<table>
<thead>
<tr>
<th>Course</th>
<th>Module</th>
<th>Tutors</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMP 22712</td>
<td>Microcontrollers</td>
<td>Paul Nutter, Jim Garside</td>
</tr>
</tbody>
</table>

**Comments:**
Lab only: feedback not provided.
Comments: Please see attached PDF.
Q1.

The was done extremely well. Many students lost marks because they could not clearly formulate, e.g. negation-as-failure, even though they knew in practical terms what it meant: in such cases, I felt I could not turn a blind eye, particularly with such an easy question. Some students lots marks on the unification question because they did not write out the full effect of unification: e.g. writing \( x \leftarrow a \) and \( y \leftarrow f(x) \). If this happened just once, I let the student off, but not if done systematically. (After all, some students gave this type of `solution' for part (ii), where there is no unifier!) I was amazed my how many candidates got parts g) and h) correct.

Q2.

This was an unmitigated disaster. About half the class couldn't state the resolution and factoring rules, which meant they stood little chance with the rest of the question. Absolutely no one got a correct proof in part e). When marking that part, I gave marks whenever I saw what looked like an approximately reasonable proof step; but most candidates had no idea. Parts c) and d) were generally done okay: I took a mark off in each case if a candidate mixed up 'and' and 'or' or if he somehow mixed up Skolemization. (I had no idea until now in just how many ways this could be done.)

In previous years, while candidates have found questions on resolution moderately difficult, there was nothing like the wipe-out we saw this year. The only two differences in presentation were: (i) the material was presented earlier in the course this year (because there were two lecturers); (ii) there was less time for revision lectures. Whatever te reason, this kind of systematic collapse in understanding is quite worrying.

Q3 and Q4 (Allan Ramsay): I did not make any changes to the marks scheme for either of my questions--there were no patterns of answering based on consistent unexpected interpretations of the questions, and nothing to suggest that I had asked a question about something which I had either omitted or taught exceptionally poorly and which therefore required compensation in the mark scheme.

Q3 was done by virtually everyone who sat the exam. The mark distribution was completely trimodal: (a) you'd been to the lectures and revised the notes, and you got between 15 and 20; (b) you'd been to the lectures and revised the notes, but had missed out the bit on shift-reduce parsing, and you got around 10; (c) you were completely unprepared for the exam and got between 0 and 5. I was a bit concerned that I had underemphasised shift-reduce parsing when I was giving the course, particularly because I didn't include it in the coursework, but enough people did do a good job on this part of the question for me to be happy about this.

Q4 was not done by many students, and was not done well by any of them. I don't think I taught this part of the course particularly badly, but students do tend to find this material hard. I suspect that this question was attempted by students who did not fancy Ian's theorem proving question, and I think the marks may have suffered from it being the last thing they attempted and hence there not being enough time to do it--there was at least one where the answer was along the right lines but the student just ran out of time.
Q1 Caches

Most students answered this question reasonably well, but a disappointingly large group failed to understand how cache consistency could be a problem in a system with peripherals accessed by DMA. The students performed well on the problem parts of the question, (e) particularly since we had covered a similar problem in lectures, and mostly adequately on part (f) which was new to them.

Q2 Storage

Answered well in general, although too many students could not distinguish between seek and search times for disks. Numerical parts (b), (c) and (d) were answered better than in previous years (although a worrying number of students, i.e. more than zero, believed that reading 4 kilobytes from a disk might take days).

Part e was answered well by only a handful of students. The fact that regenerating a RAID array after a disk has failed might take months if the regeneration has to compete with high demands from random traffic will be stressed more in lectures in the future.

Q3

Most students answered this question reasonably. The biggest points for confusion where:
* The difference between writing to the main memory and writing to the register file in the pipeline.
* The need to insert multiple NOP's into the instruction stream in some circumstances to overcome hazards.
* and that reordering of instructions is not only used for ILP, but can also reduce the presence of hazards.

With future questions of this style how hazards, forwarding and reordering is to be demonstrated with the stream of instructions should be clearer as many students simply drew a single example from the stream rather that rewriting it to include the required NOPs.

Q4

Most students covered all the book work on multi-threading correctly, however a disturbing number failed to included any reference to multiple threads and switching for the final two parts of the exercise. Instead they focused simply on the time taken to access the memory and missed that this could be hidden by changing thread. This is possibly down to Alasdair doing more of this kind of calculate in his lectures on caches than I did when talking about multiple threads. One problem that this showed with the question was that because of the high hit rate in part f, both with and without switching the processor approximated 100% throughput. If a similar question is used again a large differential between these two numbers should be ensured.
Q1 (Knapsack)

The question was answered with widely varying degree of success. Many did very well, scoring 18 out of 20, with a few achieving 20. More students achieved greater than 10 than less than 10, and the very low marks seem to be students who ran out of time, and did not attempt parts of the question.

Part (a) describing the 0/1 Knapsack problem was answered well, although a few seemed to get confuse 0/1 Knapsack with the change-making problem.

(b) on greedy was answered well by the majority. A few were again confused about the proper definition of the 0/1 problem, or how greedy operates.

(c) on the dynamic programming approach was answered satisfactorily, but very few got 6/6. Mostly, a simple example was given but not a correct general algorithm; this tended to achieve 3/6 or 4/6 depending on detail level.

(d) On branch and bound. There was a mistake in this part of the exam paper, which the students were informed about, but only after 55 mins of the 2 hour exam.

If the student did not correct the question, then their answer should have been the same as in the original marking scheme. The priority queue updates, the final answer (395), and the termination condition could all be calculated, and it was possible to obtain full marks (many students did).

If the student did correct the question, then potentially the answer had two additional steps to do. (This was because two different solutions in the priority queue had the same value and if one explored the "wrong" one first, then a dead-end is reached in two steps, which then reverts you to the correct branch of the search tree.). Again, full marks were possible and many people got these.

Given this, no student should have lost marks for either correcting or leaving the question as it appeared.

Q2

This question was concerned with graph traversal algorithms and it was linked to an examinable lab from the course. The students mainly did the first part of the question (writing a pseudo-code or explaining Dijkstra's algorithm), but this is a bookwork. The second part was more creative, as it required the application of Dijkstra's algorithm to a problem of constrained pathfinding in a graph. Thus some heuristic was required. Quite a few students did this right, and the mistakes were limited to the step-by-step application of Dijkstra's algorithm (although there weren't many of them). The third part (regarding the algorithmic complexity of graph pathfinding algorithms) seems to be the most tricky one and there was a substantial amount of poor answers (mainly the students can't explain their answer even when they have written (guessed) a correct value). The last part (finding the longest paths in an acyclic graph) was a tricky question, although the hint given was quite compelling, and the students did reasonably well. Overall, there was a substantial number of scripts with good and very good answers and the performance was satisfactory.

Q3

This question was concerned with tree algorithms. For a given sequence of keys the students were asked in the first instance to create a binary search tree. This part of the task was done correctly by most of the students. In the second part they were asked to do the same thing but for an AVL tree. The students were struggling with this part mainly because they didn't know how the AVL trees function and when
and how to perform the balancing operations, but there were some remarkably good answers. In the third part of the question they were asked to write a pseudo-code of a findElement method which finds an element with a given key in a binary search/AVL tree. Most of the students attempted this part but with varying success. The main issue was that they either did not give enough details in their pseudo-code, or they didn't appreciate a repetitive character of the operation (which required a recursion). Exception handling was also an issue here. In the final part of the question the students needed to give the asymptotic complexity (in the worst case) of the algorithm from the previous part for binary search/AVL trees. They mostly struggled with the former case (not appreciating that in some cases a binary search tree may degenerate into a list), but there were some very good answers. Overall, performance was good, but there were some disappointing answers.

Q4: Graphs and Graph Algorithms

Though some students got good to very good marks on this question (2.1 and 1st class), in general it was poorly answered with many sketchy and incomplete answers and corresponding low marks. Most students understood the graph representations, but often they were not explained well. The part asking for graph computations using the representations, where full answers were given, was well answered with many simple algorithms proposed. DFS, even for those who knew the material, was not in general well answered - with marks lost for giving tree traversals, rather than graph traversals, and for partial code. The final part on topological sorting: many saw the connection with cycles in graphs and also a DFS-based construction, but in both cases, lack of precision in the explanations, and incompleteness in the construction, lost marks.
Q1: Most students did this question, with many full answers and high marks - some gave inefficient algorithms, and others concentrated on efficient methods, but most gave full algorithms and derivations of complexity.

Q2 on complexity was generally answered acceptably (marks ranging from 7/20 to 17/20) with a mean around 11. Students struggled more with dealing with functions of two variables (n and m) and with reasoning about big omega, but this is as expected.

Q3 The question was concerned with heaps. The students needed to define the heap order property of a binary tree. Most of the students who attempted this question did this right, but this is a bookwork. In the second part they were asked to show the process (step-by-step) of removing the minimal element from the heap. This is also relatively straightforward and most of them did it right. In the third part the students needed to write and explain the asymptotic complexity of some elementary operations related to heaps. There were some mistakes in this part. In the final part of the questions the students needed to write a pseudo-code of an algorithm for finding all the keys in a heap that are smaller than the prescribed value. The trick was to apply a tree traversal algorithm and notice the heap order property from part 1 of the question which simplifies and accelerates the algorithm substantially. There were some mistakes here (most notably in assessing the asymptotic complexity of this algorithm), although some students did remarkably well.

Q4: Answers were, on the whole poor, with very sketchy and incomplete answers, showing, in general, that little real revision had taken place - from the lectures, the revision guide, website and course textbook. Marks were correspondingly low.
General comments on Questions 1 and 2: Many students did not draw the diagrams they were clearly asked to provide, and therefore lost the marks available for those diagrams.

Diagrams were often far too small, scrappily-drawn, and not properly labelled.

In two cases handwriting was so bad as to be almost illegible. In many more cases the handwriting was poor, which made it extremely difficult to understand what the answer was meant to be.

Section B - Question 1

Average 8.1/20 (41%), with STDDEV of 4.1.

This question was generally very poorly answered. The areas it was addressing were covered in the lectures, and put into practice in the labs, so it was very surprising that the level of understanding shown was so low. Specifically:

* inability to distinguish between "modelling" and "rendering"
* poor understanding of why 4x4 matrices are used for 3D transformations
* lack of understanding of the OpenGL matrix stack, despite it being used extensively in the lab.

The part of the question that was best answered was the final section -- rotation about an arbitrary 3D vector. However, in many cases the answer was written as a series of points, with no evidence of true understanding.

Section B - Question 2

Average 13/20 (65%) with STDDEV of 4.4.

Overall, quite well-answered. The section on ambient/diffuse/specular reflection was largely answered very tersely, with students just quoting the formulae with little or no narrative explanation -- or, indeed, any definition of the concepts of ambient/diffuse/specular reflection. Several students ignored the part-structure of the question, and instead gave a complete derivation of the local illumination model, with no reference to what was actually asked in the question. Only about half the class understood the Gouraud technique, which was disappointing.

Question C1
Mean = 53.6 ± 22.9%, 137 answers
A satisfactory mean but a high standard deviation due to the distribution of marks being skewed towards the low end.
Parts a was repeating a definition and was generally correct.
Part b: students either failed to give three methods or failed to give adequate definitions.
Part c was answered incorrectly by about half of the students, disappointing as it was explicitly mentioned in the handouts.
Part d required some thought and application of methods that were taught in lectures or labs, a similar application had been in the news during the course and was discussed in a lecture. Those that had understood the course and the methods discussed in lectures scored well. Otherwise, results were poor. One candidate simple repeated information in one of the laboratory handouts – good memory but poor understanding.

Question C2
Mean = 62.1 ± 20.6%
28 students answered this question, fewer than answered C1, possibly because it required more thought to suggest a solution to part c. The mark distribution suggested two types of answer: those that were successful in part c and those that were unprepared for the exam.
Parts a and b required a repeat of a definition and examples given in the course.
Part c required some thought to invent a solution, any sensible solution such as recognising the outline of a vehicle or the border of a number plate was given credit.
Q1. Compulsory question, all students answered it. No particular problems, overall a high average, quite good performance, a pleasure to mark. Small issues here and there but nothing particularly noticeable.

Q2. Popular question, over 100 students answered it. Key problems: In (a) several students failed to realize that "latency nearly zero" could still be a valid assumption. It is the interplay between high and low latencies scenarios that may create problems (and requires different solutions). In (c) several students failed to noticed that the Byzantine Generals problem is equivalent to the situation where many components need to come up with a common answer, but one behaves arbitrarily. In (d) some students connected the data consistency model with the order that processes are executed. Of course, concurrent processes can be executed in any order.

Q3. Again a popular question (over 100 students chose it). Some common issues: In (a), client and server stubs can do both marshalling and unmarshalling. In (b) the question asked for 'distributed transactions'. Issues in (c) and (d) related to the calculations.

Q4. Only 37 students answered it. Overall good answers. In (c) some students failed to realize that the server can crash after any statement and the issue is that if the transaction has not committed everything is rolled back. In (d) some students didn't realize that there are plenty of resources to cope with demand, and it is unlikely that messages will wait in a queue (subject to spikes, etc). In (e), some students suggested costly strategies (for example, start new threads), which will harm overall performance.
Question 1:
Performance on these compulsory short questions was generally good. There were the usual issues with arithmetic on parts a) and c), and a slightly worrying new trend where a few students didn't know their mW from their MW or their uW, giving a spread of 12 orders of magnitude in the result! These short questions are a straightforward test of your revision, though they are on points that were flagged during the lectures as being particularly important to someone who professes competence in mobile systems. There are no tricks or twists. If you fail to get full marks you have simply failed to revise the whole course or pick up the points that were emphasised as being vital. Overall, the average for Question 1 was pretty good, but it was not close to 100% as we hoped.

Question 2
Not too many attempted this question and for those that did the average was very low. The calculations on 2(a) are based directly on the Revisions Class, and the concepts of speech and formants are as presented and explained in Lecture 3. These must have passed most people by.
Part (c) (i) is based both on laboratory work and the material in Lecture 6.
Part (c) (ii) was covered in Lecture 6 but very few displayed any knowledge about it.

Question 3
Most people answered this question which is based on Lecture 4 and the Revision Class. The average was not too good largely because of a number of very low marks (close to zero) and the poorer than expected responses to part (a). Disregarding the six lowest (very low) marks (out of 46) produces a reasonably good average. Part (a) on the theory of JPEG was not universally answered well, though there were some pretty good answers. Part (b) was widely understood, though the 'self terminating' aspect of Huffman codes is an important feature that should be mentioned for full marks. Part (c) was almost universally answered well and very often perfectly. There are several optimal solutions. Some exam answers showed good understanding of the method, but produced workable but non-optimal solutions due to non-optimal reordering of the probabilities.
Answers to part (d) required some insight into the nature of compression in general (most people answered this correctly) and the self-terminating nature of Huffman codes (about half understood the implication of this to the effect of bit-errors).

Question 4:
The majority of students attempted this question, with most achieving over 50%. A significant number of answers displayed knowledge of the role of high and low water marks, but did not explain the constraint that they must be set with enough margin to allow for end-to-end network latency - the buffer keeps filling after "pause" is sent until the sender receives the message and all the packets already in transit have arrived, and similarly for "resume". The final part of the question, about coping with wireless network variability, was tricky and few students got full marks here. The key was feedback and adaptation, of transmit power or some other parameter.