

T3

Discrete Digital Circuits

This lab exercise gives you experience of designing, building and testing sequential digital circuits using 74HC series discrete integrated circuits. You will use basic logic gates and D-type flip-flops to implement a simple traffic light controller using an algorithmic state machine. You will also see how logic '0' and '1' are represented in real digital circuits, and gain an appreciation of different logic levels. Furthermore, you will see the effects of propagation delays in integrated circuits, comparing synchronous and asynchronous digital counters.



Schedule

Preparation time : 3 hours

Lab time : 3 hours

Items provided

Tools : n/a

Components : 74HC08 (2x), 74HC32 (2x), 74HC74 (3x),
SPST Momentary Off-(On) Push Switch

Equipment : Oscilloscope, Signal Generator, Power Supply, Traffic Light

Software : n/a

Items to bring

Essentials. A full list is available on the Laboratory website at
<https://secure.ecs.soton.ac.uk/notes/ellabs/databook/essentials/>

Before you come to the lab, it is essential that you read through this document and complete **all** of the preparation work in section 2. If possible, prepare for the lab with your usual lab partner. Only preparation which is recorded in your laboratory logbook will contribute towards your mark for this exercise. There is no objection to several students working together on preparation, as long as all understand the results of that work. Before starting your preparation, read through all sections of these notes so that you are fully aware of what you will have to do in the lab.

Academic Integrity – *If you undertake the preparation jointly with other students, it is important that you acknowledge this fact in your logbook. Similarly, you may want to use sources from the internet or books to help answer some of the questions. Again, record any sources in your logbook.*

Revision History

October 05, 2012

Geoff Merrett (gvm)

New lab for 2012/13 (loosely based on B7/8)

1 Aims, Learning Outcomes and Outline

This laboratory exercise aims to:

- Provide an introduction to combinational logic circuits
- Provide an introduction to sequential digital circuits and algorithmic state machines
- Give you experience of building simple digital systems using discrete digital ICs

Having successfully completed the lab, you will be able to:

- Understand the operation of logic gates and their implementation in ICs
- Understand the operation of D-type flip-flops
- Compare synchronous and asynchronous counters
- Design and build a sequential digital system using discrete digital ICs
- Demonstrate the function of digital electronic system and show that this is in accordance with design requirements




This lab exercise gives you experience of designing, building and testing sequential digital circuits using discrete integrated circuits (ICs), or ‘chips’, containing logic gates. You will use basic logic gates (the 74HC08 quad 2-input AND gate, and 74HC32 quad 2-input OR gate) and flip-flops (the 74HC74 dual D-type flip-flop) to implement a simple traffic light controller using an algorithmic state machine. You will also see how logic ‘0’ and ‘1’ are represented in real digital circuits, and gain an appreciation of different logic levels. You will also investigate the effects of propagation delays in ICs, and compare synchronous and asynchronous counters.

2 Preparation

Read through the course handbook statement on safety and safe working practices, and your copy of the standard operating procedure. Make sure that you understand how to work safely. Read through this document so you are aware of what you will be expected to do in the lab.





2.1 The 74HC08, and 74HC32 (AND and OR Gates)

Find the manufacturer’s datasheet for the 74HC08 and 74HC32 ICs from the T3 lab webpage (or search for one using Google). Using the datasheets where appropriate, answer the following questions:

-  *What is the difference between a 74HC08 and a 74HCT08?*
-  *What voltages represent ‘logic 0’ and ‘logic 1’ for the 74HC series ICs you will be using in the lab?*
-  *Find the pinouts for the 74HC08 and 74HC32, and copy them into your logbook.*
- Draw and complete a truth table for each of the ICs in your logbook.*

2.2 The 74HC74 (The D-Type Flip-Flop)

In the laboratory, you will use D-type flip-flops to implement a sequential digital circuit, and also explore their propagation delays. Find the manufacturer’s datasheet for the 74HC74 D-type flip-flop from the T3 lab webpage (or search for one using Google). Using the datasheet, answer the following questions:

-  Find the pinout for the 74HC74 and copy it into your logbook.
-  When does the 74HC74 D-type flip-flop transfer the logic value from the 'D' input to the 'Q' output?
-  For a D-type flip-flop such as the 74HC74, what is meant by the "hold time", "setup time" and "propagation delay"?
-  For 5V operation at 25°C, what does the manufacturer say is 'typical' on the 74HC74 for the delay between the clock being asserted and the output Q being updated?

2.3 Synchronous and Asynchronous 'Divide-by-4' Counters

In this exercise, you will investigate the performance of synchronous and asynchronous divide-by-four counters.

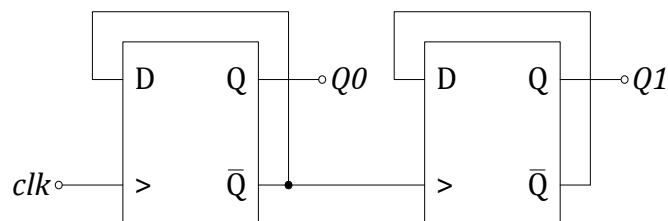



FIGURE 1: A divide-by-four ripple counter.

The schematic for a divide-by-four ripple counter is shown in Figure 1 (in this configuration, the counter counts down rather than up). Each output changes at a different time relative to the clock – the output changes ripple along the chain of flip-flops, giving the configuration its name.

-  Consider the counter shown in Figure 1. Assume that both outputs Q0 and Q1 are currently at 'logic 1'. Estimate the delay in Q1 changing to 'logic 0' after the next rising edge of the clock.

Hint: use the propagation delay you identified in Section 2.2.

Ripple counters have only limited usefulness because their shortcomings are often unacceptable. Some text books and students believe that ripple counters can be adapted to count at other moduli – these adaptations have even more shortcomings. It is always much simpler to use synchronous state machine design to yield reliable solutions. Figure 2 shows an ASM chart, state transition table and Boolean expressions for a synchronous divide-by-four counter (which counts up).

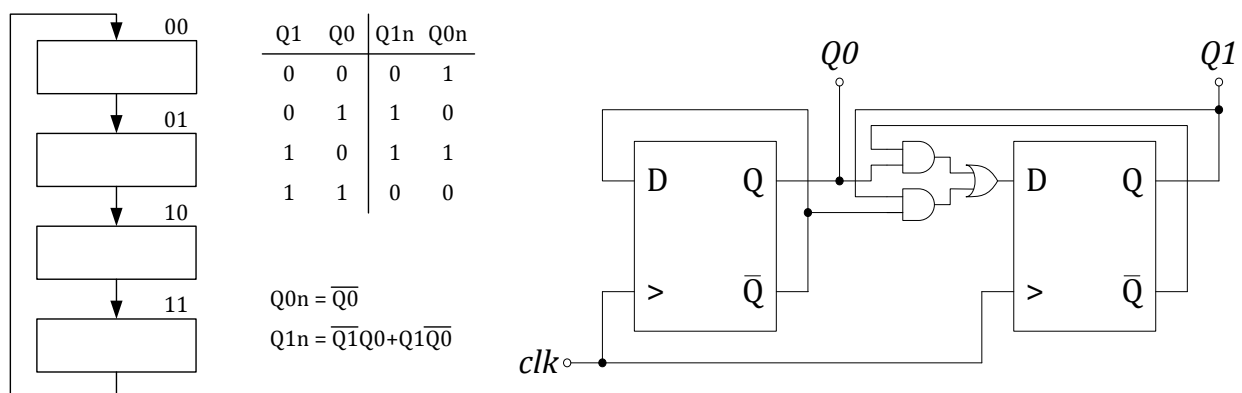


FIGURE 2: ASM chart, state transition table, Boolean expressions and schematic for a synchronous divide-by-four counter.

- ◇ Assume that both outputs $Q0$ and $Q1$ are currently at 'logic 1'. Estimate the delay in $Q1$ changing to 'logic 0' after the next rising edge of the clock.
- ◇ Consider a divide-by-128 ripple counter and a divide-by-128 synchronous counter (you do not need to design these!). For each of these, estimate the maximum propagation delay.

2.4 The Traffic Light

In the lab, you will be building and testing a simple traffic light controller.

- ◇ What is meant by a 'SPST Momentary Off-(On) Push Switch'?

A controller is required to drive a simple UK-format traffic light. The traffic light contains three LEDs: Red, Yellow and Green, and goes through the sequence shown in Figure 3.

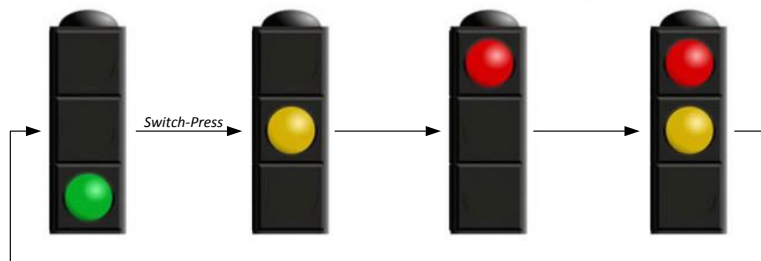


FIGURE 3: Desired operation of the traffic light controller.

The traffic light should normally be Green. When a switch (you are supplied with a SPST momentary off-(on) push switch) is pressed for more than one clock cycle, the traffic light should then go through the sequence: \rightarrow Yellow \rightarrow Red \rightarrow Red and Yellow \rightarrow Green (spending a single clock cycle in each).

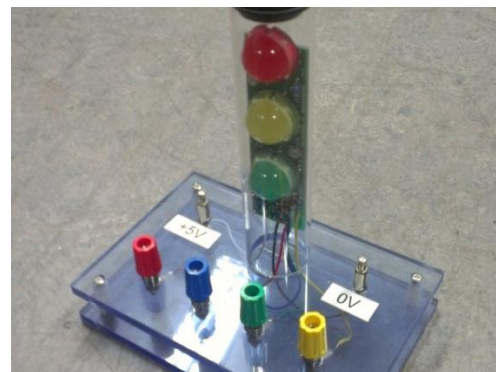
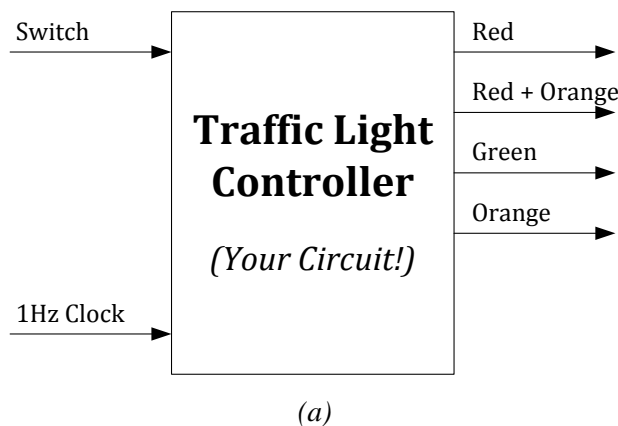


FIGURE 4: (a) The inputs and outputs to the traffic light controller, and (b) the traffic light used in this exercise. Note, the four terminals along the bottom of (b) connect to the four outputs on the right hand side of (a).

The traffic light that you will be using in the lab has four inputs (in addition to V_{cc} and Gnd), as shown in Figure 4b. The four coloured terminals along the bottom of the traffic light control the lights:

- Red Terminal: Red LED;
- Blue Terminal: Red and Yellow LEDs;
- Green Terminal: Green LED;
- Yellow Terminal: Yellow LED.

The traffic light contains internal protection circuitry meaning that you can drive the terminals directly from the outputs on your circuit (i.e. without needing to add any additional current-limiting resistors).

Figure 4a shows the inputs and outputs to the traffic light controller – the sequential digital circuit which you need to input. A timing diagram illustrating correct operation of the traffic light controller is shown in Figure 5.

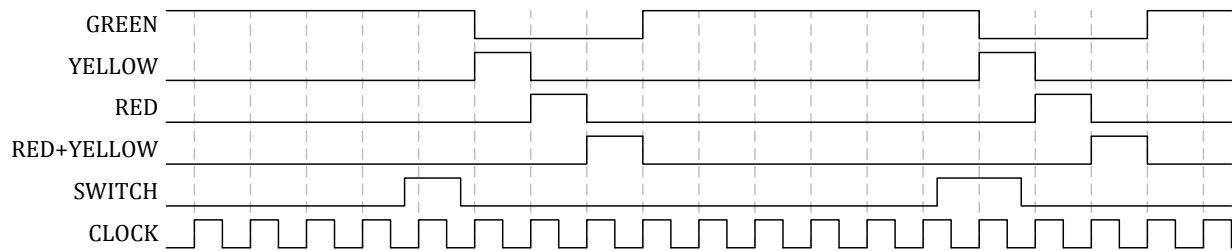





FIGURE 5: Timing diagram illustrating correct operation of the traffic light controller.

-  *Design a circuit to implement the traffic light controller. You should include an ASM chart, state transition table, Karnaugh-Maps, and a complete circuit diagram.*
-  *Identify the next-state logic and the output logic in your circuit diagram. Is your design a Mealy or Moore machine? Why? How can you also identify this from the ASM chart?*
-  *From your circuit diagram, count how many of each logic IC you are going to use (note: each IC contains multiple logic gates). On your circuit diagram, annotate each logic gate's inputs and outputs with the relevant pin number.*

When you come to your timetabled laboratory session, show your design to one of the demonstrators, who will check it for you.

3 Laboratory work

In this exercise you will use basic logic gates and D-type flip-flops to implement a simple traffic light controller using an algorithmic state machine. You will also investigate the effects of propagation delays in integrated circuits (ICs), and compare synchronous and asynchronous counters.

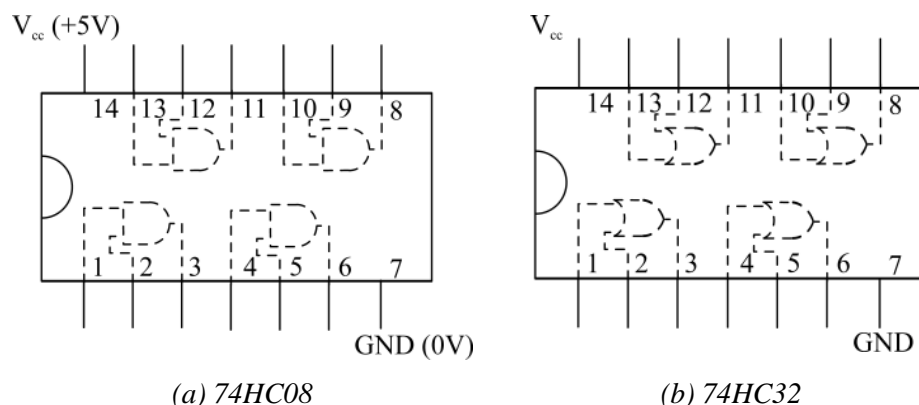




FIGURE 6: The logic gates used in this exercise (and their connections).

3.1 Basic Logic Gates


In your chip set, you will find some ICs containing logic gates. You'll find a 74HC08 which contains four 2-input AND gates, and a 74HC32 which contains four 2-input OR gates. The arrangement of the gates inside the ICs is shown in Figure 6.

Wire up the 74HC08 (quad 2-input AND gate) and verify correct behaviour by applying logic signals (use the bench power supply – 0V for 'logic 0' and 5V for 'logic 1') to the inputs while observing the outputs (using the oscilloscope).


 *Draw and complete a truth table for the 74HC08 in your logbook for the behaviour you observe.*

 *Do you get the same truth table that you obtained in your preparation?*


Setup the signal generator to provide a 100 Hz triangle wave from 0V to 5V. Use the oscilloscope to check that the voltage range is correct. Connect a BNC cable between the signal generator's 'TRIGGER' output and the oscilloscope's 'EXT TRIG' input. Setup the oscilloscope to trigger off its external input.


 *What is meant by 'triggering'?*


Connect the output of the signal generator to one input of an AND gate, and connect the AND gate's other input to logic '1'.

 *Monitor the output using your oscilloscope, and sketch in your logbook the oscilloscope trace that allows you to identify these voltages.*

Note: alternatively, you can save the trace from the oscilloscope, print it out and stick it in your logbook.

 *Identify the input voltage(s) at which the AND gate's output transitions from low-high and from high-low, i.e. what input voltages represent 'logic 0' and 'logic 1'?*

 *Is there a range of voltages where operation is 'undefined'? Is the voltage always the same? What impact does this have on circuit design?*


 *How do these measured voltages compare with those you obtained in section 2.1 of your preparation? Would you expect these voltages to be the same for every 74HC08 you tested (you don't actually need to test them to answer this!)*

3.2 A Traffic Light Controller

Build the traffic light controller that you designed in your preparation. Only use one protoboard – you'll need the second one later in the lab. Design your circuit so that you will be able to easily debug it if it doesn't work. Use different coloured wires to represent different signals (e.g. red: V_{cc} , black: Gnd, yellow: clk, etc).

Setup the signal generator to provide a 1Hz square wave between 0V and 5V, and connect this signal to your clock input(s).

You will need to use a pull-down resistor (see Figure 7) with the SPST off-(on) switch in order to ensure that it provides an output of 'logic 0' when the switch is not pressed (if the resistor was not there, the output would be 'floating').

 *How does a pull-down resistor work? What is a sensible resistance for a pull-down resistor?*

Connect your circuit up to the traffic light and verify its operation. Show your correctly working circuit to a demonstrator.

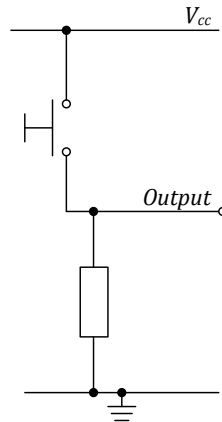


FIGURE 7: Pull-down resistor

Now increase the clock frequency to the fastest that the signal generator can produce.

- ◇ *Can you still verify that your traffic light is going through the correct sequence? If not, what is the fastest frequency you can verify it working at?*
Hint: you'll need to use an oscilloscope!

4 Optional Additional Work

Marks will only be awarded for this section if you have already completed all of Section 3 to an excellent standard and with excellent understanding.

Put your traffic light controller to one side (do not dismantle it, as the demonstrator will want to see it working when they mark the lab). Using your other protoboard, build the circuit for the divide-by-four ripple counter shown in Figure 1, and connect the clock input to a 0-5V 100 Hz square wave.

Using your oscilloscope, can you observe (and measure) the delay between the rising edge of the clock and the divide-by-four output?

- ◇ *Hint: you want to use both of your oscilloscope probes to look at both the clock and the output 'Q' of the second flip-flop. Think carefully about what you want to trigger on (the trigger source, rising- or falling-edge etc). Between which two points should you measure the delay?*

An example of the trace that you are looking for is shown in Figure 8, where the purple trace is the rising edge of the clock, while the green trace is the Q output from the second stage of the counter.

- ◇ *How does your measured delay compare with the delay you estimated in your preparation?*

You will probably find it quite hard to measure the delay, as it is approaching the capabilities of the oscilloscope. Reduce the supply voltage (and the amplitude of the clock) to 2V.

- ◇ *What is the propagation delay through the counter now?*
Hint: think again about the points you are measuring the delay between.

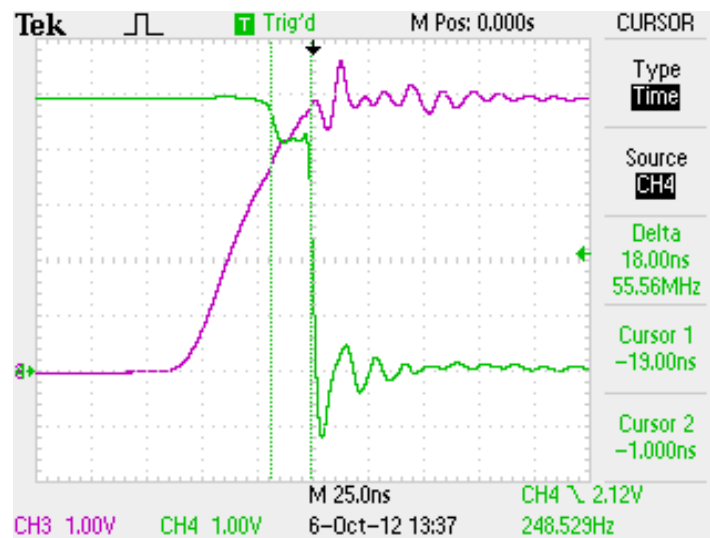


FIGURE 8: Measurement of the propagation delay through a divide-by-4 ripple counter made using 74HC74 D-Type Flip Flops. CH3 represents the input clock to the first stage, and CH4 the Q output from the second stage.

Finally, build the synchronous divide-by-four counter and measure the delay between the rising edge of the clock and the divide-by-four output.

◇ How does the performance of the two counters compare?