UNIVERSITY OF DUBLIN

TRINITY COLLEGE

Faculty of Engineering and Systems Sciences
Department of Computer Science

B.A.(Mod.) Computer Science
Junior Sophister Examination

Trinity Term 2002

3BA4 - Computer Architecture II

Monday 27th May | MANSION HOUSE | 09.30 - 12.30

Dr. Jeremy Jones, Dr. Andrew Butterfield

Attempt **FIVE** questions at least two from each section.
Use separate answer books for each section.

SECTION A

Q1. What is pipelined processor? What are the benefits of pipelining? Explain the organization & operation of the DLX five stage execution pipeline.

What are data hazards? Describe two techniques that can be used to overcome such hazards. Illustrate your answer with some simple examples.

What is a load hazard? Describe a technique that can be used to overcome such hazards. Illustrate your answer with an example

Q2. What is a delayed jump? What is a delay slot? Show how delayed jumps can improve the overall execution time of a program by considering the "unoptimised" and "optimised" RISC-I code that would be executed for the following code segment.

```
i = 0;
while (i<j)
        i = k + f(i);
```

Q3. What is a cache? How does a cache reduce the effective memory access time of a CPU?

Explain how a cache organisation can be characterised by the three constants L, K & N. What special names are given to cache organisations where (i) N=1 (ii) K=1 and (iii) K=4. Explain in detail how a cache is searched for the contents of a memory address. What actions take place on a cache hit and a cache miss?

What is the difference between a virtual and a physical cache? A single CPU system has a DMA controller that can transfer data directly from disk to memory. If the CPU contains an on-chip cache explain how the contents of the cache can become inconsistent with memory. How can this coherency problem be solved if the on chip cache is (i) a physical cache and (ii) a virtual cache?

Q4. What is the cache coherency problem? Under what conditions are the caches in a system considered to be coherent?

Explain (i) the meaning of the 4 cache line states used in the Firefly cache coherency protocol and (ii) the basic operation of the protocol. Given a 3 CPU + cache multiprocessor system, illustrate the bus traffic and cache state transitions that would occur if the following CPU memory requests are issued:

CPU 0: read a2
CPU 0: write a2
CPU 0: write a2
CPU 1: read a2
CPU 1: read a0
CPU 0: write a2
CPU 0: write a2

Assume (i) each cache is direct mapped with 2 cache lines (ii) even addresses map to line 0 & odd addresses to line 1 and (ii) the caches initially contain addresses a0 & a1.

SECTION B

Q5 Given the following logic function: Y = NOT(A+B+(C*D))

(i) Design a CMOS switch circuit that implements this function. Your design should include a determination of the widths of the transistors involved (assuming a minimum width of 0.1μm, and a μ_n/μ_p ratio of **2.5**).

(ii) Express the CMOS layout topology of your circuit using a Stick Diagram, subject to the following constraints: Power and Ground run horizontally across the top and bottom respectively, of the circuit, *inputs A and B enter on the right, while inputs C and D enter on the bottom*, on Polysilicon, and the *output emerges on the left*, also on Polysilicon.

Q6 (i) For the following types of design rules: width, separation, overlap and extension, describe the nature of the rules and explain what manufacturing problems they are designed to solve.

(ii) A particular separation rule requires that contact cuts be a certain minimum distance from transistor gate regions. What is the reason for this ?

(iii) Some electrical design rules are concerned with keeping current densities at low limits. How do these rules apply to contact cuts, and what does this mean for the layout of contact cuts designed to handle large currents ? Illustrate your answer with an example.

Q7 Context is important in IC Design - a given logic function may have different forms in various parts of an integrated circuit according to the differing relationships it has with its neighbours. Use the design of a full adder (described below) as an example to show how some of its subparts with the *same* logic function end up having *different* physical implementations.

```
Fulladder(A,B,Cin) = (S,Cout)
     where
          (S1,C1) = HalfAdder(A,B)
          (S,C2)  = HalfAdder(S1,Cin)
          Cout = C1 + C2

HalfAdder(A,B) = (S,C)
     where
          S = EXOR(A,B)
          C = A + B
```

Q8 (i) Show how CMOS technology, in an appropriate form, can be used to produce regular programmable logic arrays (PLAs).

(ii) Sketch out a PLA, *with only four product lines*, to implement the following logic (here /X means NOT(X)):

X = /A*B*C + /C
Y = /A*C
Z = /A*/B*C + A

where A, B and C are inputs and X, Y and Z are outputs. Your answer should show how the PLA building blocks are put together and the effect of the programming.