
Embedded System Design

실습 4

Cho Yeongpil
Hanyang University



9주차 실습 수업 개요

- 9-1회차 : Startup Code 작성
- 9-2회차 : UART & TIMER 구현 Code 작성
- 각 회차별로 과제있음



목차

1. Data Sheet
2. Startup Code



DATA SHEET



Data Sheet

- **Data sheet란?**

- 부품, 하부시스템, 소프트웨어 등의 성능과 특성 등을 모아 놓은 문서
- 제조사에서 만들어서 배포

- **Data sheet에 들어가는 정보**

- 제품의 특성
- 간단한 기능 설명
- 핀 접속 다이어그램
- 공급전압, 전력 소비량, 입력 전류 등의 최대/최소 값
- 입/출력 파형도
- ...

수업에 필요한 Data Sheet

- **DDI0344K_cortex_a8_r3p2_trm.pdf**
 - ARM CORTEX A8의 데이터시트
 - CORTEX A8 프로세서에 관한 정보를 기재
 - 프로세서가 제공하는 기능
 - 레지스터 설정을 통한 해당 기능의 사용법
 - ...
- **S5PC100_UM_REV101.pdf**
 - SAMSUNG S5PC100 어플리케이션 프로세서의 데이터시트



STARTUP CODE



VPOS 커널을 포팅하기 위한 준비

1. 커널 컴파일 + 커널 이미지를 RAM에 적재
2. **Startup code 작성**
3. UART 설정
4. TIMER 설정
5. **Hardware Interrupt Handler 구현**
 - (1) UART Interrupt
 - (2) Timer Interrupt
6. **Software Interrupt Entering/Leaving Routine 구현**



Startup code

- **Startup code?**
 - ASM 코드 (어셈블리 코드)
 - 임베디드 타겟 보드의 초기화
 - C 소스에서 액세스하기 힘든 초기화 과정을 처리
 - C코드의 main 함수가 실행되기 전에 먼저 실행
 - 코드 마지막에 branch 명령어를 사용하여 main 함수를 실행
- **Startup Code에서 처리하는 작업**
 - Variable 초기화
 - PLL(Phased-Locked Loop) 설정
 - Memory 설정
 - Stack 설정
 - UART 등 주변 장치 설정
 - C 코드로 점프
- **소스 코드 파일의 위치**
 - hal/cpu/HAL_arch_startup.S



HAL_arch_startup 파일

❑ HAL_arch_startup 파일

- Startup code와 HAL 관련 code로 나뉘어짐

• Startup code

- 벡터 테이블
- Variable 초기화
- 캐시와 메모리 정책을 설정
- CPU 모드마다 스택을 할당

• HAL 관련 code

- SWI
- HWI
- Context Switching



HAL_arch_startup 파일의 분석

• 심볼 정의

- Startup code에서 사용할 외부변수, 외부함수, 전역변수 등을 정의

.extern : 외부 심볼/레이블을 참조
해당 심볼이 다른 모듈에 정의되어 있음

.global : 전역 심볼/레이블을 정의

.equ : 심볼에 값을 할당
심볼을 참조하기 전에 미리 값을 할당해야 함

```
#include "../include/vh_cpu_hal.h"

.extern VPOS_kernel_main
.extern vk_undef_handler
.extern vh_hwi_classifier
.extern vk_swi_classifier
.extern vk_pabort_handler
.extern vk_dabort_handler
.extern vk_fiq_handler
.extern vk_not_used_handler
.extern vk_irq_test

.extern vk_sched_save_tcb_ptr

.global vh_VPOS_STARTUP

.global vk_save_swi_mode_stack_ptr
.global vk_save_swi_current_tcb_bottom
.global vk_save_irq_mode_stack_ptr
.global vk_save_irq_current_tcb_bottom
.global vk_save_pabort_current_tcb_bottom

.global vh_restore_thread_ctx
.global vh_save_thread_ctx
.global vh_save_ctx_bottom

.equ vh_USERMODE, 0x10
.equ vh_FIQMODE, 0x11
.equ vh_IRQMODE, 0x12
.equ vh_SVCMODE, 0x13
.equ vh_ABORTMODE, 0x17
.equ vh_UNDEFMODE, 0x1b
.equ vh_MODEMASK, 0x1f
.equ vh_NOINT, 0xc0

.equ vh_userstack, 0x21200000
.equ vh_svcstack, 0x21400000
.equ vh_irqstack, 0x21600000
.equ vh_abortstack, 0x21800000
.equ vh_undefstack, 0x21a00000
.equ vh_fiqstack, 0x21c00000

.equ vh_vector_base, 0x20000044 // relocated vector table base address
.equ vh_VICBASE, 0xe4000000
```



HAL_arch_startup 파일의 분석

- **vh_VPOS_STARTUP**
 - 커널 코드 중 가장 처음 실행되는 부분
 - 링커 스크립터에서 ENTRY(vh_VPOS_STARTUP)로 설정
 - nop : No-operation. 아무 것도 하지 않는 명령어
 - 58 line의 'b vh_VPOS_reset' 명령어를 통해 리셋 작업 시작

```
.text
vh_VPOS_STARTUP:
    /* Camouflaged code for imitating linux
    Linux has a header that includes 8 nop operation, branch code, magic number, binar
ffset */
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    nop
    b      vh_VPOS_reset
    magic: .long 0x016F2818      // Linux magic number
    startn: .long 0x00000000     // start address(offset) is 0
    endn:   .long 0x0000d8fc     // end address(offset) is file size(byte)
    /* Camouflaged code end */

    nop
    nop
    nop
    nop
```

HAL_arch_startup 파일의 분석

- **vh_vector_start**

- 벡터 테이블
- Exception이 발생하면,
 - CPU는 벡터 테이블의 베이스 주소에 exception의 오프셋을 더하여 해당 exception handler를 실행
- ctags를 설치했다면, 레이블에 커서를 올리고 'ctrl키+] '로 해당 레이블로 이동할 수 있음

Exception	Mode	Vector table offset
Reset	SVC	+0x00
Undefined Instruction	UND	+0x04
Software Interrupt(SWI)	SVC	+0x08
Prefetch Abort	ABT	+0x0c
Data Abort	ABT	+0x10
Not assigned	-	+0x14
IRQ	IRQ	+0x18
FIQ	FIQ	+0x1c

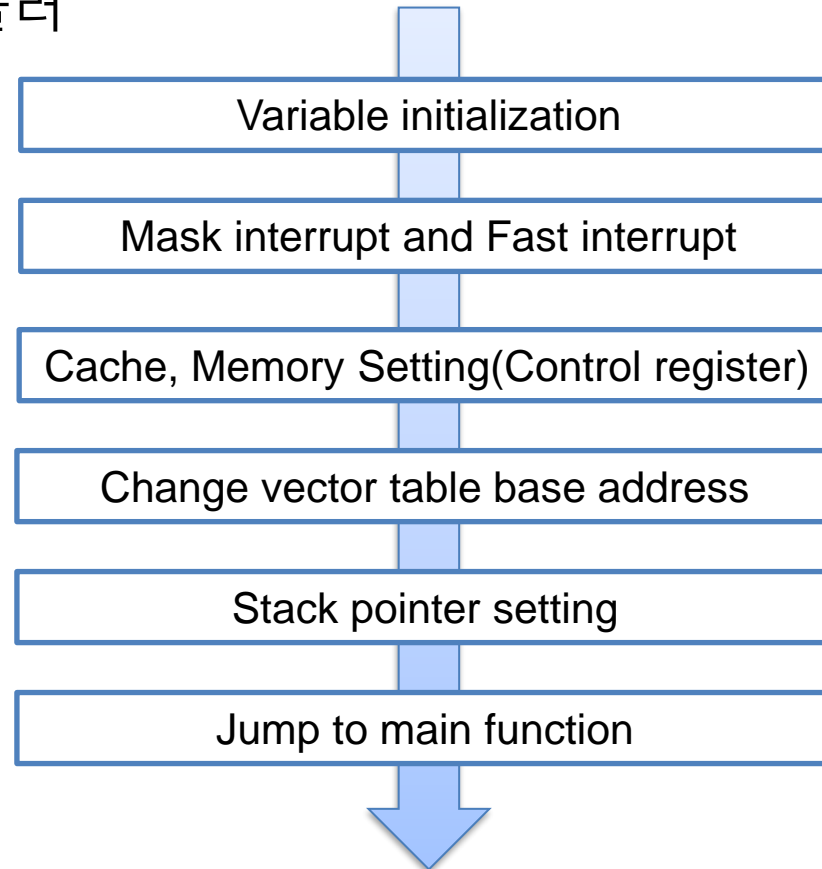
vh_vector_start:

```
b      vh_UP0S_reset
b      vk_undef
b      vh_software_interrupt
b      vh_pabort
b      vk_dabort
b      vk_not_used_handler
b      vh_irq
b      vk_fiq_handler
```



HAL_arch_startup 파일의 분석

- **vh_VPOS_reset**
 - 리셋 핸들러



vh_VPOS_reset 분석

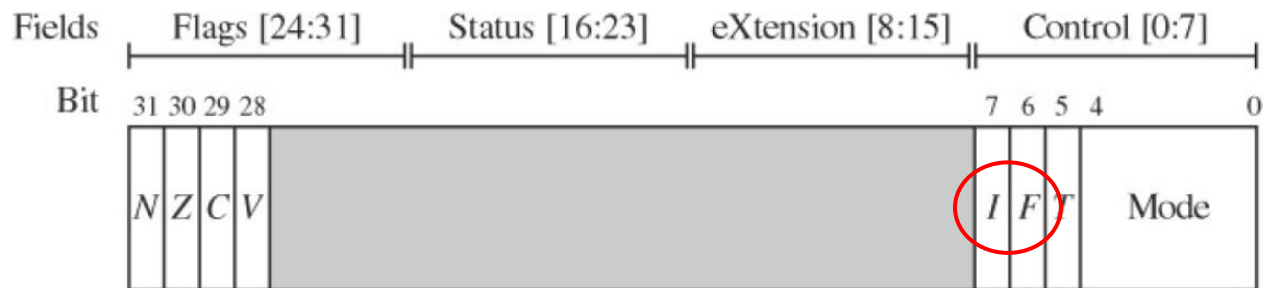
- Variable initialization

- 레지스터 r0를 0으로 초기화
- HAL code에서 사용할 변수를 초기화

```
vh_VPOS_reset:
    // variable initialization
    mov     r0, #0x00
    str     r0, vk_save_swi_mode_stack_ptr
    str     r0, vk_save_swi_current_tcb_bottom
    str     r0, vk_save_irq_mode_stack_ptr
```

vh_VPOS_reset 분석

- **Mask interrupt and fast interrupt**
 - CPSR의 I bit와 F bit를 1로 set
 - I bit : IRQ의 인터럽트를 마스크시킴
 - 1 : Interrupt Disable
 - 0 : Interrupt Enable
 - F bit : FIQ의 인터럽트를 마스크시킴
 - 1 : Fast interrupt Disable
 - 0 : Fast interrupt Enable



vh_VPOS_reset 분석

- Mask interrupt and fast interrupt

- Code

```
// Mask interrupt and fast interrupt
```

```
mrs      r0, cpsr
```

-----> CPSR를 r0에 복사

```
orr      r0, r0, #0xc0
```

-----> 7번 비트와 6번 비트를 1로 set

```
msr      cpsr, r0
```

-----> r0를 CPSR로 복사

- m : move / r : register / s : program status register

- mrs = mov reg PSR / msr = mov PSR reg



vh_VPOS_reset 분석

- Invalidate all instruction caches

- Code

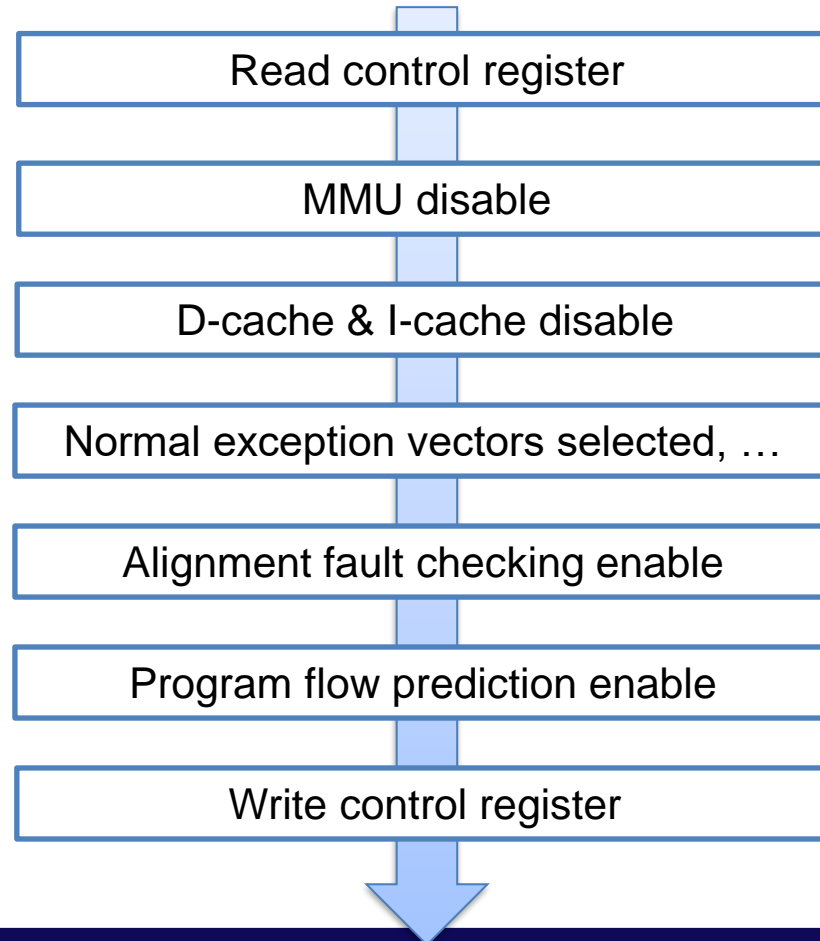
```
// Invalidate all instruction caches
mov     r0, #0x00
mcr     p15, 0, r0, c7, c5, 0
```

- Cortex Data sheet (page. 88)

CRn	Op1	CRm	Op2				
c7	0	c0	0-3	Undefined	-	-	-
			4	NOP (WFI)	WO	WO	-
			5-7	Undefined	-	-	-
		c1-c3	0-7	Undefined	-	-	-
		c4	0	Physical Address	R/W	R/W, B	0x00000000
			1-7	Undefined	-	-	-
		c5	0	Invalidate all instruction caches to point of unification	WO	WO	-

vh_VPOS_reset 분석

- Control Register(c1) Setting



vh_VPOS_reset 분석

- **Control Register(c1) Setting**

- Data sheet

- Page. 122 ~ 125 참고
 - Read control register (p.124)

To access the Control Register, read or write CP15 with:

`MRC p15, 0, <Rd>, c1, c0, 0 ; Read Control Register`

- Write control register (p. 125)

`MCR p15, 0, <Rd>, c1, c0, 0 ; Write Control Register`

- 나머지 setting은 Table 3-46 Control Register bit functions 참고 (p. 123~124)

- m : move / r : register / c : control register

vh_VPOS_reset 분석

- Control Register(c1) Setting

- Code

```
// Control Register Setting
```

<code>mrc</code>	<code>p15, 0, r0, c1, c0, 0</code>	----->	Read Control Register
<code>bic</code>	<code>r0, r0, #0x01</code>	----->	MMU disable
<code>bic</code>	<code>r0, r0, #0x04</code>	----->	D-cache disable
<code>bic</code>	<code>r0, r0, #0x1000</code>	----->	I-cache disable
<code>bic</code>	<code>r0, r0, #0x2000</code>	----->	Normal exception vectors, base address 0x00000000
<code>orr</code>	<code>r0, r0, #0x02</code>	----->	Alignment fault checking enable
<code>orr</code>	<code>r0, r0, #0x800</code>	----->	Program flow prediction enable
<code>mcr</code>	<code>p15, 0, r0, c1, c0, 0</code>	----->	Write Control Register

vh_VPOS_reset 분석

- **Change vector table base address**

- 벡터 테이블의 베이스 주소를 저장
- exception이 발생하면 저장된 베이스 주소에 해당 exception의 오프셋을 더하여 핸들러로 점프
- Data sheet
 - Page. 195~196 참고

MCR p15, 0, <Rd>, c12, c0, 0 ; Write Secure or Nonsecure Vector Base
; Address Register



vh_VPOS_reset 분석

- Change vector table base address

- 베이스 주소 : 0x20008044

```
.text
vh_VPOS_STARTUP:
    /* Camouflaged code for imitating linux
       Linux has a header that includes 8 nop operation, branch code, magic
       number, binary file start offset, and file size(end offset) */
    nop ← 0x20008000
    nop
```

```
    nop
    nop
vh_vector_start:
    b    vh_VPOS_reset ← 0x20008044
    b    vk_undef
    b    vh_software_interrupt
    b    vh_pabort
```

- Code

```
// change vector table base address (0x20008044)
ldr    r0, =vh_vector_base
mcr    p15, 0, r0, c12, c0, 0
```

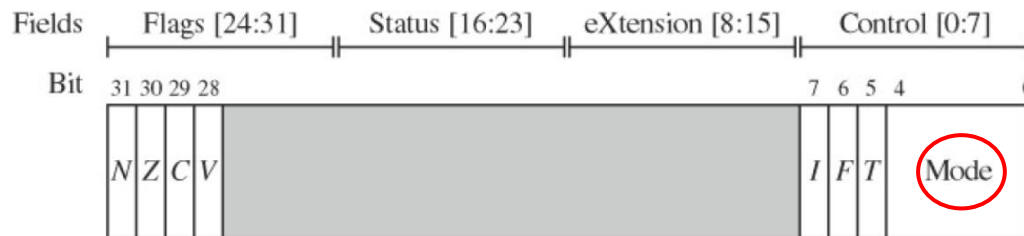
vh_VPOS_reset 분석

- Stack pointer setting

- ARM CPU는 Abort, FIQ, IRQ, Supervisor, System, Undefined, User 모드를 가짐
- 각 모드별로 스택 포인터(r13, sp)에 스택 시작 위치를 설정

- Setting Flow

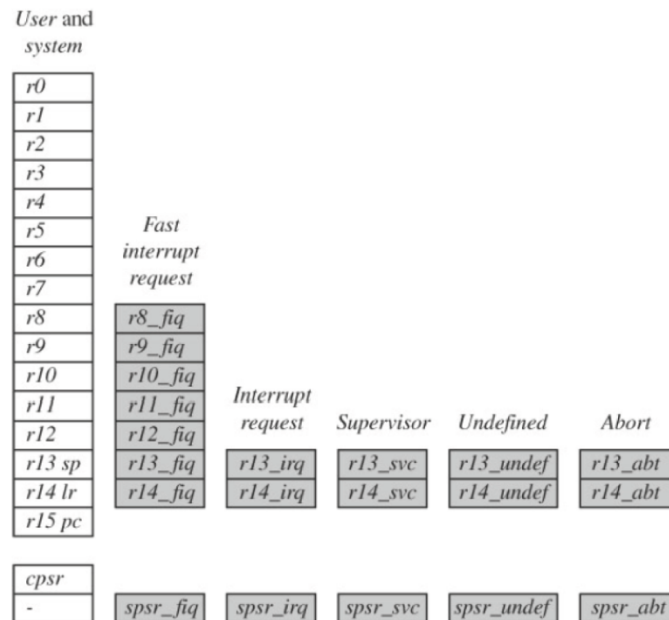
- cpsr를 r0에 복사 (mrs 명령어)
- 프로세스 모드를 표시하는 5 bits [4:0]을 수정하여 CPU 모드 변경
- 해당 모드의 스택 포인터에 스택 시작 위치를 저장



vh_VPOS_reset 분석

- Stack pointer setting

- ARM은 각 모드별로 sp와 lr를 가지고 있음
 - Undef 모드에서 sp를 바꿔도 다른 모드의 sp는 전혀 영향을 받지 않음
 - 특정 모드의 sp를 수정하고 싶으면 해당 모드로 진입해야 함



vh_VPOS_reset 분석

- Stack pointer setting

- Code

각 모드 비트와 인터럽트 마스크 비트를 나타내는 심볼은 .equ로 값이 할당되어 있음

Ex) vh_UNDEFMODE = 0x1b

vh_MODEMASK = 0x1f

vh_NOINT = 0xc0

CPU 모드의 변경법 :

1. mrs 명령어로 CPSR을 가져옴
2. 모드 비트와 인터럽트 마스크 비트를 0으로 클리어
3. 원하는 모드의 모드 비트를 설정하고 인터럽트 마스크 비트도 함께 set
4. msr 명령어로 CPSR에 저장

```
// stack pointer setting
mrs r0,cpsr

bic r0,r0,#vh_MODEMASK|vh_NOINT
orr r1,r0,#vh_UNDEFMODE|vh_NOINT
msr cpsr_cxsf,r1
ldr sp,=vh_undefstack

bic r0,r0,#vh_MODEMASK|vh_NOINT
orr r1,r0,#vh_ABORTMODE|vh_NOINT
msr cpsr_cxsf,r1
ldr sp,=vh_abortstack

bic r0,r0,#vh_MODEMASK|vh_NOINT
orr r1,r0,#vh_IRQMODE|vh_NOINT
msr cpsr_cxsf,r1
ldr sp,=vh_irqstack

bic r0,r0,#vh_MODEMASK|vh_NOINT
orr r1,r0,#vh_FIQMODE|vh_NOINT
msr cpsr_cxsf,r1
ldr sp,=vh_fiqstack

bic r0,r0,#vh_MODEMASK|vh_NOINT
orr r1,r0,#vh_SVCMODE|vh_NOINT
msr cpsr_cxsf,r1
ldr sp,=vh_svcstack
```



vh_VPOS_reset 분석

- 과제

- USER 모드의 sp를 설정하기
- 모드 비트는 'vh_USERMODE'를 사용 `.equ vh_USERMODE, 0x10`
- 단, USER 모드에서는 인터럽트를 Enable 해야함!!
- Startup code에 작성하고, 보고서로 제출



vh_VPOS_reset 분석

- Jump to main function

- Code

`b VPOS_kernel_main`

- Branch 명령어로 VPOS_kernel_main 함수로 점프

- VPOS_kernel_main() : VPOS 커널의 main 함수
 - vpos/kernel/kernel.start.c에 위치

- Reset 작업 종료



VPOS_kernel_main()

- Code

- VPOS 커널 데이터 구조체를 초기화
- 시리얼 장치와 타이머 등 하드웨어를 초기화
- 인터럽트 enable
- 부팅 메시지 출력
- 셸 스레드 생성
- 스케줄러 호출하는 VPOS_start 루틴으로 진입

```
void VPOS_kernel_main( void )
{
    pthread_t p_thread, p_thread_0, p_thread_1, p_thread_2;

    /* static and global variable initialization */
    vk_scheduler_unlock();
    init_thread_id();
    init_thread_pointer();
    vh_user_mode = USER_MODE;
    vk_init_kdata_struct();

    vk_machine_init();
    set_interrupt();

    printk("%s\n%s\n%s\n", top_line, version, bottom_line);

    /* initialization for thread */
    race_var = 0;
    pthread_create(&p_thread, NULL, VPOS_SHELL, (void *)NULL);
    pthread_create(&p_thread_0, NULL, race_ex_1, (void *)NULL);
    pthread_create(&p_thread_1, NULL, race_ex_0, (void *)NULL);
    pthread_create(&p_thread_2, NULL, race_ex_2, (void *)NULL);

    VPOS_start();

    /* cannot reach here */
    printk("OS ERROR: VPOS_kernel_main( void )\n");
    while(1){}
```



보고서 제출

- 보고서

- 학과, 학번, 이름
- 수업 중 작성한 코드도 첨부
 - hal/cpu/HAL_arch_startup 파일의 startup code 부분만 첨부
 - Jump to main function 부분까지 첨부
- 과제로 내준 코드를 작성하고, 설명



수고하셨습니다.

