Examination Feedback for EEE6032 – Operating Systems
Autumn Semester 2013-14

# Feedback for EEE6032 Session: 2013-2014

**Feedback:** Please write simple statements about how well students addressed the exam paper in general and each individual question in particular including common problems/mistakes and areas of concern in the boxes provided below.  Increase row height if necessary.

**General Comments:**

The general standard of answers ranged form excellent to dire. Clearly a number of students had seriously underestimated the amount of revision that was required for this course. In fact, one student's answer to question 2 comprised nothing but an explanation of why s/he had done so badly on the exam and that they had had only allowed two days to revise, which they now realised was insufficient, because they thought the course was "easy".

A number of students tried the time-honoured trick of rewriting a question which they could not answer in a different way but including no additional information over what was given in the question. These answers attracted a mark of precisely zero!

**Question 1**:

(a) Many wrote about the microkernel philosophy of minimalism rather the implementation of a microkernel, which the question was addressing. Many though a microkernel was a good candidate for a RTOS for the mistaken reason that it is "faster".
(b) Generally well-answered although a few bizarre responses.
(c) OK but many talked about multi-level queues which the question did not address.
(d) A challenging question requiring some insight. A range of incorrect answers around the top half running in user space (not so – it's part of an interrupt handler!), "safer" execution, etc.

**Question 2**:

(a) Most got the correct answer and identified the proper reasoning for the number of processes created. The clone section was poorly done with many confusing this system call with copy-on-write, which is a more efficient implementation of the classical UNIX fork.
(b) Critical section done OK apart from a few bizarre answers about scheduling.
(c) OK apart answers about CFS being concerned with the number of pages held by a process.

**Question 3**:

(a) Generally well done although most failed to mention the dependence on a SO library in terms of exposing bugs.
(b) Symbol resolution done well. The second part on relocation was less well answered with some rather woolly accounts that failed to address the key issue of assigning unique addresses.
(c) Broadly, well answered.
(d) This flummoxed a lot of people. Such a simple but profound question!

**Question 4**:

(a) Reasonably well done but a number talked about minimising holes and page faults, which are irrelevant to the question asked.
(b) Many found this tricky. In particular, many failed to identify the key point that if the kernel does not know about threads and a user-level thread blocks, then the whole process blocks because this is all the kernel 'sees'. The part about the use of thread yield produced similarly deficient answers:  the key point is that the user-level threading has to use cooperative multitasking because there is no hardware support for threads.
(c) OK but there was some confusion over responsiveness (which is what multi-threading can provide) and execution speed.
(d) Generally well done although some produced confused answers talking about "files" being stored in the linked-list.