COMPUTER SCIENCE TRIPOS Part IB

Wednesday 5 June 2013 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

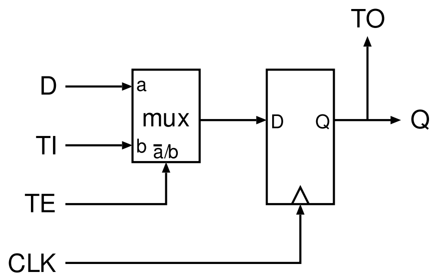
SPECIAL REQUIREMENTS
Approved calculator permitted

1 Computer Design

(*a*) What is the difference between production test and functional test? [2 marks]

(*b*) What is path sensitisation for production test? Provide an example. [4 marks]

(*c*) Below is the circuit for a JTAG scan flip-flop. How are the signals TI, TO, TE and CLK used to undertake production test? [6 marks]



(*d*) In SystemVerilog, what is the purpose of an "initial" block and how is timing specified? [4 marks]

(*e*) What is static timing analysis and why is it important for a design to pass timing if it is to be functionally correct? [4 marks]

2 Computer Design

The version of Thacker's Tiny Computer 3 (TTC3) that was used in the 2012 ECAD Laboratory sessions (instruction set summary is below) has the following pipeline stages:

fetch	decode/register fetch	execute/memory access	write-back

Currently the implementation only supports one instruction in the pipeline at a time, i.e. the next instruction is only fetched when the current one finishes in the write-back stage.

If the implementation were to attempt to fetch a new instruction every clock cycle, explain the following microarchitectural issues:

(*a*) What data hazards would exist and how can they be resolved whilst preserving the programmer's sequential model? [5 marks]

(*b*) What are control hazards and how can we avoid exposing them to the programmer? [5 marks]

(*c*) When are branch target addresses computed on the TTC3 and how many bubbles will be introduced when taking a jump? Assume that such a tiny computer would not have a branch predictor. [5 marks]

(*d*) On the TTC3, every instruction (except jump) can conditionally skip the next instruction. How might skip be implemented and how many pipeline bubbles need to be introduced? [5 marks]

TTC3 Instruction Set Summary

Rw	LC	Ra	Rb	Func	Shift	Skip	Opcode
31–25	24	17	10	7	5	3	0

Function:
- 0: A+B
- 1: A-B
- 2: B+1
- 3: B-1
- 4: A & B
- 5: A | B
- 6: A ^ B
- 7: reserved

Shift (rotates right):
- 0: no shift
- 1: RCY 1
- 2: RCY 8
- 3: RCY16

Skip:
- 0: never
- 1: ALU<0
- 2: ALU=0
- 3: InRdy

Opcode:
- 0: normal: Rw=F(Ra,Fb), skip if condition
- 1: storeDM: DM[Rb]=Ra, Rw=F(Ra,Rb), skip if condition
- 2: storeIM: IM[Rb]=Ra, Rw=F(Ra,Rb), skip if condition
- 3: out: OutStrobe, Rw=F(Ra,Rb), skip if condition
- 4: loadDM: Rw=DM[Rb], ALU=F(Ra,Rb), skip if condition
- 5: in: Rw=in_data, ALU=F(Ra,Rb), skip if condition
- 6: jump: Rw=PC+1, PC=F(Ra,Rb), no skip
- 7: reserved

LC=load constant (bits 23:0 of the instruction), no skip
PC=program counter
ALU=Function(Ra,Rb), where the Function is specified by the Func bits
F(Ra,Rb)=rotate(Shift, ALU), where the rotation is specified by the Shift bits

3 Computer Design

(*a*) Moore's law is an observation by Gordon Moore in the 1960s that trends in electronic manufacturing technology would result in transistor density doubling every 18 to 24 months.

(*i*) Explain how Moore's law can be applied to processor speed and hard disk density. [5 marks]

(*ii*) Today transistor scaling favours transistors over wires and thermal densities limit performance. Why is this resulting in commercial chips having many processor cores rather than one high-performance processor core? [5 marks]

(*b*) PCI, used to connect I/O boards to a PC, has been replaced with PCIe. This transition has resulted in parallel communication being replaced by bundles of serial communication channels.

(*i*) What is the difference between parallel and serial communication? Why are multiple serial channels now preferred to a parallel link? [5 marks]

(*ii*) Why might the latency of a single load of a register on a PCIe device take longer than on PCI? [5 marks]

4 Computer Networking

(*a*) A spy elects to use a self-synchronizing scrambler to encode his secret message. Explain why this will not give him any privacy and why his self-synchronising approach would be better used by a communications engineer. [5 marks]

(*b*) With the assistance of annotated diagrams explain CSMA/CD and CSMA/CA.

In your explanation, note the physical constraints on packets and networks that these approaches impose. [10 marks]

(*c*) Consider the network buffer sizing formula $B = 2T \times C$

(*i*) Explain this formula. [2 marks]

(*ii*) Discuss the network architecture and traffic assumptions made in the use of this formula. [3 marks]

5 Computer Networking

Here are four options for improving web page performance.

Option 1: HTTP Caching with a Forward Proxy

Option 2: CDN using DNS

Option 3: CDN using anycast

Option 4: CDN based on rewriting HTML URLs

You have been asked to help reduce the costs for networking in the University.

(*a*) The University pays its service provider *networks'r'us*, based on the bandwidth it uses; bandwidth use is dominated by students downloading external web pages. Which, if any, of the above four options would reduce the bandwidth usage?

Explain your choice. [4 marks]

(*b*) The delivery of online courses has become a tremendous success – but this has led to a significant increase in network costs for the University.

You must select one of the options above to minimize load on the servers. Compare the operation of each option and justify a selection that provides the finest granularity of control over load to the content-servers and a selection that will serve each customer from the closest CDN server. [12 marks]

(*c*) You have looked up the IP address of your favourite search engine on the University network and noticed the answer is different from that given to your friend when he did the lookup in Newfoundland, Canada.

For each option above, indicate why it might, or might not, be used by your favourite search engine to improve web page performance. [4 marks]

6 Computer Networking

(*a*) Considering either TCP/IP or UDP/IP, write a description of how server-port, client-port, source-port and destination-port relate to each other. You may wish to give examples and use diagrams as appropriate. [4 marks]

(*b*) What is a routing-loop? Include a diagram in your answer. [4 marks]

(*c*) Describe a mechanism that prevents routing-loops in Ethernet networks. [4 marks]

(*d*) (*i*) Describe and, with the aid of an example, illustrate the IP Time-To-Live (TTL) mechanism for minimising the impact of routing-loops. [2 marks]

(*ii*) Assuming, in part (*d*)(*i*), a perfect implementation, describe a disadvantage of the approach including the symptoms that might be experienced in a network subject to this disadvantage, and a test that may identify the problem. [2 marks]

(*e*) Explain the technical and architectural argument behind the decision in IPv6 to retain header TTL but not a header checksum. [2 marks]

(*f*) Explain why there is ambiguity about handling packets with TTL values of 1 and give a practical solution. [2 marks]

7 Concurrent and Distributed Systems

(*a*) Deadlock is a classic problem in concurrent systems.

(*i*) What are the four necessary conditions for deadlock? [4 marks]

(*ii*) Deadlock is often explained using the Dining Philosopher's Problem. In this pseudo-code, each fork is represented by a lock:

```
Lock forks[] = new Lock[5];

// Code for each philosopher (i)
while (true) {
    think();
    lock(fork[i]);
    lock(fork[(i + 1) % 5]);
    eat();
    unlock(fork[i]);
    unlock(fork[(i + 1) % 5]);
}
```

Partial ordering is a common deadlock prevention scheme. Describe modifications to the above code, changing only array indices, such that philosophers can be fed not only safely, but also deadlock-free, using a partial order. [4 marks]

(*b*) Priority inversion can occur when threads of differing priorities synchronise on access to common resources – threads of greater priority may end up waiting on threads of lesser priority, leading to undesirable realtime properties.

(*i*) Describe how this problem can be solved for mutexes using priority inheritance. [2 marks]

(*ii*) Describe how priority inheritance would need to be modified to handle reader-writer locks. [2 marks]

(*iii*) Priority inversion can also arise between two threads involved in process synchronisation – for example, when one thread uses a semaphore to signal completion of work. Why might implementing priority inversion be more difficult with process synchronisation than with mutual exclusion? [4 marks]

(*iv*) What could we do to solve the problem in (*b*)(*iii*)? [4 marks]

8 Concurrent and Distributed Systems

(*a*) The ACID properties are often used to define transactional semantics.

(*i*) Define "atomicity" as used in the ACID context. [1 mark]

(*ii*) Define "durability" as used in the ACID context. [1 mark]

(*b*) Write-ahead logging is a commonly used scheme to accomplish transactional semantics when storing a database on a block storage device, such as a hard disk.

(*i*) Under what circumstances, during write-ahead log recovery, can a transaction in the UNDO list be moved to the REDO list? [2 marks]

(*ii*) Synchronously flushing commit records to disk is expensive. How can we safely reduce synchronous I/O operations on a high-throughput system without sacrificing ACID properties? [2 marks]

(*iii*) Describe two performance changes that might arise from using your solution to part (*b*)(*ii*). [2 marks]

(*c*) (*i*) Transaction records in a write-ahead logging scheme contain five fields: $\langle \mathit{TransactionID}, \mathit{ObjID}, \mathit{Operation}, \mathit{OldValue}, \mathit{NewValue} \rangle$, but storing the complete old and new values can consume significant amounts of space. One strategy that might be employed, for reversible operations being applied to some data such as XOR by a constant, is to store only the constant arguments, rather than the full before and after data. What problems might occur as a result of this design choice? [4 marks]

(*ii*) Write-ahead logging systems must know the actual on-disk sector size for the write-ahead log to behave correctly. An errant disk vendor decides to rebrand its 512-byte sector disks as 2K-sector disks, and adjusts the value reported back to the database system. How might this affect database integrity? [4 marks]

(*iii*) Explain how a database vendor who is aware of the problem described in part (*c*)(*ii*) mitigate this problem in software, and what limitations might there be to this approach. [4 marks]

9 Concurrent and Distributed Systems

Sun's Network File System (NFS) is the standard distributed file system used with UNIX, and has gone through a progression of versions (2, 3, 4) that have gradually improved performance and semantics.

(*a*) Remote procedure call (RPC)

(*i*) Explain how Sun RPC handle byte order (endianness). [2 marks]

(*ii*) This approach may result in unnecessary work. State when this occurs and how might this be avoided. [2 marks]

(*b*) Network File System version 2 (NFSv2) and version 3 (NFSv3)

(*i*) A key design premise for NFS was that the server be "stateless" with respect to the client. State what this means for distributed file locking in NFSv2 and NFSv3. [2 marks]

(*ii*) Another key design premise for NFSv2 was the "idempotence" of RPCs; what does this mean? [2 marks]

(*iii*) One key improvement in NFSv3 was the addition of the READDIRPLUS RPC. Explain why did this helps performance. [4 marks]

(*iv*) NFSv3 implements what is termed "close-to-open consistency" for file data caching: if client C1 writes to a file, closes the file, and client C2 now opens the file for read, then it must see the results of all writes issued by C1 prior to close. However, if C2 opens the file before C1 has closed it, then C2 may see some, all, or none of the writes issued by C1 (and in arbitrary order). Close-to-open consistency is achieved through careful use of synchronous RPC semantics, combined with file timestamp information piggybacked onto server replies on all RPCs that operate on files.

Explain how close-to-open consistency allows performance to be improved. [4 marks]

(*v*) NFSv3 adds a new RPC, ACCESS, allowing the client to delegate access control checks at file-open time to the server, rather than performing them on the client. This allows client and server security models to differ.

Explain how this addition also helps implement close-to-open consistency in the presence of read caching. [4 marks]

END OF PAPER