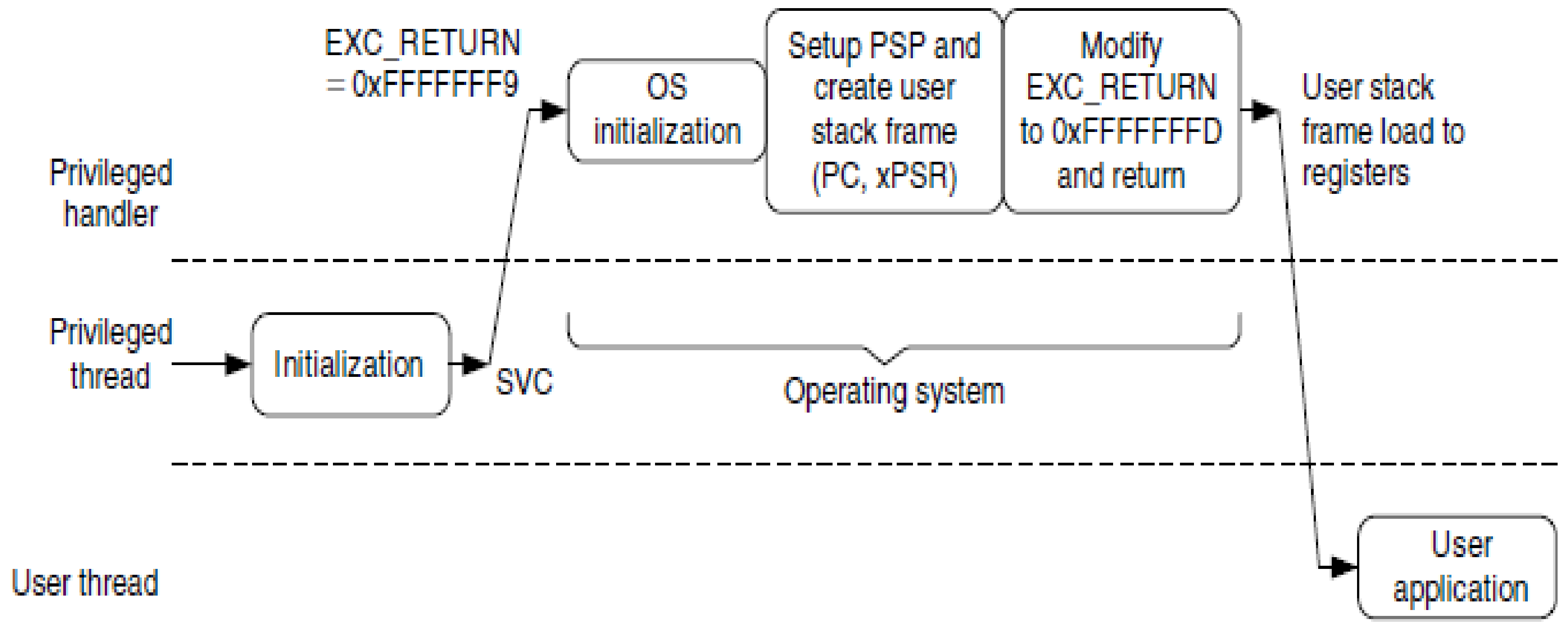
ECE254 Lab0 Tutorial

RL-RTX Kernel Programming Tutorial

Irene Huang

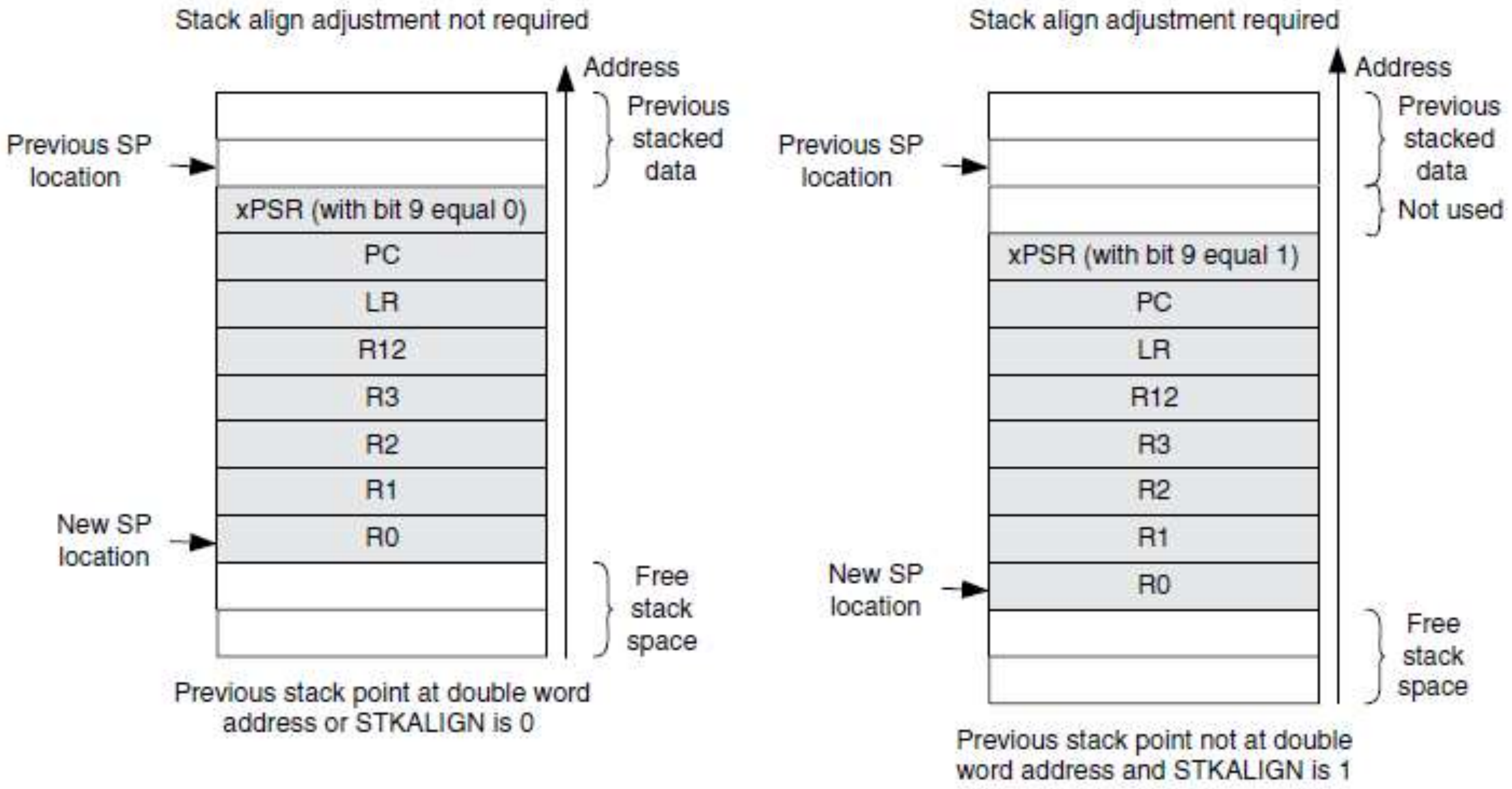
(last updated: 2013/05/20)

OS Initialization Mode Switch



(Image Courtesy of [1])

Exception Stack Frame



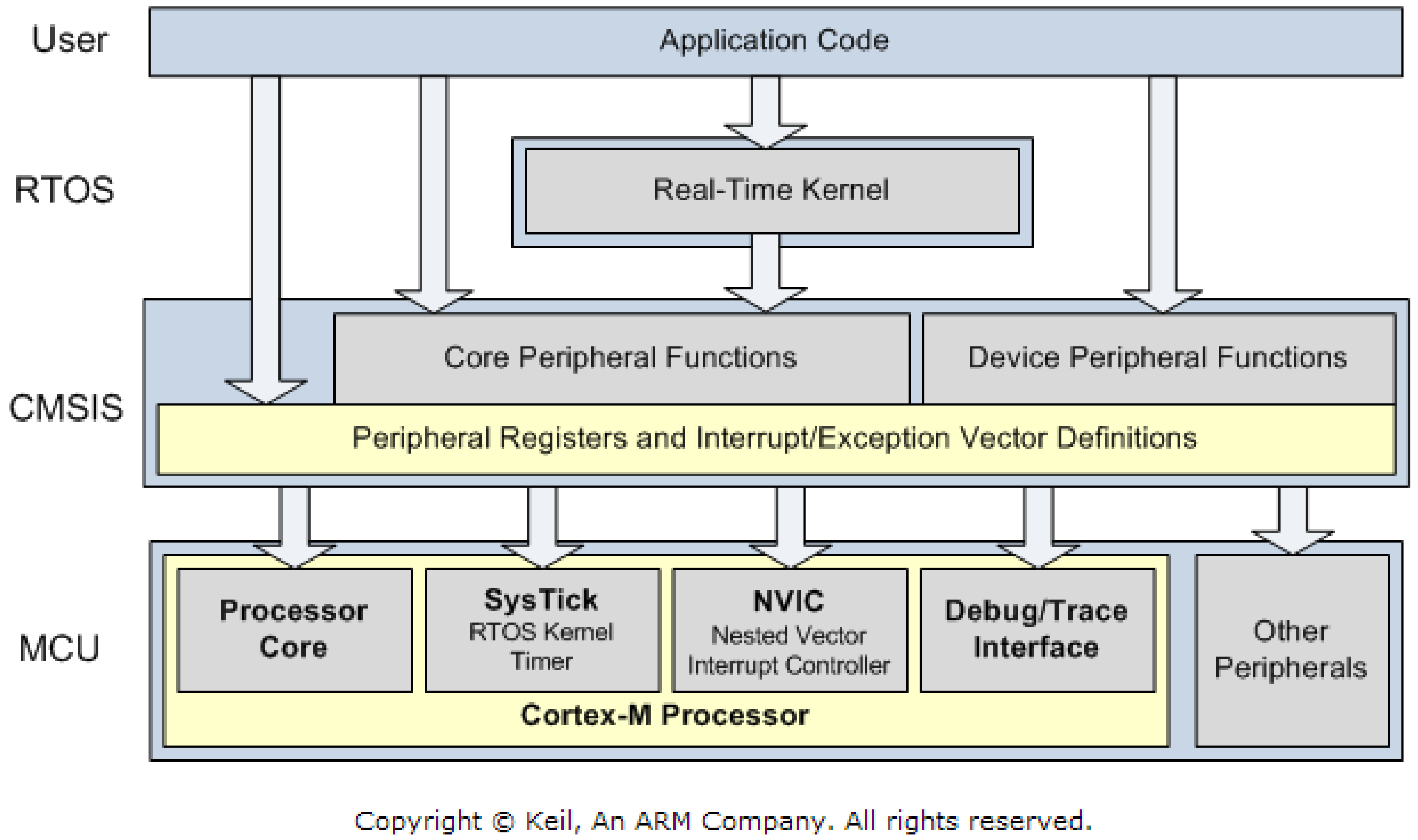
Stack align adjustment not required
Address
Previous stacked data
Previous SP location
xPSR (with bit 9 equal 0)
PC
LR
R12
R3
R2
R1
R0
New SP location
Free stack space
Previous stack point at double word address or STKALIGN is 0
Stack align adjustment required
Address
Previous stacked data
Not used
Previous SP location
xPSR (with bit 9 equal 1)
PC
LR
R12
R3
R2
R1
R0
New SP location
Free stack space
Previous stack point not at double word address and STKALIGN is 1

(Image Courtesy of [1])

AAPCS (ARM Architecture Procedure Call Standard)

- R0-R3 , R12
  - Input parameters Px of a function. R0=P1, R1=P2, R2=P3 and R3=P4
  - **R0** is used for **return value** of a function
- R12, SP, LR and PC
  - R12 is the Intra-Procedure-call scratch register.
- R4-R11
  - Must be preserved by the called function. C compiler generates push and pop assembly instructions to save and restore them automatically.

CMSIS Structure


User
Application Code
RTOS
Real-Time Kernel
CMSIS
Core Peripheral Functions
Device Peripheral Functions
Peripheral Registers and Interrupt/Exception Vector Definitions
MCU
Processor Core
SysTick RTOS Kernel Timer
NVIC Nested Vector Interrupt Controller
Debug/Trace Interface
Cortex-M Processor
Other Peripherals
Copyright © Keil, An ARM Company. All rights reserved.

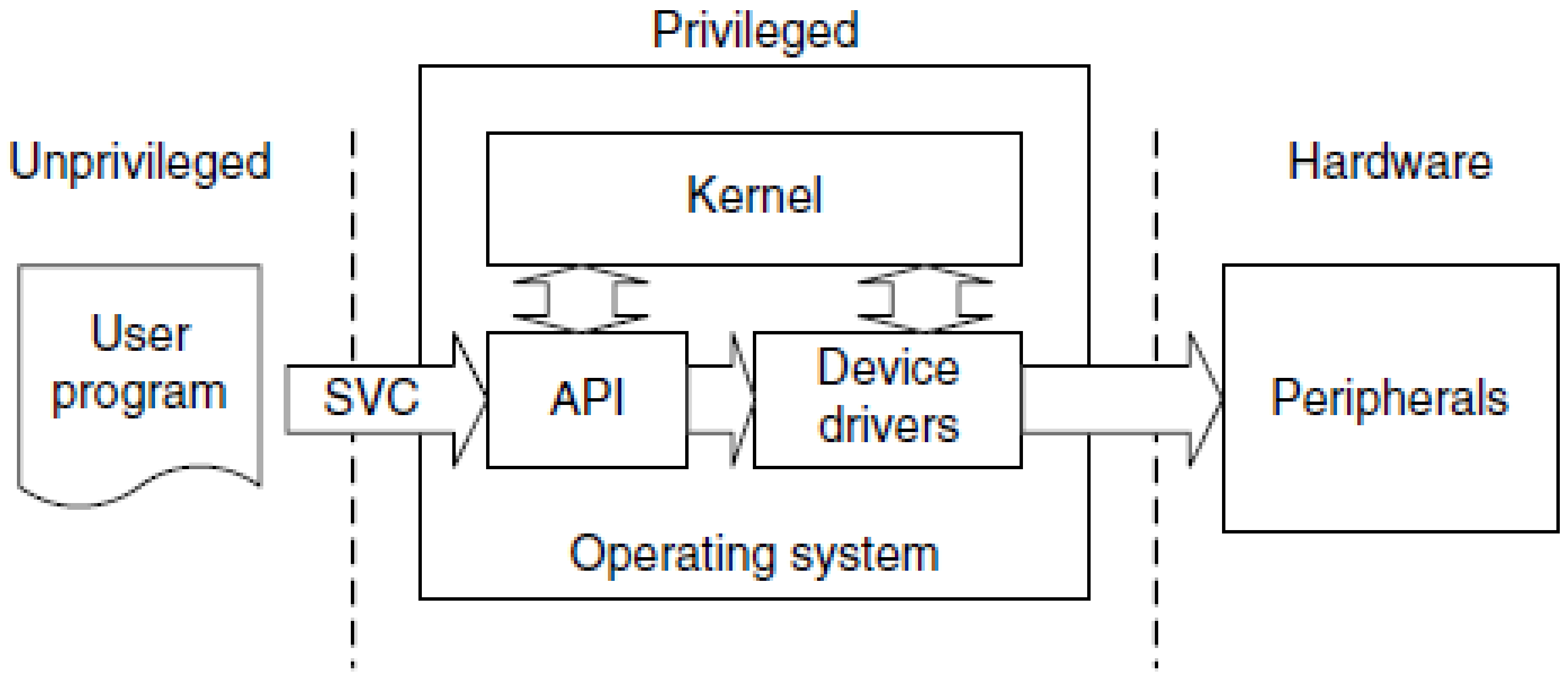
(Image Courtesy of MDK-ARM Primer V4.60)

Exception Handler Programming

- Hardware Abstraction Layer file: `HAL_CM3.c`

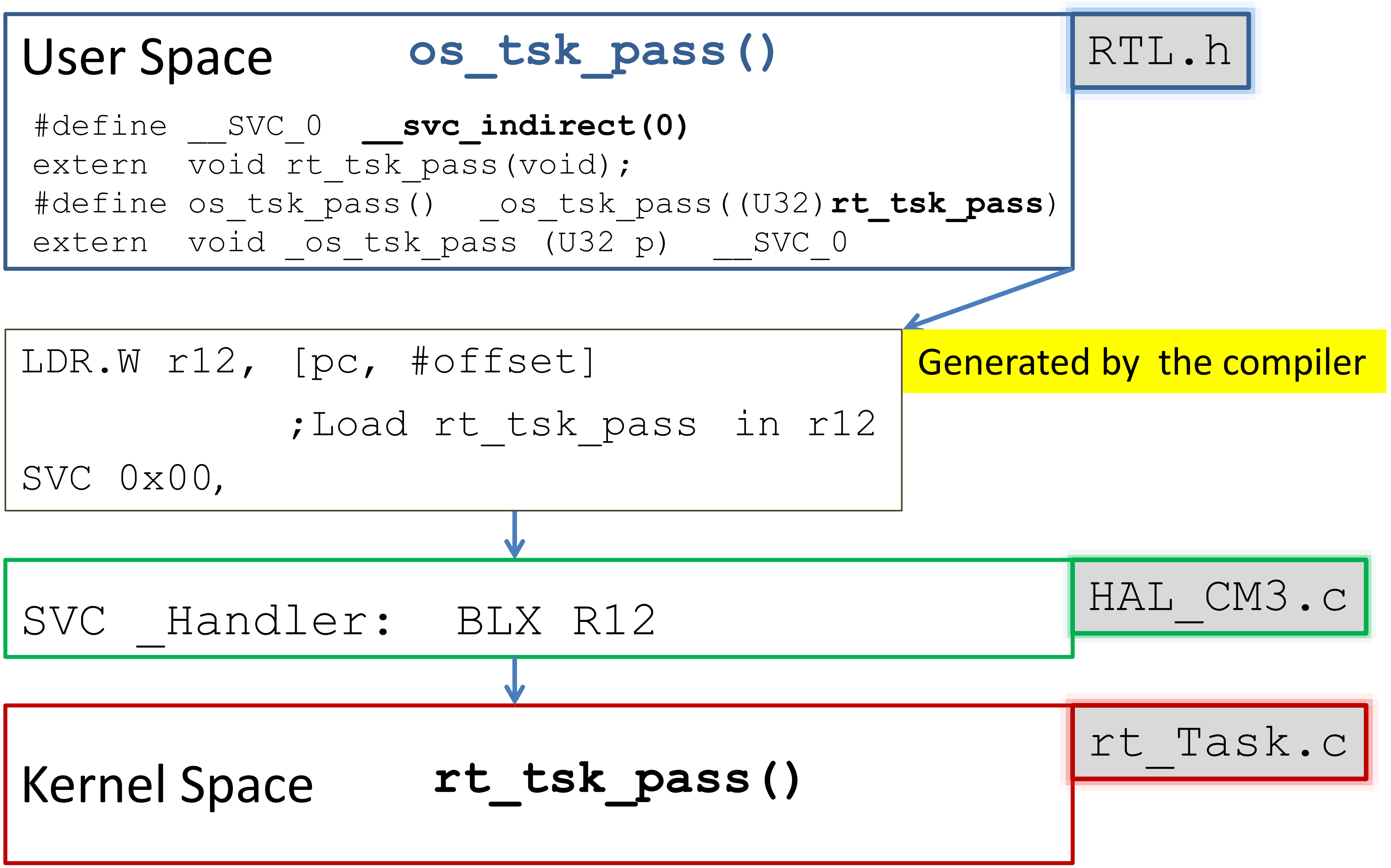
```
__asm void SVC_Handler (void) {
    PRESERVE8                  ;8 byte alignment of the stack

    IMPORT  SVC_Count          ; an external ASM symbol
    IMPORT  SVC_Table          ; an external ASM symbol
    IMPORT  rt_stk_check       ; an external C symbol

    MRS     R0,PSP             ; Read PSP
    LDR     R1,[R0,#24]        ; Read Saved PC from Stack
    LDRB    R1,[R1,#-2]        ; Load SVC Number
    CBNZ    R1,SVC_User        ; if SVC# != zero, goto SVC_User

    LDM     R0,{R0-R3,R12}; Read R0-R3,R12 from stack
    BLX     R12                ; Call SVC Function
    ; omit the rest of the code below
}
```

SVC as a Gateway for OS Functions



(Image Courtesy of [1])

System calls through SVC in C



User Space **os_tsk_pass()**

RTL.h

```
#define  __SVC_0   __svc_indirect(0)
extern   void rt_tsk_pass(void);
#define  os_tsk_pass()   _os_tsk_pass((U32)rt_tsk_pass)
extern   void _os_tsk_pass (U32 p)   __SVC_0
```

```
LDR.W r12, [pc, #offset]
              ;Load rt_tsk_pass  in r12
SVC 0x00,
```

Generated by the compiler

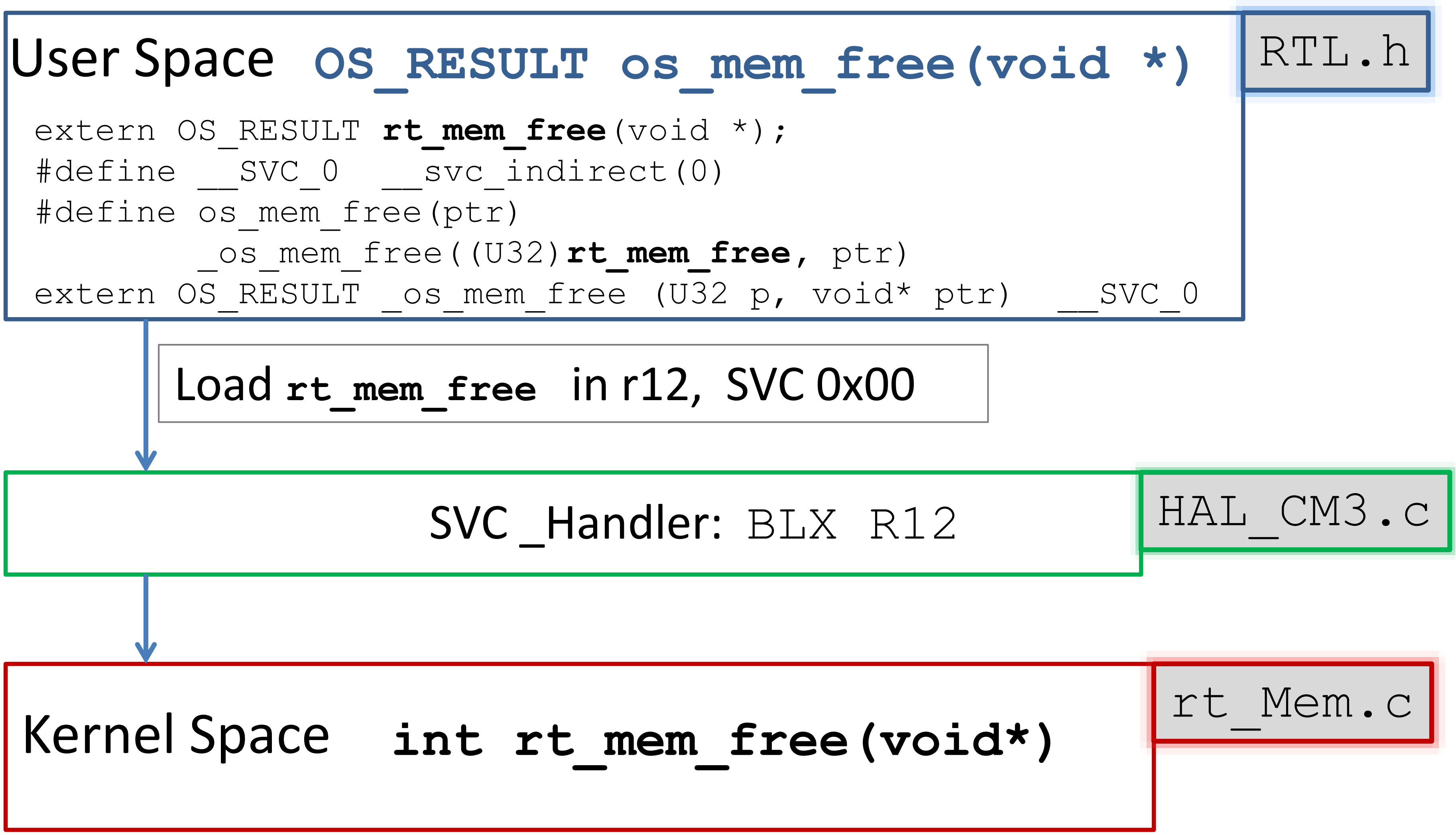
```
SVC _Handler:   BLX R12
```

HAL_CM3.c

Kernel Space **rt_tsk_pass()**

rt_Task.c

rt_Mem.c



```
OS_RESULT os_mem_free(void *)
extern OS_RESULT rt_mem_free(void *);
#define __SVC_0   __svc_indirect(0)
#define os_mem_free(ptr)
        _os_mem_free((U32)rt_mem_free, ptr)
extern OS_RESULT _os_mem_free (U32 p, void* ptr)  __SVC_0
```

```
int rt_mem_free(void*)
```

RL-RTX Kernel Files

- RL-RTX Kernel Source Code
  - C:\Software\Keil\ARM\RL\RTX\SRC\CM
  - No standard C library function calls
- Add kernel files as part of your RTX Lib project
  - Do not add HAL_CM1.c
  - Do not add HAL_CM4.c
- Do not specify the RTX as the OS
- MicroLib is optional
- In Practice, build kernel library and link with it.

References

1. Yiu, Joseph, *The Definite Guide to the ARM Cortex-M3*, 2009
2. *RealView® Compilation Tools Version 4.0 Developer Guide*
3. *ARM Software Development Toolkit Version 2.50 Reference Guide*
4. *LPC17xx User's Manual*