = 0x200005D8



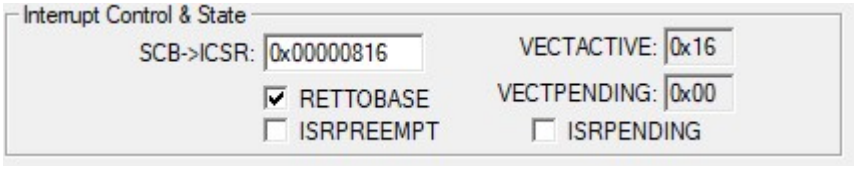

R0 : 0x00000040
R1 : 0xE000E200
R2 : 0x00000000
R3 : 0x20000260
R12 : 0x20000040
LR : 0x08000423
PC : 0x08000366
xPSR : 0x41000200

서브 인터럽트로 들어가기전 사용하는 레지스터를 스태킹하여 보존하는 모습을 볼 수 있다

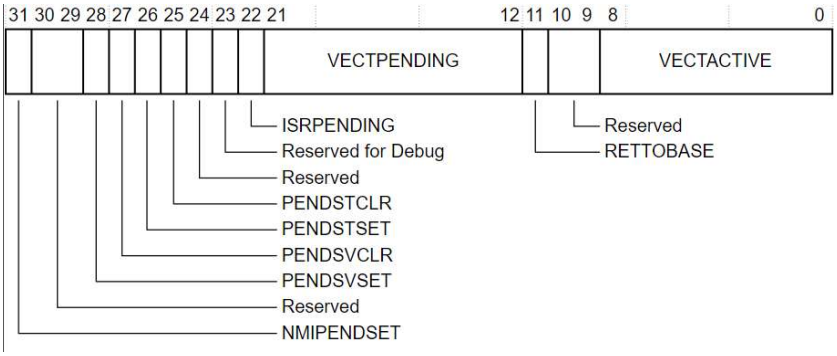



| | | | | | | |
|---|---|---|---|---|---|---|
| 23 | EXTI Line1 interrupt | EXTI1 | 1 | 0 | 0 | 0 = 0 s0 |
| 24 | EXTI Line2 interrupt | EXTI2 | 0 | 0 | 0 | 0 = 0 s0 |



0x00000816 => 1000 0001 0110(2)




RETTOBASE의 박스가 체크되어있다. 그 이유는 위의 ICSR 비트표에 따라 ICSR[11]이 RETTOBASE에 해당하는 비트이기 때문이다. 1000 0001 0110의 12번째 비트의 값은 1이므로 RETTOBASE는 1이다. 또한 VECTACTIVE에 해당하는 ICSR[8:0]에 해당하는 곳의 값이 0x0 0001 0110이기 때문에 VECTACTIVE의 값은 0x16이 된다. VECTPENDING 영역은 모두 0이기 때문에 0x00이다.

R14 (LR) 0xFFFFFFE9

= 1111 1111 1111 1111 1111 1110 1111 1001

LR[31:4] = 0xFFFFFFE

LR[3] = 1(thread return mode)

= 0 (handler return mode)

LR[2] = 1(Process stack)

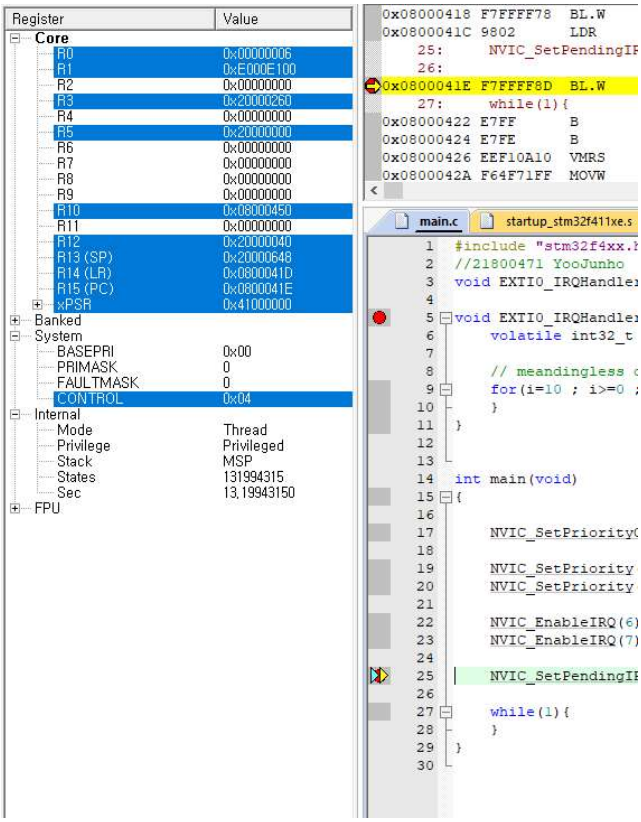
= 0(main stack)

LR[1] – reserved

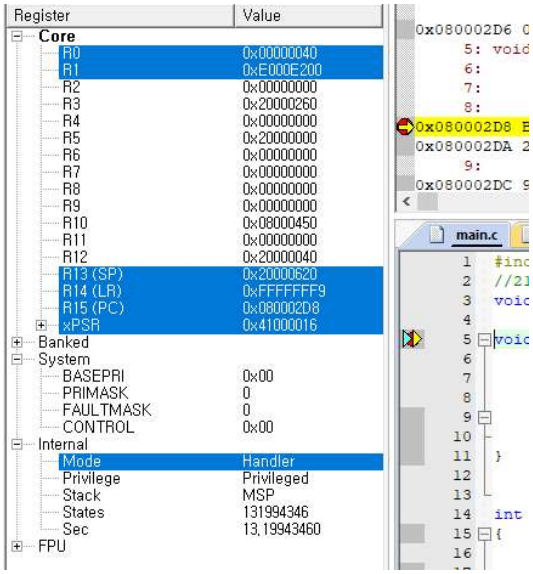
LR[0] = 1

CCR[9]의 값이 0이다. 즉 스택이 4바이트 즉 워드 단위로 정렬이 된다는 의미이다.

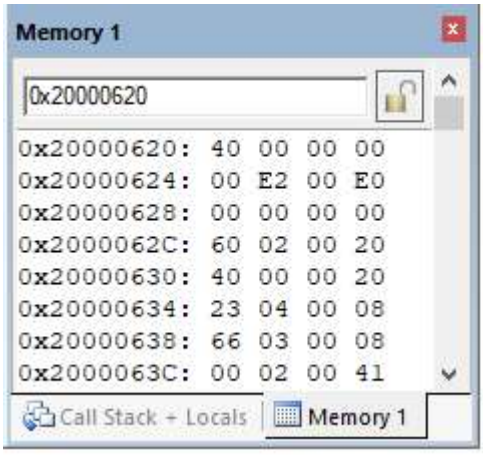
(e) 디버거를 reset시키고 (RSR 버튼 클릭) Run 버튼을 클릭하시오. 그러면 25번 breakpoint에서 실행이 일단 멈출 것이다. ((c)에서의 SP 값이 같음을 일단 확인하시오.) 이 때 CONTROL register의 값을 0x00으로 변경한 다음, Run 버튼을 클릭하시오. 그러면 5번 breakpoint에서 실행이 일단 멈출 것이다. 이때 SP 값은 무엇인가? 그리고 Stack에 저장된 내용들을 설명하시오. (**5점**) 그리고 LR에 나타난 EXC_RETURN 값의 의미를 설명하시오. (**2점**)




SP = 0x20000648 (C에서와 같음)




SP의 값이 0x20000620이 되었음




R0, R1, R2, R3, R12값이 낮은 주소값 으로부터 큰 주소값으로 저장되어 있으며

(f) (d)와 (e)의 결과가 다른 이유를 설명하시오. (**5점**)

| CONTROL[0] :<br>0 = privileged thread mode<br>1 unprivileged(user) thread mode | CONTROL[1]<br>0 = 디폴트 스택(MSP)<br>1 = 프로세스 스택(PSP) | CONTROL[2]<br>0 = 부동 소수점 관련 명령어가 사용되지 않았다.<br>1 = 부동소수점 관련 명령어가 사용되었다. |
|---|---|---|

CONTROL register의 값을 0x04에서 0x00으로 바꿨기 때문에 부동 소수점 관련 명령어가 사용되지 않았다.