Work for Digital Electronics Supervision III

The questions below are partitioned into two sets: the *core* questions and the *extension* questions. Please attempt all four core items and hand in your work to me as usual by 1800 on Thursday, 29th October. Attempting the extension questions is optional.

Core questions

1. Please start by re-attempting any of questions 7, 8 and 9 from last week's supervision work for which you made significant errors. A "significant error" includes forgetting to take account of "don't care" states, or for a state diagram, anything other than a single misplaced arrow. It does not include a heedless error in just one or two of the flip-flop inputs in a state transition table.

2. For the following Boolean function,

$$F = \overline{A}.\overline{B}.\overline{C} + A.\overline{C}.\overline{D} + \overline{A}.C.D + B.C.\overline{D} + \overline{B}.C.D$$

   show how it may be implemented using:

   (a) one 16:1 multiplexor

   (b) one 8:1 multiplexor and one or more NOT gates.

   For (b), try both of the two reasonable methods: first use Boolean algebra, and then use a method based on the function's truth table. Compare/contrast the answers that you obtain.

3. Examination question: 2003 Paper 2 Question 2 (attached towards the end of this document).

4. Anybody who has ridden in a taxicab from Churchill College to the train station will be all-too-familiar with the crossroads at the junction of Lensfield Road, Regent Street and Hills Road. There are traffic lights at this junction to support the multiplexing of the following use cases:

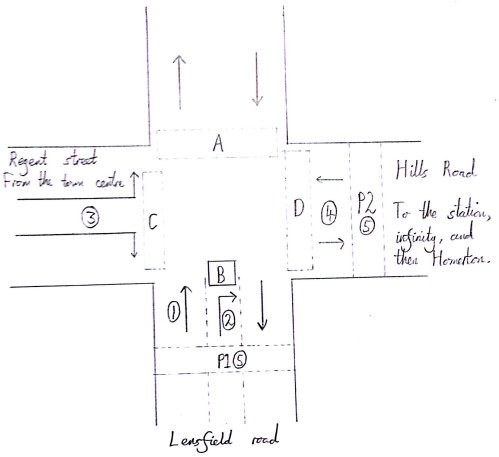
   (1) Cars going straight on along Lensfield Road.

   (2) Cars turning right from Lensfield Road onto Hills Road.

   (3) Cars turning left or right, or going straight on, from Regent Street.

   (4) Cars turning left or right from Hills Road onto Lensfield Road.

   (5) Pedestrians crossing the roads at the points marked $P1$ and $P2$ on the following diagram:

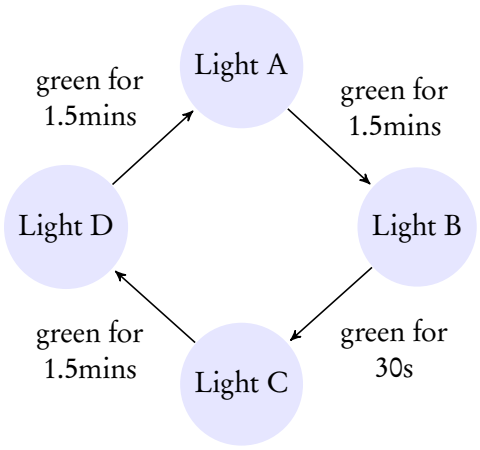


The circled numbers on the diagram correspond to the above use case numbers, and the letters within boxes are used to label each set of traffic lights. (The eagle-eyed observer will spot the incongruity with the actual road layout, which I've inserted so that solving the problem does not become overly tedious.)

Only one of the sets of traffic lights labelled A, B, C and D may be green at any one time; the others must be red. For example, if the set of lights A is green, all others must be red, and traffic can successfully travel straight on along Lensfield Road. Clearly this supports use case (1). If instead set B were at green, and all others were at red (for use case (2)), traffic could turn right from Lensfield Road onto Hills Road, but no other traffic could move. Lights always transition from green to red and back again via the normal UK traffic light sequence: Red→Red+Amber→Green→Amber→Red→ ….

Ordinarily, for use cases 1, 3 and 4, lights stay at green for 1 minute and 30 seconds before transitioning to red so that another set of lights can transition to green. However, for reasons known only to themselves, Cambridge City Council only allow cars following use case (2) to go for 30 seconds, exactly one third of the time allotted for each of the other use cases. This is despite full knowledge that use case (2) supports by far the most important movement of vehicles: those travelling from Churchill College to the station!

The following diagram represents the ordinary pattern in which the lights allow cars through (transitions for individual lights between Red/Green/intermediate states are omitted):

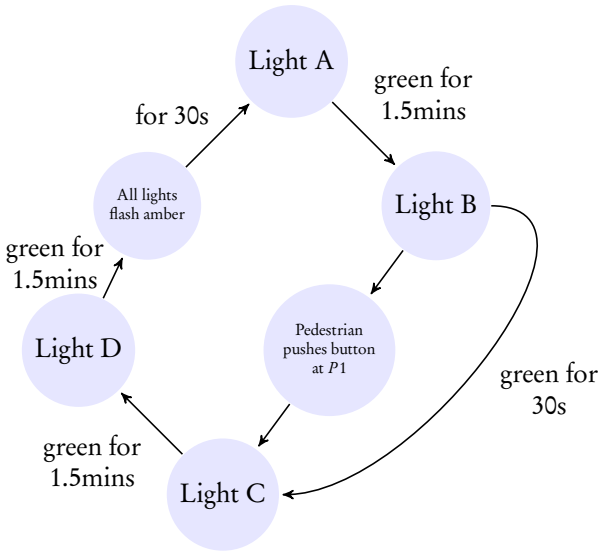


However, this sequence can be interrupted if a pedestrian presses one of the buttons at crossing $P1$ or $P2$. In either of these cases, the ordinary flow of events should continue until Light D has transitioned to red and then:

(a) Light A should *not* immediately transition to green as it would normally

(b) Instead all four sets of lights should flash amber for 30 seconds. While flashing, the amber light should spend 1 second on, followed by one second off, and so on.

(c) Then Light A can transition to Green (through Red+Amber) as normal, and the ordinary flow of events can resume:



Note that this diagram only illustrates one point at which the pedestrian can press the button: it can, in fact, be pressed at any time.

It is your task to implement the logic that controls such behaviour of the traffic lights at this junction.

You are supplied with the following resources:

(a) An ideal 1Hz clock signal.
(b) As many D flip-flops and/or JK flip-flops as you require.
(c) As many combinatorial components and connecting wires as you require.
(d) A debounced input from each of the two pedestrian crossings, each active high.
(e) Three buffered outputs to each traffic light, connected directly to each bulb (i.e. there are 12 outputs in total), all active high.

Do not answer this question in the style of an exam question, where you would present and explain a good solution to the problem and justify why you chose it. Instead, at each stage of your answer, compare and contrast all the possible choices that you could make, and then go on to choose one of them, and explain why what you chose is sensible.

Examples of such design decisions include, but are not limited to:

- A Mealy or Moore machine?
- Possible state assignments
- Possible state encodings

Lay out your answer as if you were writing a proposal: start by outlining and contrasting high-level approaches, and then fill in the details when the approach that you're taking is clear. If any aspect of the desired light behaviour is unclear, please feel free to email me for clarification, or come up with your own sensible interpretation of events.

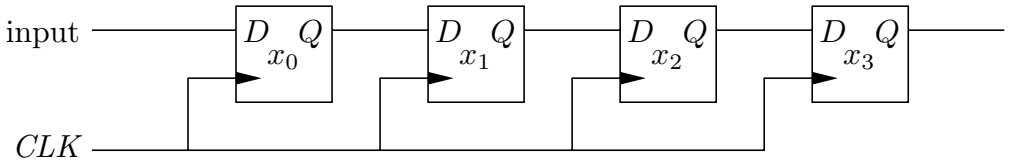
Extension questions

5. Examination question: 2005 Paper 2 Question 2
6. If you didn't do it last week, have a look at 1996 Paper 2 Question 3.

2003 Paper 2 Question 2

Digital Electronics

(a) A 4-bit shift register constructed from edge-triggered D-type flip flops is shown below. If, on successive rising edges of the clock signal *CLK*, the input takes on the values $1, 0, 1, 0, 1, 1, 1, 0$, what are the contents of the shift register after each edge of the clock? You may assume that the register contains all zeroes initially.



[4 marks]

(b) Using a (possibly larger) shift register, show how one may detect a particular pattern in the input shown. As an example, use the 8-bit pattern `0xF0`. High-order bits precede low-order bits in the input stream. [4 marks]

(c) The input stream is framed by a one byte frame pattern (`0xF0`) every 256 bytes. However, the frame pattern may also appear at an arbitrary position in the input stream.

It is required to design a framing circuit which generates two outputs: *framelock*, which is asserted when the circuit "believes" it has determined where the frame boundaries are, and *frame pointer*, which is asserted on the clock edge immediately after the frame marker is detected. The circuit "believes" itself to be locked to the frame structure when two successive frame patterns have been found 256 bytes (i.e. 2048 bits) apart. The circuit should not respond to unaligned frame patterns while it believes itself to be in lock or if, once in lock, it has missed fewer than two expected frame patterns.

Draw a state diagram for the finite state control of the circuit. You may assume the existence of an 11-bit resettable counter. You should consider the process of assuming lock, maintaining lock and the accommodation of a single missed framing pattern. State explicitly any additional assumptions you make.

[10 marks]

(d) Outline the complexity in gates and flip flops for an implementation of the framing circuit. [2 marks]

2005 Paper 2 Question 2

Digital Electronics

(a) A Moore machine is required which produces the counting sequence 0,1,2,3,4,5,0. Give a minimum sum-of-products for each of the next state variables for an implementation of this Moore machine. [6 marks]

(b) Design a two-bit Gray code counter which produces the binary sequence 00,01,11,10,00. The counter should be designed as a Moore machine consisting of D flip-flops (with enable inputs) and a minimal number of logic gates. An additional input (`E`) is required to enable or disable counting which can be connected directly to the enable inputs of the D flip-flops. What is the final circuit diagram? [6 marks]

(c) The 0→5 and Gray-code counters are coupled together to produce a state machine with following state sequence and output pattern in Morse code for SOS ($\cdots\ ---\ \cdots$):

State sequence		Output
00	000	0
00	001	1
00	010	0
00	011	1
00	100	0
00	101	1
01	000	0
01	001	1
01	010	1
01	011	1
01	100	0
01	101	1
11	000	1
11	001	1
11	010	0
11	011	1
11	100	1
11	101	1
10	000	0
10	001	1
10	010	0
10	011	1
10	100	0
10	101	1

With the aid of a circuit diagram, explain how the two counters are coupled together to produce the sequencer, and how the required Morse code output can be generated from this sequencer. [8 marks]