UG Exam Performance Feedback
First Year - Semester 1

Ian Watson:
Q1a) Those who knew the answer performed well. The most common mistake was to answer that a pseudo instruction was the textual form rather than binary.
Q1b) Many answered well. The most common mistake was to give arm code for the expression evaluation, not zero address code.
Q1c) Many good answers. Several strange answers about deleting incorrect code!

Q2 a&b) Many correct answers about indirect and PC relative addressing. Usually answered well.
Q2c) A lot of correct answers but few used the obvious diagram to save a lot of words
Q2d) The question expected a little bit more than just the raw instructions with no explanation. Say what a full descending stack means.
Q2e&f) This seemed to be questions where either the answer were either near enough correct or completely wrong.

Pete Jinks:
General comments:
- Start each new (20-mark) question at the top of a page. Leave a few blank lines between the parts of one question.
- Don't use red ink.
- Read the whole question, before you answer it, and again afterwards, to make sure that you have done everything it asks for.
- Remember that an answer to a part of a question worth e.g. 5 marks will be equivalent to about 10 minutes of exam time, and you might be expected to e.g. clearly state or explain 5 (or even 10) different things, so don't assume that you have done enough after writing just a couple of sentences. Equally, don't spend lots of time and space on something that is only worth a few marks.

COMP10031:
Q1
d) Hex 12A is NOT decimal 12*16 + 10; 16*16 is NOT 960. The sensible way to answer is to convert the hex directly to binary, NOT hex to decimal and then decimal to binary. Give some explanation (a few lines should be enough, or even just show your workings); DON'T just give the numerical answers.
e,f,g) I don't want to see e.g. "an assembler assembles code but a compiler compiles programs" - this gets no marks. I want to see a clear explanation so I can be sure that you understand.

Q3
a) The question is very explicit about what it wants, and is worded similarly each year, and most people still lose marks because they don't answer every part of the question! Talk about ALL of:
- instruction fetch and the role of the PC register what the operations do, including fetching and storing operands
- the resulting (new) values in registers and memory locations fetches and stores mean that information is moving between the memory and the CPU you can also mention the effects of caches if you want to impress me, but don't waste much time doing so - no extra marks)
STR is not the same as LDR! Every year some students confuse the two, or at least fail to explain the difference clearly.
b) I just want the code; you don't need to explain how it works unless I actually ask for that
(e.g. in part c).
- get the difference between LDR and MOV right
- if I ask for optimised code, don't do e.g.: MOV R1,#2; CMP R0,R1 do CMP R0,#2
- optimising a loop: LDRs before it starts, STRs after it ends
- if I say something like "this is just part of a larger program" then remember to do STRs
  and not an SWI #2 at the end

c) I asked for examples of translating Java to ARM, so give examples for each of the 5 steps
  - of compiling Java, NOT of assembling ARM
  - of compiling to ARM, NOT to byte code I don't want e.g. "code generation is when the
    code is generated" see Q1efg above)
General

The marks were widely distributed across almost the whole possible range. Some candidates clearly had learnt almost nothing whilst others probably found the exam rather easy.

The questions were set with the greater ‘problem-solving’ elements in the longer question (2 or 3) with Q1 being more ‘straightforward’. This is reflected in the results where it is clear that candidates either can or cannot tackle the more serious problems.

Q1 Overall this question elicited slightly higher than typical marks with ~1/3 of candidates achieving marks in the first-class bracket.

a) Mixed range of answers, mostly done quite well; several people confused XOR with XNOR. Some attempts where even simple ‘sanity checking’ would fail, however.

b) Generally well done. Syntax was rarely perfect but that was not penalised. Most common problem amongst students who understood what to do was missing the sensitivity list, or elements thereof. Several different forms of implementation were proposed: anything which works is fine.

c) In general this was disappointing. Many candidates could spot the difference in the leading edge of the data change but a high proportion thought that counting would still continue despite the active reset. Other ‘interesting’ behaviour included the data state being zeroed and then reappearing when reset was inactivated as if nothing had happened! A few did not understand what was meant by ‘reset’ - despite several examples in lectures and labs.

d) Basically a ‘giveaway’. Most could identify two (different) tools although a few do not understand what a ‘tool’ is - as opposed to, for example, a component.

e) Well answered probably should have been marked a little more strictly as most answers were terse. Some confusion between flip-flops and latches.

f) Disappointingly very few people picked up on the choice of an edge-triggered flip-flop here which ‘instantaneously’ snapshots its input so that many devices can communicate synchronously. This was emphasised when the model was introduced.

g) Many students who answered this just latched onto the words dynamic and static and said that dynamic RAM allowed instructions to be stored anywhere but static RAM only allowed instructions to be stored in fixed locations. It was clear they had not attended the lecture or read the notes. The ones that understood gained both marks for this easily.

h) This was generally answered well, apart from a few students who decided tri-state devices were something to do with FETCH, DECODE and EXECUTE!

i) This was also answered well by students who attempted it. Most gained the full two marks.

j) Nearly all students who attempted this knew the function of a programme counter and so gained one mark. Several were unsure of why it needed to be inc. or dec. by more than one. Several answered this in terms of word/byte addressing rather than talking about jumps, but I gave half a mark for this.

k) This was also well answered and most could give two reasons for designing complete systems on a single chip.

l) Several students demonstrated that they did not understand the concept of an address bus. They calculated the number of pixels correctly, but could not translate this to a number of address lines. The need for 8 bit data to translate into the 256 grey levels also proved too much for some to understand.

Q2 About 75% of the candidates opted for this question.

a) This part was answered well, but was very straightforward.
b) In this part there were several candidates who could not distinguish between asynchronous serial comms and asynchronous systems. Most new that synchronous involved a clock, but several forgot to mention it was a common clock for receiver and transmitter.

c) This was answered well on the whole, but several candidates confused start and stop bits with two wire handshaking.
d) The calculation of bandwidth and latency in this part was generally done well, but many did not notice that 11 bits were transmitted for one data byte.

Q3 Not a favourite question - attempted by about 25% of candidates. The question may look intimidating but, taken a step at a time, should not present too many problems. Marks distribution was uneven: nearly half these candidates achieved very good marks with another quarter earning almost nothing.

a) Easy syntax interpretation. Candidates should be warned to be specific: for example "the last bit" is ambiguous.
b) There is a big clue in the (source) line above! Most spotted this. Multiplication ("2*y") is not a sensible option as multiplication ISaddition. One looks to minimise hardware and a shift is effectively free.
c) There is one state bit which should be a clue to the number of states. Some tried very hard to over-complicate this. Otherwise not badly done although the labelling of transitions was rarely completely rigorous.
d) Either well set out or not done at all. Those who followed these steps through should have no problem with part e)
e) See above

f) Some reasonable descriptions although none related to the machine already studied. Handshaking is being used with 'start', etc. where the timing is 'asynchronous' as evaluation takes an externally undetermined number of clocks. Typical 'long' questions are usually based around a theme.

g) Completing the signals needed for communication. Those who reached this point produced some imaginative (if sometimes sub-optimal)answers for which credit was given.