COMP12111 Fundamentals of Computer Engineering

Exam Feedback 2013/14

Statistics

Total number of students who sat the exam – 180.

Q1 – All students (compulsory), average mark 74%
Q2 – All students (compulsory), average mark 43%
Q3 – 95% of students attempted this question, average mark 55%
Q4 – 5% of students attempted this question, average mark 38%

Average for the exam paper was 56%. However, the high lab average will lift the mark of most students.

Feedback

On a general note, some of the answers provided were very untidy: scruffy diagrams and incoherent messy handwriting (we appreciate that some students are DSO registered). It is essential that you make every effort to present your answers neatly. If we cannot read your answers, then we can't tell if you have answered the question correctly, so you will lose marks. Make use of the space available in the answer booklets – you can use as many booklets as you need!

Q1 – PWN

A compulsory question marked out of 10. Answer 5 from 8. In the cases where more than 5 questions were answered the best five marks were taken.

a) A popular question.

1 mark was awarded for using DeMorgan’s theorem to recast the expression in the form of NAND functions – ½ mark was awarded for each of the two steps in the process. 1 mark was awarded for producing a correct schematic of the function using 3 NAND gates (½ mark for the gates connected correctly, ½ for labeling all the inputs and outputs).

Problems:
- Not showing all the steps when recasting the expression.
- Producing a function that does not include just NAND (.) gates (in a number of cases it was expressed using NOR gates (+)).
- Not using NAND gate symbols in the schematic (EXOR were used on few occasions).
- Not labeling inputs and outputs on the schematic.
b) A popular question.
   Marks were awarded for the waveforms as follows: ½ mark for labeling the
   waveforms, ½ mark for getting the waveforms correct, ½ mark for including
   a time axis, and ½ mark for recognizing that the simulation stops at 120ns.

   Problems:
   - Not having a time axis.
   - Not showing the simulation stopping at 120ns (or one stopping
     earlier).
   - Not labeling the waveforms.
   - Producing hastily drawn, scruffy diagrams.

c) A popular question.
   1 mark was awarded for recognizing that a D-type flip-flop is the basic
   component of a register (flip-flop was accepted). 1 mark was awarded for
   discussing that the CE, or clock enable, input is used to control which clock
   edge data is loaded (active when high, inactive when low).

   Problems:
   - A latch is not the basic building component – registers use flip-flops.
   - Not recognizing the role of CE (it does not reset the register!)

d) A very popular question.
   There were 4 errors in the state transition diagram, each worth ½ mark:
   - X being used as an input and an output – it can’t be both.
   - No defined path out of State_3 – which does it take?
   - Incorrect transitions out of State_4 – what if A=B=1?
   - State_2 and State_4 have the same state code – they should be unique.

   Problems (lots of different problems here):
   - Answers stating that the assignment of state vectors in wrong, and
     that they should be in numerical order: State_2 should be 01, State_3
     should be 10 etc. – this is not true. State vectors can be assigned in any
     arbitrary manner, as long as they are unique between states.
   - Producing a “corrected” state transition diagram – you were not asked
     to do this, but to identify the errors. You cannot “correct” the state
     transition diagram without knowing the specification.
   - Not recognizing that X is shown as both an input and an output – a
     number of answers stated that the X in State_2 should be on the
     transition – how would you know this?
   - Stating that there is no condition for the transition out of State_2 –
     there doesn’t have to be, it will move out of State_2 at the next clock
     pulse after entering it. You do not need inputs variables controlling
     the movement out of a state. In cases such as this there will only be
     one possible path out of the state.
   - Only listing 2 errors (2 marks were available, but this doesn’t mean
     there were only 2 answers!).
- stating that state codes should always be in the circle – not true, state
codes are often outside. The circle usually contains the state name and
any outputs that are asserted in that state.

e) Not many answered this question.
Two examples were required, ½ mark for each, ½ mark for the discussion of
each. Example include: schematic entry, waveform viewer, and the simulator.

Problems:
- the main issues related to not presenting a CAD tool (abstraction and
  hierarchy are methodologies).
- little discussion of the tool.

f) A very popular question.
½ a mark was awarded for each correct answer.

Problems:
- Not representing a negative number correctly using two’s
  complement (invert bits and 1).
- Not recognizing that +36 cannot be represented using only 6-bits as it
  is outside the range -32 to 31.
- Giving an answer that was 7-bits – the question specifically asks for a
  6-bit value.

g) A number of students attempted this one.
Setup time – time before a rising clock edge when the data must be stable – ½
mark. Hold time – time after a rising clock edge when the data must be stable
– ½ mark. Consequences – in the case of a flip-flop if the data isn’t stable
during these times then it may not be latched properly – 1 mark.

Problems:
- Not clearly stating before and after clock edge for the two times.
- Not clearly discussing the consequences.
- Not mentioning clock.

h) A few attempted this question.
Marks were awarded for a correct schematic showing the generation of
a variable and c’ - ½ mark, labeling a variable on the schematic – ½ mark,
showing the generation of q – ½ mark, and labeling all inputs and outputs –
½ mark.

Problems:
- No real problems. Marks were generally lost for not labeling nets.
Q2 – VP

A compulsory question marked out of 10. Answer 5 from 8. Not everyone attempted this question.

a) A popular question.  
1 mark was awarded for describing the basic characteristics (difference) of the two architectures. Another mark was awarded for describing the various advantages and disadvantages. Some students also provided a drawing of the architectures which was informative and nice (wasn’t required though)

Problems:
- Failing to state the basic difference/characteristic of the two architectures
- Many answers were verbose but irrelevant.
- Some answers just tried for one mark, not discussing at all the pros and cons of the two architectures

b) A popular question.
1 mark was given for correct answer on volatility of all types of memories and ½ marks for the correct order in terms of speed and another ½ mark for correct sorting in terms of storage density.

Problems:
- Many students seemed to not have understood the word “volatility” and provided reversed answers.
- Many answers sorted the memory types but did not indicate a descending or ascending order. We assumed that as long as the order was correct, the mark was given.
- Many answers missed the storage density part.

c) A popular question too and supposedly easy to try; yet very few managed to get full marks on this one.  
½ marks were given for each correct answer and ½ marks were taken off for any incorrect answer. The negative marks were there to discourage random answers.

Problems:
- Most of the answers failed to capture the clock period right. Great problems with the units.

d) A relatively popular question.
1 mark was given for describing the TDM and another mark for giving an example of its usage. Many answers were correct and some offered a diagram similar to the one on the slide, which was nice.

Problems:
- TDM was not spelled out or it was spelled out incorrectly. Some confused TDM with DMA.
- Also some missed the expected example

e) A quite popular question.
1 mark was awarded for determining the number of chips and another mark for determining the size in BITS of each memory chip.

Problems:
- There was a great misunderstanding about how memory chips are addressed in many answers.
- Many errors had to do simply with multiplications and power of two!
- Many answers did not provide the size in bits (as it was explicitly asked) but gave some random numbers in words or without units.
- Some provided only numbers with no derivation. No marks were given to those even if the numbers were correct! We are interested in seeing your understanding not just dry results.
- We had worked on many examples in the class and better scores on this one were expected.

f) A quite popular question.
1 mark was given for the correct assignment and 1 mark for describing the principles used in making this decision.

Problems:
- Many answers got this just right and gave nice descriptions.
- Some answers did not relate correctly the size of the cache and the size of the memory parts given. This could have been determined by looking at the addresses of each segment and the given cache size. From there it could have been determined how many parts fit into the memory. Then the appropriate parts should have been selected.
- Many answers confused memory size with memory utilization when making this selection.

g) A not so popular question.
2 marks were given for the correct derivation and result.

Problems:
- Some answers just provided an integer number, which was not given any marks even if the number was correct. We wanted to see how you thought about this problem and how you derived the results, not just a guess, even if it was an educated one.
- An example in the class was provided along with an analytic derivation which was straightforward. Some answers were a bit different but still valid and received full marks.
- Some answers did not include the CPU time and although the result is invariable (it just happened to be the case), ½ marks were taken off.
h) A popular question.
½ marks were awarded for describing the PC role and 1 mark for describing how it is updated (two cases here for ½ mark each). Another ½ mark was given for providing some examples of its applicability.

Problems:
- Many answers claimed that a program counter stores program data!
- Many answers forgot the branching case.
- Some answers only described its role and not how it is updated during program execution.
Q3 – PWN

A very popular question, with almost everyone attempting it. Marked out of 20.

a) The aim of the question was to discuss the general structure of a Verilog module. 3 marks in total awarded for mentioning:

- begins with keyword “module” – ½ mark.
- ends with keyword “endmodule” – ½ mark.
- the header contains the name of the module (after keyword “module”) – ½ mark.
- followed by the input/output declarations – ½ mark.
- any internal variables are defined – ½ mark.
- variables are assigned values using assigns, or within always blocks – ½ mark.

1 mark is awarded for stating that the module is used in a schematic using a symbol.

Problems:
- Not discussing the general structure, but incorrectly discussing the example module given.
- Drawing a finite state machine (why?).
- A lot of answers failed to answer the last part of the question.
- Not stating that the module is instantiated using a symbol in a schematics (mention was made of logic gates, fsms etc).

b) The * in the sensitivity list (½ mark) of the always block means that if any of the input signals in the always block change value, then the always block will be executed (½ mark).

Problems:
- Not explicitly mentioning sensitivity list (in relation to *).
- Incorrectly stating that the always block “always” runs.
- Incorrectly stating that ALL input signals changing causes the always block to execute.
- Incorrectly stating that all inputs and outputs changing causes the always block to execute.
- General lack of clarity in the answer.

c) The default case is in place to catch state codes that are not listed in the case list - ½ mark, as well as any states that are undefined (have an X in them) - ½ mark.

Problems:
- Not mentioning BOTH cases.
- Going into detail about setting the value of the state on startup – this a consequence of the default since initially the state is undefined.
- General lack of clarity in the answer.
d) An easy question. The always block executes on the rising edge 0 -> 1 transition of the clock.

Problems:
- Very few. The main problem was incorrectly stating that it executes when clock is high.
- General lack of clarity in the answer.

e) A sketch of the state transition diagram was required. 1 mark for a correct 000 state, with exit from the state when start is high, staying in the state when start is low. 1 mark for highlighting the reset condition. ½ per remaining state for getting the state correct (3 marks in total).

Problems (a large variety):
- Not getting the state 000 correct. Some answers incorporated reset in this to form a Boolean expression with start and reset (which is fine), but in these cases I checked that all Boolean conditions were satisfied, which in a lot of cases they weren't.
- Two transitions from state 110 to 000, one with reset. There is only one transition regardless of the value of reset, it always moves to 000.
- Incorrectly stating “posedge” in the transition between states – there is no control signal called “posedge”.
- Incorrectly stating “clock” in the transition between states – clock is implicit in the state transition diagram, it is not a control signal. A transition from one state to another (or the same) always happens at a clock edge; the inputs determine which transition is taken.
- Having a not(clock) path back into every state – again, clock is not a control signal.
- Not showing arrow heads on lines – which way is the transition?
- No showing the reset transitions.
- A number of answers provided a state transition table, and not the required state transition diagram.

f) A Verilog always block that sets the two output signals was required. The best solution is to use a case statement with current_state as the state variable. Marks were awarded as follows:

- always @ (current_state) – 1 mark.
- case(current_state) – 1 mark. If nested if ... else statements were provided that marks were awarded appropriately.
- Specifying binary values correctly, i.e. 3'b000, or 5'b00110 – 1 mark.
- Correctly assigning sequence values for all cases – 1 mark
- Correctly assigning flash values for all cases – 1 mark
- Having a default statement (for a case example) – 1 mark
- Having a syntactically correct implementation – 1 mark
- Providing comments – 1 mark.
Problems:
- Not setting the state of “flash” in all cases. It should be set to ‘0’ in all but states 101 and 110.
- Missing default.
- Using sequence and flash as inputs (they are outputs).
- posedge clock in the always block sensitivity list – you are creating a block of combinatorial logic, so the always block “executes” when the relevant input signals change.
- Very few commented their code.
- Missing “begin” and “end” when there are multiple assignments - syntactically incorrect.
- In some cases individual bits of the output sequence were set, the value of sequence should be set to a 5-bit value in all cases.
Q4 – VP

Very few answered this question and I thank you for that! Marked out of 20.

Although few students tried this question, I believe it was not difficult but it required to have digested the taught material throughout the term. Let’s look at each part.

a) 1 mark was given for the correct timing diagram, where it was provided on the slide and another 2 marks for a short(!) discussion on what happens if the timing is not appropriate.

b) Most of the students tried this one as a similar example was presented in class. 2 marks were given for the critical path, 2 marks for the correct path delay and 1 mark for the correct frequency.

Problems:
- Most answers got the critical path correctly but some added the delay wrong. There were mistakes in addition!
- To determine the clock frequency the set-up time of the latching FF or register should be added. Two set-up times were added resulting in the wrong clock frequency.

c) 2 marks were given for each way to add/subtract numbers. Most of people got this right as it was discussed in the labs. Although the offered schematics were not correct to every detail, they demonstrated the basic operation. Some text to support the diagram was also expected and that was mostly given.

d) 1 mark was given for the correct diagram and 2 marks for the revised diagram with bit stuffing and an explanation of bit stuffing. This was very similar to the examples we worked in class, just a different bit sequence. Some answers got this right, some answers showed no understanding of the basic principle of data encoding for the USB protocol. A short description of the significance of the bit stuffing (2-3 sentences) was also expected.

e) 3 marks were given for the correct truth table and 2 marks for the Boolean expression and the schematic (one mark for each). The basic requirement in this question was to recognize that many “don’t care” existed and to use this to simplify the truth table and, therefore, the required Boolean expression. The requested circuit implementing the INT output was ONLY a 4-input logic gate.