1.6 Assignment

1.6.1 Questions

The RTX\_lib.c is located at ARM\RV31\INC under the default Keil installation directory3.

The variable os\_active\_TCB is de\_ned in this \_le and mainly used in rt\_Task.c \_le.

The rt\_TypeDef.h has the kernel data structure de\_nitions. Read rt\_TypeDef.h and

rt\_Task.c \_les and answer the following questions.

1. (1 point) Which global variable has the os\_active\_TCB array length information?

**os\_maxtaskrun**

2. (1 point) What does os\_active\_TCB array in RTX\_lib.c contain?

**It contains an array of active task pointers.**

3. (1 point) Which global variable has the os idle task TCB information?

**os\_idle\_TCB**

4. (1 point) Is the TCB of the os idle task an element in the os\_active\_TCB?

**No**

5. (1 point) Read rt\_get\_TID() code in rt\_Task.c \_le. Assume that a non-idle task

has a task ID of n, what is the index of this task's TCB in the os\_active\_TCB array?

**n-1**

6. Read the rt\_TypeDef.h \_le and answer the following questions.

\_ (1 point) What is the purpose of the p\_lnk variable in the struct OS\_TCB?

**When the task’s state is ready and waiting for a semaphore, it needs an address on waiting list. This p\_lnk contains the pointer to that address on the waiting list.**

\_ (1 point) What is the purpose of tsk\_stack and stack variables in struct OS\_TCB?

**During task switching action, the current task’s data in the register will be saved to somewhere else in the memory. The tsk\_stack variable is the value for R13 and the value stored in R13 is a pointer pointing to the memory block where the top stack address is. However, the \*stack pointer is the pointer pointing to the starting address of the stack (bottom of the stack).**

\_ (1 point) If you have a variable with type of struct OS\_XCB \*, will casting this variable to struct OS\_TCB \* keep the p\_lnk field?

**It will keep this field**

7. Read the rt\_Task.c and RTX\_lib.c \_les and answer the following question.

\_ (2 points) What is the purpose of variables mp\_tcb and mp\_stk?

You may want to search the entire source code to further explore how these variables

are set and used (see Figure 1.28).

**The purpose of mp\_tcb is to allocate a number of memory blocks for TCB tasks. The purpose of mp\_stk is to allocate a number of memory blocks for the stack.**

8. Read Section C.2.3 and study rt\_init\_stack() and rt\_get\_PSP() functions source

code in HAL\_CM3.c \_le and answer the following questions.

\_ (2 points) Which registers are saved on the task stack?

**From R0 to R15 registers which includes LR, SP.**

\_ (2 points) Which bits of which global variable contain the default task stack size

in bytes?

**The bits 2 to 31 of variable size**

\_ (2 points) How to determine the memory address of the first item that is pushed

onto a task stack?

**That is a pointer that points to the bottom of the stack, which is the variable \*stack**

\_ (2 points) For a task that is not in RUNNING state, how to determine the

memory address of the last item that is pushed onto its task stack (i.e. the top

of the stack)?

**Firstly find that not running task from *idle task list* to get the TCB instance, then get** **the pointer that points to the top of the stack, which is the variable tsk\_stack, and it is stored in R13.**

\_ (2 points) For a task that is in RUNNING state, how to determine the memory

address of the last item that is pushed onto its task stack (i.e. the top of the

stack)?

**Firstly find the running task from os\_active\_TCB to get the TCB instance which has state is “RUNNING”, then get** **the pointer that points to the top of the stack, which is the variable tsk\_stack, and it is stored in R13.**