UNIVERSITY OF DUBLIN

TRINITY COLLEGE

Faculty of Engineering and Systems Sciences
Department of Computer Science

B.A. (Mod.) Computer Science           Trinity Term 2005

Junior Sophister Examination

3BA4 – Computer Architecture II

Wednesday 1\textsuperscript{st} June 2005   Regent House   09:30 – 12:30

Dr. J. Jones & Dr. A. Butterfield

Instructions to Candidates:

Attempt \textbf{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.

(4 marks)

What are data hazards? Describe two techniques that can be used to overcome data hazards in the DLX pipeline. Show how the data hazards in the following code sequence are overcome by the DLX CPU.

\[
\begin{align*}
  r1 & \leftarrow r2 + r3 \\
  r4 & \leftarrow r1 + r0 \\
  r10 & \leftarrow r1 + r4 \\
  r11 & \leftarrow r1 + r10
\end{align*}
\]

(10 marks)

What is a load hazard? Describe a technique that can be used to overcome such hazards. Show how the load hazards in the following code sequence can be overcome when executed by the DLX CPU.

\[
\begin{align*}
  r1 & \leftarrow \text{mem}[a] \\
  r1 & \leftarrow r1 + r1 \\
  r2 & \leftarrow \text{mem}[b] \\
  r2 & \leftarrow r2 + r2 \\
  r3 & \leftarrow r3 + r3
\end{align*}
\]

(6 marks)

Q2. What effect would a "simple" implementation of branch instructions have on the execution of the DLX pipeline?

(2 marks)
What is branch prediction? Explain the operation of a branch prediction scheme that could be added to the DLX processor to improve the execution of branch instructions. Show how the branch prediction hardware would improve the execution of the following code sequence.

\[
\begin{align*}
  r1 &\leftarrow 10 \\
  L1: &\quad \ldots \\
  &\quad \ldots \\
  &\quad r1 \leftarrow r1 - 1 \\
  &\quad \text{BNEZ } r1, L1 \\
\end{align*}
\]

(10 marks)

What is a "Two bit Branch Prediction Scheme" and explain how, and in what circumstance it would improve the execution of the code sequence above.

(4 marks)

Compare the effective cycles per instruction (CPI) of (i) a "simple" branch implementation and (ii) a branch prediction scheme. Assume 20% branch instructions, 95% probability of hitting the branch target buffer, 95% probability of a correct prediction and a 1 cycle penalty if the branch target buffer needs updating.

(4 marks)

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

(1 mark)

Explain how a cache organisation can be characterised by the three constants LKN. Explain how a data item is searched for in an LKN cache.

(6 marks)

What special names are given to cache organisations where (i) N=1 (ii)
K=1 and (iii) K=4.

(1 mark)

Compute the number of hits & misses if the subsequent list of hexadecimal addresses is applied to

(1) a 128 byte direct mapped cache with 16 bytes per line and
(2) a 128 byte 4-way set associative a cache with 16 bytes per line.

0x0000 → 0x0004 → 0x0008 → 0x000c → 0x0010 → 0x0020 →
0x0030 → 0x0040 → 0x0080 → 0x0060 → 0x0000 → 0x0040 →
0x0000 → 0x0020 → 0x0014 → 0x0030

Assume that (i) the lower 4 bits is used as the offset into the cache line and the next \( \log_2(N) \) bits select the set and (ii) all cache lines are initially empty/invalid.

(12 marks)

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

(2 marks)

Clearly define the meaning of the states used by Firefly cache coherency protocol and briefly describe its operation.

(4 marks)

Given a three CPU multiprocessor system using the Firefly protocol, illustrate what happens when and the following memory requests are made (e.g. bus traffic and cache line state transitions):
CPU 0: read a0
CPU 0: read a2
CPU 0: write a2
CPU 0: write a2
CPU 1: read a2
CPU 1: write a2
CPU 1: write 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 (iii) the caches initially contain addresses a0 & a1.

(12 marks)

What difference would it make if the protocol didn't have a shared and dirty state (SD)?

(2 marks)
Section B

Q5. (i) Design a CMOS circuit that implements a two-input OR-gate: \( F = OR(A,B) \). Your design should include a determination of the widths of the transistors involved (assuming a minimum transistor width of 90nm, and a \( \mu_n/\mu_p \) ratio of 2).

(10 marks)

(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 \( B \), and output \( F \) all enter on both the top and bottom, on Polysilicon (A, B and F need not occur in the same order on both top and bottom).

(10 marks)

Q6. Consider a process technology where the effective resistance and load capacitance of a minimum-sized inverter are 10kΩ and 1fF respectively.

(i) Design a chain of inverters to drive a load of 250fF as fast as possible.

(10 marks)

(ii) Design a 1000\( \mu \)m long, 1\( \mu \)m wide line with inverters added at intervals, to be as fast as possible, given that the line has the following characteristics: sheet resistance: 20Ω/square, Capacitance/Unit Area: 0.05fF/\( \mu \)m\(^2\).

(10 marks)

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 an 2-input EXOR gate (using 4 NANDs, as shown below) as an example to show how some of its subparts with the same logic function end up having different physical implementations.

2-input EXOR gate built from 4 NAND-gates:

\[ \text{Exor}(A, B) \]

(20 marks)

Q8. (i) Explain the term “pseudo-nMOS”, show how it is used to reduce the transistor count in CMOS logic gates, and explain its disadvantages.

(6 marks)

(ii) Explain how the existence of two clock signals, one active low, the other active high, with non-overlapping active periods, can be used to produce “dynamic CMOS” logic. Illustrate you answer by showing how the logic function \( Y = \text{NOT}(A + B \cdot C) \) would be implemented as a dynamic gate.

(10 marks)

(iii) Explain where a dynamic gate stores information, and hence explain why processors implemented with dynamic logic have a minimum clocking speed.

(4 marks)