Review Questions and Problems Chapter 7

1. Why is *concurrency* so important in real-time computer systems?

2. There are three ways to *implement concurrency*. Name them and explain how they work.

3. Explain the difference between *POSIX processes* and POSIX *threads*. In most real-time systems you do not find POSIX processes. Please explain why not.

4. What is the purpose of the *task control block* (TCB)? Please list some of the more important parts of a TCB and explain their function.

5. In a real-time operating system, what makes up the *context* and what happens during a *context switch*?

6. Is it possible that a task that has suspended itself can resume itself again? Please explain.

7. What criteria are needed to determine the size of the run-time stack for a task and for the system stack in a multiple (nested) interrupt system?

8. Tasks in a real-time operating system can reside in a number of *states*. Sketch a typical *task state diagram* and explain the different states. Why is it not possible for a task to switch from “runnable” to “blocked”?

9. Explain the difference between an *active* and a *passive object*.

10. Consider a software system with an RTOS and a priority based scheduler. The main task has priority 60. It creates two new tasks T1 and T2 with priorities 70 and 50, respectively. It then starts Task T2, which in turn starts Task T1.

    Draw the *activity diagram* for this scenario. Draw the *timing diagram* for this scenario.

11. For the following code fragment for a priority based scheduling on a real-time operating system, how many different outputs are possible, considering different priorities?

```
void task1(void)
{
   createAndStartTask(task2, prio2);
   printf("1");
}

void task2(void)
{
   printf("2");
}

int main()
{
  createAndStartTask(task1, prio1);
  printf("0");
  exit(0);
}
```

12. In many general purpose operating systems, context switches from one task to the next are enforced when a task has exceeded a certain run time, called the “quantum”. Assuming that a context switch takes TC seconds, and the quantum is TQ seconds, give an expression for the computational efficiency η in terms of TC and TQ, assuming that the execution times of the tasks are much larger than TQ.