ENGINEERING TRIPOS   PART IIA

ELECTRICAL AND INFORMATION SCIENCES TRIPOS   PART I

Friday 9 May 1997   9 to 12

Paper E6

COMPUTING SYSTEMS

Answer not more than **five** *questions.*

All questions carry the same number of marks.

The **approximate** *number of marks allocated to each part of a question is indicated in the right margin.*

(TURN OVER

1 (a) Explain what is meant by a *pipeline hazard.* Distinguish between data and branch hazards. [3]

(b) The following extract of MIPS assembler code increments all the elements of an n-element array by the contents of `$10`. The starting address of the array is in `$9` and $4n$ is in `$11`.

```
Loop:   lw $8,0($9)          # $8 loaded with data at address $9+0
        add $8,$8,$10        # $8 loaded with $8+$10
        sw $8,0($9)          # $8 stored at address $9+0
        addi $9,$9,4         # $9 loaded with $9+4
        bne $9,$11,Loop      # Jump back 4 instructions if $9≠$11
```

The instructions are executed on a pipelined datapath with the following five stages:

Stage 1	Instruction fetch
Stage 2	Instruction decode and register fetch
Stage 3	Execution and effective address calculation
Stage 4	Memory access
Stage 5	Write back data to registers

The datapath has no facilities for flushing the pipeline. If hazards are resolved by stalling the pipeline, how many clock cycles does the code segment take to execute? [2]

(c) The datapath is upgraded to include a data forwarding facility. How many clock cycles does the code in (b) take to execute now? Illustrate your answer with a diagram showing clearly how data are forwarded to the `add` and `sw` instructions. [3]

(cont.

(d) The compiler has an optimization facility which "unrolls" loops. Assuming n is a multiple of 2, show how the code in (b) can be rewritten to execute in $13n/2$ clock cycles on the datapath *with* forwarding. Your code should loop $n/2$ times, processing two elements of the array each time round the loop. [3]

(e) Another compiler optimization facility allows instructions to be re-ordered. Show how, by changing the zero offset in the `sw` instruction, the code in (b) can be re-ordered to execute in $8n$ clock cycles on the datapath *with* forwarding. [3]

(f) Comment on the relative advantages and disadvantages of the optimization schemes in (d) and (e). What assumptions did you make when calculating the execution times of the code segments? Under what circumstances might the code in (e) execute faster than the code in (d)? [3]

(g) Explain how *delayed branches* could be used further to reduce the number of stalls. [3]

(TURN OVER

2 (a) Figure 1 shows a 4-bit adder constructed from four 1-bit full adders. Explain how *carry lookahead* can be used to determine the carry inputs to each full adder. Using generate and propagate signals gi and pi, show that the expression for c1 is

$$c1 = g0 + p0.c0 \qquad (1)$$

and derive similar expressions for c2 and c3. State how gi and pi are calculated from the input signals ai and bi. If the propagation delay through an AND or OR gate is T, how long does the adder take to perform a 4-bit addition? [3]

An alternative way to construct a 4-bit adder is to connect together two 2-bit adders using a higher-level of carry lookahead, as shown in Fig. 2. Derive expressions for G0 and P0, the higher level generate and propagate signals, in terms of p0, p1, g0 and g1. Hence obtain an expression for C1 in terms of G0, P0 and c0. If the 2-bit adders use the expression in equation (1) to obtain their internal carry signals, how long does this alternative design take to perform a 4-bit addition? [3]

Discuss the relative advantages and disadvantages of the two designs. Under what conditions might the two-level lookahead adder operate faster than the single-level lookahead adder? [4]

(b) Why are processor-memory buses generally synchronous, while I/O buses tend to be asynchronous? [3]

In a particular computer system, it takes a total of 12 clock cycles to read an 8-word block from memory into the CPU, and 14 clock cycles to write an 8-word block from the CPU into memory. The CPU has a write-back cache, and the miss rate is measured at 0.05 misses per instruction. 40% of cache misses require a write-back operation, while the remaining 60% require only a memory read.

(i) Suggest why it takes longer to write a block into memory than it takes to read a block from memory. [2]

(cont.

(ii) Assuming the processor is stalled for the duration of a miss, find the number of cycles per instruction spent handling cache misses. [2]

(iii) The cache system is changed to work with 16-word blocks. If the larger block size halves the miss rate, with the same fraction of misses requiring write-back, estimate how many cycles per instruction are spent handling cache misses now. [3]

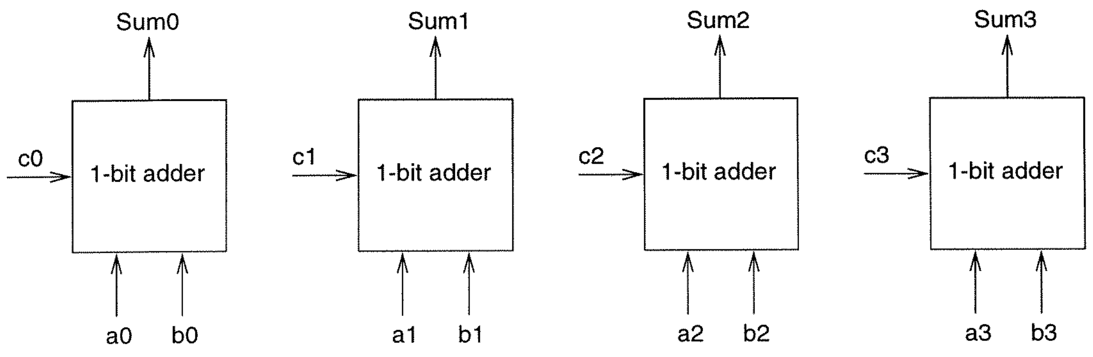

Fig. 1

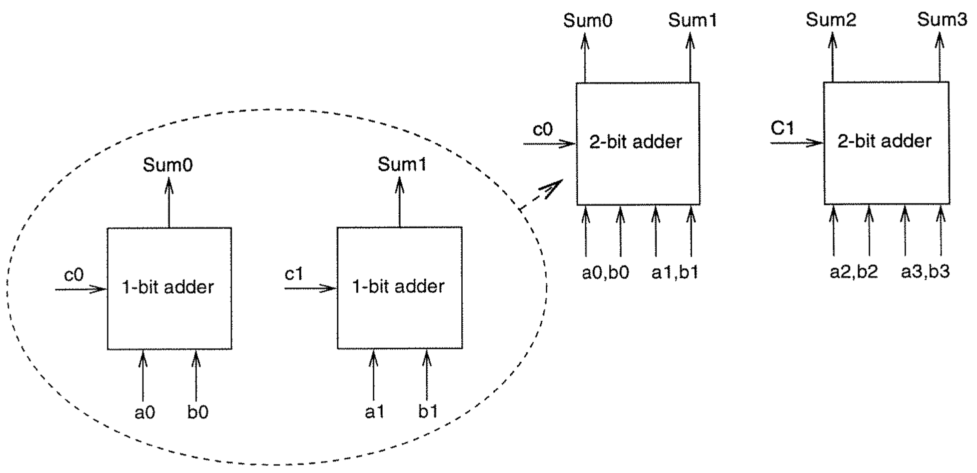

Fig. 2

(TURN OVER

3 Figures 3 and 4 illustrate part of a Pascal module which implements the collection of data from a set of sensors connected via a small input/output (IO) network. Fig. 3 provides the definitions and Fig. 4 shows the code.

(a) Explain how the readability, modifiability and maintainability of this program could be improved by using features present in modern programming languages like C++ and Ada. [6]

(b) Illustrate the use of these features by rewriting the code in Fig. 4 using a Pascal-like notation enhanced with these features. You may assume that similar features have been added to the `byteio` module. You should briefly describe the semantics of any language features which you add. [9]

(c) What, if any, further changes would need to be made if this code were to be run in a multi-process system in which `read_sensor` could be called concurrently by more than one process? [5]

```
{-- Definitions imported from byteio module --
CONST OP_READ = 1;
      OP_WRITE = 2;
      OP_OK = 0;
      OP_FAIL = 255;
FUNCTION send_byte (b:byte):boolean;
{send byte b on IO network, return success/fail}
FUNCTION read_byte (VAR b:byte):boolean;
{get byte b from IO network, return success/fail}
PROCEDURE reset_byte io;
{remove any pending input/output bytes from IO network}
-- End of definitions from byteio module -- }
```

Fig. 3

(cont.

```
TYPE sensortype = (switch, xy, temperature);
     sensor = RECORD
                id:byte;
                stype :sensortype;
                data :ARRAY [1..4] OF BYTE; {up to 4 bytes}
              END;
FUNCTION read_sensor (VAR s:sensor):boolean;
VAR  ret:boolean;
     tmp:byte;
BEGIN
  ret:=false;
  IF NOT send_byte(OP_READ) THEN goto end_read_sensor;
  IF NOT send_byte(s.id) THEN goto end_read_sensor;
  IF NOT read_byte(tmp) THEN goto end_read_sensor;
  IF tmp <> OP_OK THEN goto end_read_sensor;
  IF NOT read_byte(tmp) THEN goto end_read_sensor;
  IF tmp <> s.id THEN goto end_read_sensor;
  CASE s.stype OF
    switch:
      IF NOT read_byte(s.data[1]) THEN goto end_read_sensor; {0 or 1}
    xy:BEGIN
      IF NOT read_byte(s.data[1]) THEN goto end_read_sensor; {x low byte}
      IF NOT read_byte(s.data[2]) THEN goto end_read_sensor; {x high byte}
      IF NOT read_byte(s.data[3]) THEN goto end_read_sensor; {y low byte}
      IF NOT read_byte(s.data[4]) THEN goto end_read_sensor; {y high byte}
    END
    temperature:  BEGIN
      IF NOT read_byte(s.data[1]) THEN goto end_read_sensor; {low byte}
      IF NOT read_byte(s.data[2]) THEN goto end_read_sensor; {top 4 bits}
    END
  END;
  ret:= true;
end_read_sensor:
  IF NOT ret THEN reset_byte_io;
  read_sensor:= ret
END;
```

Fig. 4

(TURN OVER

4 (a) Explain what is meant by the term *software lifecycle* and describe how this can be split into a number of interacting stages. Indicate briefly the purpose of each stage, the types of personnel involved and the types of notation employed. [8]

(b) A system is to be built which produces a sorted list of employee names with the corresponding email addresses and telephone numbers. The names to be listed are supplied as input to the system. The telephone number and email address are to be looked up in two separate databases both keyed on the name in a standard form (e.g. A.N. Other); the input may be presented in variants of this (e.g. Alan N. Other). The final output is to include text formatting instructions (e.g. such as for the LaTeX system).

Produce a dataflow diagram describing this system. [6]

(c) Explain how the dataflow diagram produced in (b) could be used in:

(i) a procedural design strategy; and [4]

(ii) object oriented design. [2]

5 Explain briefly what is meant by *Bayes' minimum error rate pattern classification.* [4]

A diagnostic problem of identifying people exposed to a certain type of adverse medical condition uses one feature variable, denoted x. The probability distributions of this feature for the adverse and benign conditions are assumed to be Gaussian with parameters given by $\{\mu_a, \sigma\}$ and $\{\mu_b, \sigma\}$ respectively. Note that the means are distinct and the variances are equal.

Taking $\mu_a = 1, \mu_b = 0$ and $\sigma = 1$, sketch the two distributions.

Stating any assumptions made, derive an expression for the posterior probability of adverse outcome given a measurement x, for a Bayes' optimal classifier. Sketch this probability as a function of x. [5]

Consider a classification rule of the form

$$\begin{cases} \text{adverse if } x > \theta \\ \text{benign if } x \leq \theta \end{cases}$$

Show, by shading the appropriate area in a sketch of the distributions, how the *true positive rate* and the *false alarm rate* can be calculated. [5]

Compute their values for

(a) $\theta = 0.2$;

(b) θ set to the Bayes' optimal value. [6]

(TURN OVER

6 A two-class pattern classification problem characterised by one variable, denoted x, is defined by the following measurements:

$$\begin{cases} \text{Class One}: \; -2,\; 0,\; 2 \\ \text{Class Two}: \; 3,\; 4 \end{cases}$$

A linear discriminant function of the form $g(x) = w_0 + w_1 x$ is to be designed to classify the data. The *target values* are defined as

$$\begin{cases} g(x) = +1, \; x \text{ in Class One} \\ g(x) = -1, \; x \text{ in Class Two} \end{cases}$$

Compute the unknown parameters of this discriminant function that minimise the least squares interpolation error over the given data. Sketch the resulting discriminant function and comment on your results. [10]

Deriving the expressions necessary, explain how a *gradient descent* algorithm may be used to minimise the above interpolation error. [6]

How would the *perceptron algorithm* perform on the above task? [4]

7 (a) One way of representing intelligent behaviour in Artificial Intelligence is by means of search trees. Within this context:

(i) Explain briefly the term *combinatorial explosion.* [2]

(ii) Explain briefly the difference between *blind* and *heuristically informed* search, mentioning why the latter method can alleviate the problem of combinatorial explosion. [3]

(iii) State the formula for calculating the number of nodes searched in a tree of branching factor b and depth d when using blind search. [2]

(b) Figure 5 shows a simplified map of Southern England. Using this map, you have to plan a journey from Bristol to London that takes the shortest total distance.

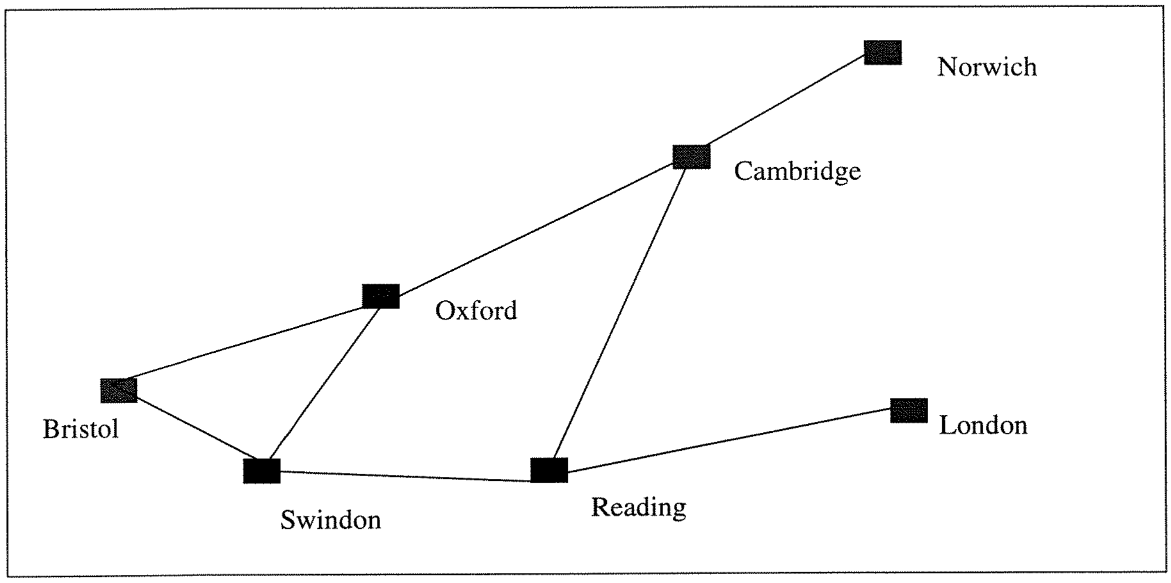

Fig. 5

(TURN OVER

(i) Draw a tree that represents this search problem. Label each node clearly with the initial letter of each town. [3]

(ii) Write down the order of letters corresponding to a path taken in an exhaustive: a) depth-first, b) breath-first search. Do not include letters for re-traced steps. [2]

(iii) In the general case where a map has N towns, what is the maximum number of levels in the corresponding tree representation? [2]

(c) The evaluation function for A* search is:

$$f^*(n) = g^*(n) + h^*(n)$$

where

$g^*(n)$ is the minimum cost of a path from the start node to a node n,

$h^*(n)$ is the estimate of the minimum cost of a path from node n to a goal node, and

$f^*(n)$ is the estimate of the cost of a path passing through node n.

Explain how $f^*(n)$ could be used to guide the search.

In a large journey planning problem, briefly state what happens to the planning behaviour if $h^*(n)$ is:

(i) exactly equal to the true distance,

(ii) considerably less than the true distance,

(iii) considerably more than the true distance?

Describe briefly the factors to be considered when designing the function $h^*(n)$. [6]

8 (a) Draw a block diagram showing the main components of a typical expert system. Summarise the function of each component. [5]

(b) You are building an expert system to aid in the process of material selection. Your expert gives you the following knowledge:

(i) “Diamond is harder than steel”.

(ii) “Steel is harder than copper”.

(iii) “If material x is harder than material y, then x will cut y”.

(iv) “If material x can cut material y, and material y can cut material z, then x will cut z”.

Your program accepts queries in predicate logic format and returns either ‘true’ or ‘false’. By going through the following steps, show how your program would prove the truth of the assertion that diamond can cut copper, expressed thus:

Cuts(diamond, copper)

(i) Express the above knowledge in first-order predicate logic.

(ii) Convert the expressions into clause form.

(iii) Using resolution, show how the assertion would be proven. [9]

(c) A system is to be designed which contains information on 1000 different materials, each having 50 properties. Discuss the feasibility of extending the above approach to this application. [6]

END OF PAPER