COMPUTER SCIENCE TRIPOS Part IB

Tuesday 4 June 2019 1.30 to 4.30

COMPUTER SCIENCE Paper 5

*Answer **five** questions.*

*Submit the answers in five **separate** bundles, each with its own cover sheet. On each cover sheet, write the numbers of **all** attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

STATIONERY REQUIREMENTS
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
Approved calculator permitted

1 Computer Design

(*a*) In SystemVerilog, what is the difference between:

(*i*) The ternary operator `?` and `if...then...else` statements? [2 marks]

(*ii*) `always_ff` and `always_comb`? [2 marks]

(*iii*) Blocking, non-blocking and continuous assignment? [3 marks]

(*iv*) Logic values 0, 1, x and z and how these values propagate through Boolean logic gates? [3 marks]

(*v*) The way that synchronous and asynchronous reset are declared in an `always_ff` statement? [2 marks]

(*b*) The following module attempts to implement a reset control circuit that should have the following behaviour: when the user-controlled reset button (which needs to be debounced) is pressed the `asyncButton` signal is high and should result in the `rst` going high and remaining high for a minimum of 10^6 clock cycles. `rst` should be generated immediately after the rising clock edge to allow time for it to propagate.

```
module timeResetBad(input  logic clk,
                    input  logic asyncButton,
                    output logic rst);
   logic ctr [18:0];
   logic ctrAtMax;
   always_comb begin
      ctrAtMax = &ctr;
      rst = !ctrAtMax;
   end
   always_ff @(posedge clk)
      ctr  <= asyncButton ? 0     :
              !ctrAtMax   ? ctr+1 : ctr;
endmodule
```

(*i*) What is wrong with the `timeResetBad` module? [4 marks]

(*ii*) Write a corrected version `timeResetBad` that makes minimal changes and adds no new modules. [4 marks]

2 Computer Design

(*a*) What is the difference between memory latency and bandwidth? [2 marks]

(*b*) What is the difference between an interrupt and an exception? [3 marks]

(*c*) Why is a computer's memory implemented as a hierarchy of memories and yet the programming model represents physical memory as a flat linear address space? [5 marks]

(*d*) What is the difference between a direct-mapped and a set-associate cache, and how do their cache-line replacement policies differ? [5 marks]

(*e*) Contemporary commercial processors use a register file to store intermediate results. Historically accumulators or operand stacks were used instead of a register file. What are the advantages of a register file over an accumulator and an operand stack? [5 marks]

3 Computer Design

(*a*) Describe each of the four models defined by OpenCL's specification. [4 marks]

(*b*) Describe the different types of memory available to OpenCL kernels. [4 marks]

(*c*) Contrast how calls to a kernel, e.g. DAXPY, are invoked and grouped for execution in OpenCL compared with CUDA. [4 marks]

(*d*) Describe, with the aid of a diagram, how a GPU executes data-parallel kernels efficiently, including the two main pieces of hardware support. [4 marks]

(*e*) Describe the trade-offs between using a GPU or a specialised accelerator for tasks containing data-level parallelism. [4 marks]

4 Computer Networking

The TCP transport protocol is an example of an ARQ.

(*a*) Provide a definition of an ARQ. [1 mark]

(*b*) Describe the design and operation of a simple ARQ for a lossy communication channel. [3 marks]

(*c*) When and why does a simple ARQ at the transport layer have a significant negative impact upon application performance? [3 marks]

(*d*) Describe what additions are required to a simple ARQ to support windowing. When and why will an ARQ which supports windowing provide better application performance than a simple ARQ? [5 marks]

(*e*) Describe two situations when the performance of a windowed ARQ is no better than a simple ARQ. [4 marks]

(*f*) Would the QUIC protocol, when based upon UDP rather than TCP, overcome the two situations you outlined when answering Part (*e*)? [4 marks]

5 Computer Networking

(*a*) You need to urgently deliver 500 TByte of data from Zurich to London.

(*i*) NoWayNet is offering a 10 Gbit/s reliable delivery service between Zurich and London (about 776km). Should you use either NoWayNet, or an overnight package delivery? Why? [2 marks]

(*ii*) A new company, UnlikelyComms, is offering a 400 Gbit/s reliable delivery service between Zurich and London, but it takes a very indirect 2600 km path. Should you use UnlikelyComms? [2 marks]

(*iii*) Following your successful urgent delivery of 500 TBytes of data, this has become an hourly task. Alongside the need to regularly deliver 500 TByte data between Zurich and London, you have an interactive virtual reality system; it requires six displays each needing 50 Gbit/s and an end-to-end latency of less than 5ms.

Fortunately, a startup, FlyByNight, boasts an offering with an effective bandwidth of over 16 Tbit/s, using a *special* transport with no end-to-end latency at all. The downside is it can only transfer data in 500 TByte units, once every 4 minutes. Explain whether FlyByNight is, or is not, suited to your two workloads. [4 marks]

(*b*) Ethernet standards enable 1Gbit/s over 4 pairs of twisted cabling, yet the physical media has a bandwidth much less than 1GHz, e.g., 250MHz is common.

(*i*) Explain how such high data rates are achieved, and [4 marks]

(*ii*) explain how physical media errors are reduced or eliminated. [2 marks]

(*c*) Explain, with the aid of diagrams, how Code-Division Multiple Access permits two or more pairs of nodes to communicate over a common medium (e.g., wireless) simultaneously. [6 marks]

6 Computer Networking

(*a*) Provide *five* examples of resource-multiplexing in computer networks. In each case state the layer of the network stack where this takes place, which resource is multiplexed and the multiplexing mechanism. [2 marks each]

(*b*) A relative has contacted you for help; they believe *the Internet is broken*. The fault turns out to be that a piece of network-protection software had installed a specific IP address entry for an (alternative) default DNS server.

(*i*) What should have provided the correct DNS entry? [1 mark]

(*ii*) By concentrating on how DNS is intended to work, describe why network applications may not work correctly. [3 marks]

(*iii*) How would you diagnose this problem? [2 marks]

(*c*) DNS substitution may also be used maliciously.

(*i*) Outline how spyware might use an (alternative) DNS entry. [2 marks]

(*ii*) Describe a network-centric method for overcoming such treachery. [2 marks]

7 Concurrent and Distributed Systems

(*a*) In the Network Time Protocol (NTP), a client (C) and a server (S) exchange (request, reply) messages to compute corrections to the time at C. Assume the time at S is always correct, and that C is synchronised to S at 13:30:00.

(*i*) Thirty days later the time at S is again 13:30:00 but C now believes the time to be 13:28:30. Define and compute *skew* and *drift* for C. [2 marks]

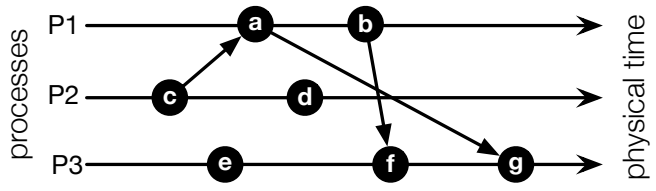
(*ii*) NTP estimates the *offset* and *delay* using four timestamps (T_0, T_1, T_2, T_3) from a request-reply message exchange. Two such exchanges occur between C and S, producing timestamps $(310.000, 400.100, 400.102, 310.202)$ and $(311.000, 401.150, 401.160, 311.410)$ respectively, denoting all timestamps as seconds since a common fixed point. Show on a diagram the point in the message exchange at which each timestamp $T_0 \ldots T_3$ is taken. Give definitions for *offset* and *delay*, and compute both for each set of timestamps. Which of the two offsets you have computed would you prefer to use to adjust the time at C, and why? [5 marks]

(*iii*) What happens to your estimates of offset and delay if network delays are no longer symmetric? [2 marks]

(*b*) It is often necessary to agree only on the *ordering* of events, not their times.

(*i*) $x \to y$ indicates event x *happens-before* event y. Define *happens-before*. Explain why it provides only a partial order on events. [2 marks]

(*ii*) *Vector clocks* can be used to implement *happens-before*. Give the vector clock values at each event, $a \ldots g$, and explain whether each of the following relations is true or false: $b \to c$, $c \to e$, $c \to f$, $d \to g$. If false, give the relation that does hold between the given pair of events.



[8 marks]

(*iii*) An earlier approach used *Lamport Clocks*, defining $L(x)$ such that, for two events x and y, $x \to y \Rightarrow L(x) < L(y)$ but $L(x) < L(y) \not\Rightarrow x \to y$. Explain how vector clocks resolve this issue and ensure $L(x) < L(y) \Rightarrow x \to y$.

[1 mark]

8 Concurrent and Distributed Systems

(*a*) Programs with concurrency are vulnerable to classes of problems that are not exhibited in single-threaded programs.

(*i*) Explain the concepts of *deadlock* and *livelock* in a multithreaded program. [2 marks]

(*ii*) Explain the four conditions required for deadlock. A sentence explaining each condition is sufficient. [4 marks]

(*b*) Modern instruction set architectures provide instructions for performing atomic operations over memory locations.

(*i*) One class of instructions are generically referred to as Compare And Swap (CAS). Describe how the CAS instruction on the x86 architectures is used to perform an atomic operation. Briefly explain how the instruction uses the instruction operands when executed. [3 marks]

(*ii*) Databases often utilise a technique known as *write-ahead logging* to provide durability guarantees. Describe how a disk-based transaction log might be implemented, and what the atomic operation used in this technique is. [3 marks]

(*c*) The below snippet of C code uses `pthreads` for concurrent execution. It uses a mutex `M` and a condition variable `C` to ensure that the `run_critical_code` function only executes when the `condition` boolean is `true`.

```
L1:    pthread_mutex_lock(&M);
L2:    run_critical_code ();
L3:    if (!condition)
L4:      pthread_cond_wait(&C, &M);
L5:    if (!condition)
L6:      pthread_cond_broadcast(&C);
L7:    pthread_mutex_lock(&M);
```

Unfortunately, there are four bugs in the code that prevent it from working correctly. List each of the four bugs and describe how each bug affects programme execution. Write down a new version of the code snippet with the four bugs corrected. [8 marks]

END OF PAPER