Work for Digital Electronics Supervision I

The questions below are partitioned into two sets: the *core* questions and the *extension* questions. The core questions are based on material that the course should have covered by the end of lecture 3, and you should be familiar with. Nevertheless, they will likely demand a bit of thought.

The extension questions are based on the same material, but will take longer, and require a lot more thought. Attempting them is optional. The more of the extension questions that you try, the better you will become at Digital Electronics and the easier you will find the work in future weeks. However, do not sacrifice your progress in other subjects or your health for the sake of completing them.

Please attempt all five core items and as many of the extension questions as you can, and submit your work to me as instructed by 1500 on Friday, 14th October. Please include all your answers, working and any additional thoughts in your submission.

Core questions

1. Examples sheet questions 1, 2 [AM], 11 [AM], 7 and 8.

   For 1, also state the name of the operation implemented by each circuit.

   For 2, also show that $x + \left(\overline{x}.y\right) = x + y$. Furthermore, show (using Boolean algebra or otherwise) that $\overline{x+y} = \overline{x}.\overline{y}$ and $\overline{x.y} = \overline{x} + \overline{y}$. Don't do $a.b + \overline{a}.c = (a+c).(\overline{a}+b)$; it's needed for core question 4 below.

2. The tiles in the floor of the "street" (public walk way) in the William Gates Building show the pattern of a punched paper tape along its length. Decode the message.

3. (a) How many dyadic Boolean functions are there? Why?

   (b) Show that NOR is a complete base, all by itself (HINT*: just show that* AND *and* NOT *can be implemented using* NOR).

   (c) Is $\{., \equiv, \oplus\}$ a complete base? Why (not)? $\equiv$ is the equivalence operator $x.y + \overline{x}.\overline{y}$.

4. In this question, whenever a definition is required, start by explaining the term meticuously and unambiguously using precise scientific language. Reinforce your prose with algebraic expressions where necessary. In general, a shorter definition will be clearer than a longer one; furthermore, constructing a more concise definition will force you to think harder about what you are trying to say. Be careful not to neglect any edge cases (e.g. *don't care* terms), and try to make your definitions as general as possible (i.e. encompass the n-variable case, not just the 2-variable case). Finally, where appropriate, conclude each definition with an example relating what you have written to a concrete instance on a Karnaugh map.

   It is permissible to refer to the lecture notes if you need to, but try not to answer the question with them open in front of you. It's preferable to write down an answer and then use the notes to correct it, rather than to refer to the notes before your first attempt.

   (a) What is De Morgan's theorem?

   (b) What are *minterms*, *essential terms* and *prime implicants*? What does it mean for a term to *cover* another? What is the *covering set*?

   (c) Define *sum-of-products form*, *minimised sum-of-products form* and *disjunctive normal form*.

   (d) [AM] What are the prime implicants, essential terms and minimum sum-of-products for $f = (a.b) \oplus (c+d)$?

(e) If we do not care about the output of f whenever $g = a.b$ is true, determine the new minimum sum of products form for f.

(f) Using a four-variable Karnaugh map, fill it with 1s and 0s to find a function for which the minimised POS form has fewer terms and literals than the minimised SOP form. Write down the number of terms and literals present in each form.

5. (a) Show that $(X+Y).(X+Z) = X+Y.Z$

(b) Show that $(X+Y).\left(\overline{X}+Z\right) = X.Z+\overline{X}.Y$

(c) [AM] Which common functional unit does the Boolean expression in part (b) describe?

(d) Using the results in parts (a) and (b), or otherwise, simplify

$$P = \left(A+B+\overline{C}\right).\left(A+B+D\right).\left(A+B+E\right).\left(A+\overline{D}+E\right).\left(\overline{A}+C\right)$$

(e) How does your solution to (d) compare with the result obtained from a 5-variable Karnaugh map? (HINT: you may have to introduce a new kind of cell adjacency in order to be able to represent 5 variables on the map. You may find your answer to part (d) helpful in choosing the ordering of variables.)

(f) Confirm your answer to part (e) using the Quine-McCluskey method.

6. Design and draw out a circuit to subtract a positive 3-bit binary number $a = a_2a_1a_0$ from the positive 3-bit binary number $b = b_2b_1b_0$ (in general, there are no constraints on the relative magnitudes of a and b).

Extension questions

7. (This question contains non-examinable content; it is designed to test your ability to apply what you have learnt in lectures to a new, unseen scenario. We separate the meaning of a truth value from how it is physically encoded.)

Throughout lectures and your work so far, you have used a single bit to encode a logical `0` and logical `1`. That is, we use a low voltage on a single wire to indicate a logical `0` and a high voltage on the same single wire to indicate a logical `1`, as follows:

code Q_0	**meaning**
`0`	logical `0`
`1`	logical `1`

Suppose we now introduce a second wire into our encoding, to form a *dual-rail encoding*, as follows:

code Q_1Q_0	**meaning**
`00`	clear
`01`	logical `0`
`10`	logical `1`

The encoding `11` is unused (one quarter of the available encoding space is wasted). This is an example of embedding control signals (here, that computation has finished) within the data. Until now, our control paths (e.g. a clock) and data paths (e.g. the binary encodings of two numbers to be added) have been completely separate.

We could use the additional "clear" meaning to encode the validity of a signal. That is, we could design circuits where the circuit itself is able to tell us whether the resulting value is valid, or still being computed within the system. The fact that all signals start at `0` means that the outputs are guaranteed to be free of hazards.

(a) Design an inverter circuit that works in this new *dual-rail* encoding. Recall that an inverter is a circuit that changes a logical `0` into a logical `1`, and a logical `1` into a logical `0`. It does not make sense for an inverter to modify the validity of a signal.

(b) Recall that a conventional half-adder computes the sum and carry bits given two single-bit inputs, A and B. It has the following truth table:

A	B	H	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

where H is the sum of A and B, and C is the carry bit.

In our new dual-rail encoding, observe that this truth table becomes:

A_1	A_0	B_1	B_0	H_1	H_0	C_1	C_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	1	0	1
0	1	1	0	1	0	0	1
1	0	0	0	0	0	0	0
1	0	0	1	1	0	0	1
1	0	1	0	0	1	1	0

Explain why the second and third lines of this truth table are correct, establish Boolean equations for H_1, H_0, C_1 and C_0, and hence draw a circuit for the new half-adder implementation in the *dual-rail* encoding.

(c) Compare the number of gates used in each implementation of the half adder (that is, the conventional implementation and the *dual-rail* version). Assume that XOR gates are *not* available.

(d) Assuming for this supervision that all gates have an equal propagation delay (i.e. it takes T picoseconds for a signal to propagate through each of an AND gate, an OR gate, or a NOT gate), what is the maximum delay of each half adder implementation?

8. Hozier College is a constituent college of the University of Grantabridge. The Porters' Lodge at Hozier College contains two rooms: one that is visible to students and the public, and one that is concealed, that only the Porters may enter. For most of the day, the Porters are accustomed to napping in the latter of these rooms. However, there have recently been complaints about the lack of "portering" that is occuring in the College, so the Porters have purchased a Porter-Poking-Machine (PPM) to wake them up when necessary. It is your job to design the circuit that activates the PPM.

   You are supplied with 5 inputs:

   (i) The plodge is non-empty ($\overline{U}$).

   (ii) The fire alarm is not silent ($\overline{W}$).

(iii) The fellows are not happy (they need bringing more port from the cellar) ($\overline{X}$).

(iv) The students are not calm (they are rioting) ($\overline{Y}$).

(v) The post is all sorted (Z).

The PPM wakes up the porter when its input is low. It should wake up the porters when one or more of the following scenarios are true:

(i) The plodge is empty, BUT the fellows need more port, OR

(ii) The plodge has a student waiting in it AND the fire alarm is going off AND the students are rioting elsewhere in College, OR

(iii) The plodge has a student waiting in it AND the fire alarm is going off AND the post needs sorting, OR

(iv) The plodge has a student waiting in it AND the post needs sorting BUT the students are calm, OR

(v) The fellows are happy AND the fire alarm is going off AND there is a student waiting in the plodge.

Given the above information,

(a) Write down a Boolean expression for $\overline{P}$, i.e. a Boolean expression detailing when the PPM should be triggered. Don't try and simplify it yet, just copy the conditions verbatim from the outline above.

(b) What do you notice about the Boolean expression that you have just written down (hint: look at question 4)?

(c) Draw out a circuit for a minimised version of your PPM driver.