Data Provided: None

The University Of Sheffield.

DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING

Autumn Semester 2007-2008 (2 hours)

Advanced Computer Architectures 4

Answer **THREE** questions. **No marks will be awarded for solutions to a fourth question.** Solutions will be considered in the order that they are presented in the answer book. Trial answers will be ignored if they are clearly crossed out. **The numbers given after each section of a question indicate the relative weighting of that section.**

1. a. A method for producing the sine/cosine of an arbitrary angle using only trivial multiplies (shifts) can be based on an implementation of the CORDIC (**co**-ordinate **r**otation **di**gital **c**omputer) algorithm. The algorithm begins with two values x_0 =1 and y_0 =0, an initial angle, β_0 =0, and an angle, θ, and iterates towards the final values of $x_N = \cos\theta/K$, $y_N = \sin\theta/K$, and $\beta_N = \theta$ as follows:

$$x_{i+1} = x_i - S_i 2^{-i} y_i$$
$$y_{i+1} = S_i 2^{-i} x_i + y_i$$
$$\beta_{i+1} = \beta_i + S_i \gamma_i$$

where S_i = 1 when $\theta - \beta_i > 0,$
= -1 when $\theta - \beta_i < 0$

and γ_i = arctan(2^{-i})

Somebody had devised some simple hardware to implement the algorithm, as shown in **Figure 1**.

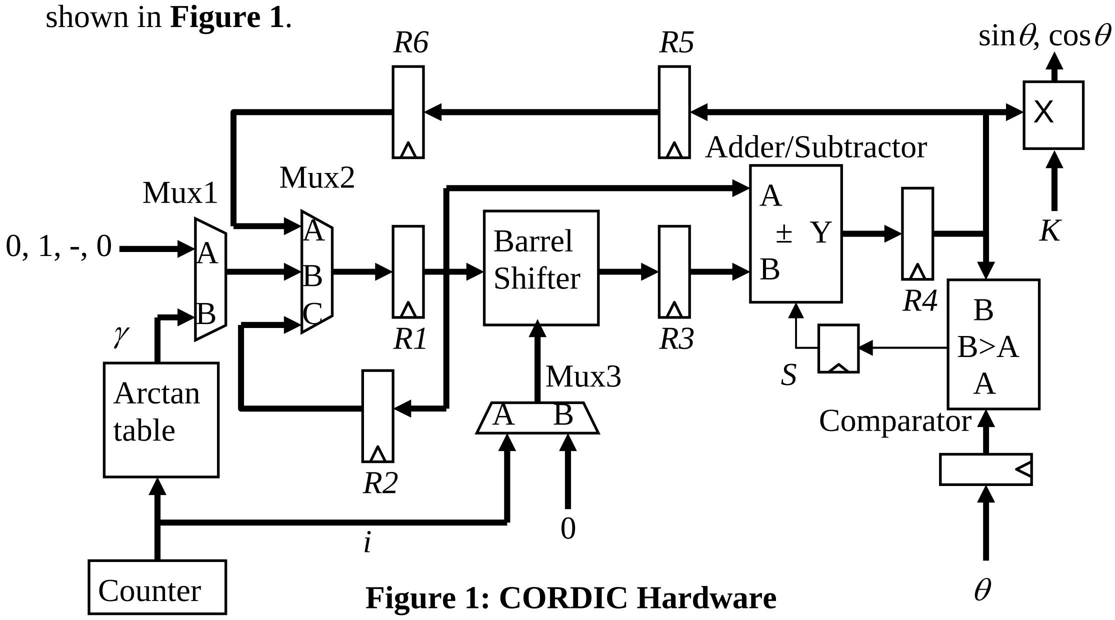

Figure 1: CORDIC Hardware

The counter provides the value of *i* that is used to control the barrel shifter (which can provide a controllable shift of the data equivalent to multiplying it by 2^{-i}) during each iteration and uses a table to look-up the value of arctan(2^{-i}). During each iteration, at the appropriate point, the comparator is used to compare the value of β_{i+1} (calculated during iteration *i*) with θ to set the value of S_{i+1} for the next iteration. The value of *S* controls the adder/subtractor, which can perform A+B, A-B, and B-A, to add or subtract as required.

All that the designer tells you is that the data moves, word-serially, around the system in the following order: y_i, x_i, -, β_i, γ_i (- denotes no specific value) and that the initial conditions are set by inserting the values 0,1,-,0,arctan(1) at the left hand side whilst the initial value of *S* is set to 1.

The final output is provided by multiplying x_N, and y_N by K (=0.60725) after the final iteration.

i. Show how the algorithm is mapped onto the hardware by producing a reservation table for the first few iterations of the algorithm. Ensure that you identify the state of the two multiplexers at each clock cycle and the operation performed by the adder/subtractor. **(15)**

ii. You know that 40 iterations will produce the correct result but you want to improve the design so that the result will be output in the minimum number of iterations: how might this be done? **(5)**

2. a. Describe the organisation of a vector processor and how it differs from a scalar processor. Ensure that your answer includes a diagram identifying the various parts clearly and the role that they play. **(6)**

b. i. What problems are encountered when loading vectors from multi-dimensional arrays held in memory? **(4)**

ii. How might these problems be addressed? **(4)**

c. Which of the following code *snippets* would transfer easily onto a vector processor and why would this be the case?

i. `for (i=0; i<N; i++) z[i]=x[i)*y[i];` **(2)**

ii. `for (i=L; i<N; i++) z[i]=x[i]*z[i-L];` **(2)**

iii. `for (i=0; i<(N-1); i++) z[i]=x[i]*z[i];` **(2)**

where `N` *is the length of the array and* `L` *is a positive integer that is less than* `N`

3. A set of 32 processors is interconnected via a Cross-Point-Switch as shown in **Figure 3**.

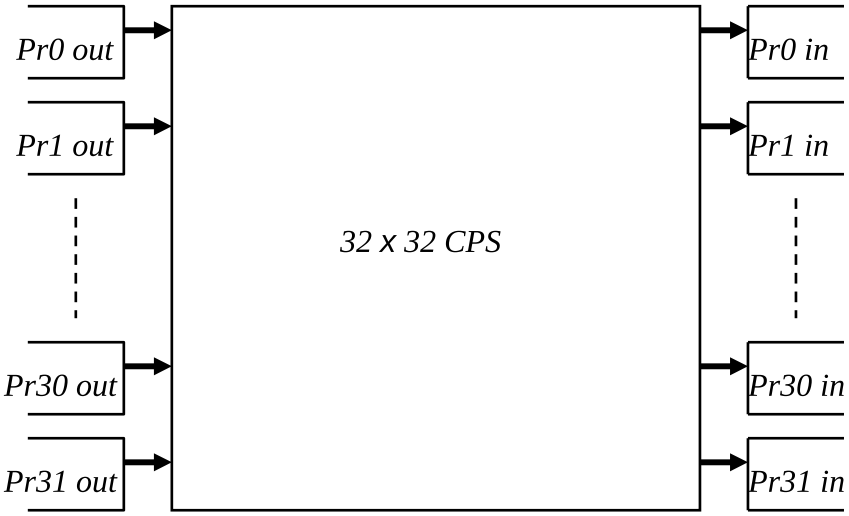

Figure 3: Processor Organisation

The network of processors is running an application where each processor sends messages to all of the other processors at a rate of approximately 300,000 messages per second (you can assume that the messages are evenly distributed and that the number of processors is large enough so that the fact that a processor does not send messages to itself does not affect the answer). The time taken to send each message (without contention) is 1.5 µs.

a. Given that the situation in **Figure 3** resembles the case of a set of processors connected to shared memory:

i. state the expression that gives the probability, p_A, that each message sent by a processor will be accepted at its destination without contending with another message, with an explanation of the terms; **(4)**

ii. calculate the probability for this case. **(2)**

b. In the event of contention, the processor will wait for 1.5 µs and try again. On this basis, calculate the approximate, average time taken to send each message. **(6)**

c. The application is changed such that all of the processors continue to send messages at the same rate but directed (equally) only to processors *Pr0…Pr3*.

i. How does this affect the probability p_A and the average time taken to send each message? **(3)**

ii. What would the consequence of this change be? **(5)**

4. **a.** Identify the important features of a SIMD array processor. Ensure that your explanation identifies how such a processor could be described as SIMD. **(6)**

b. What is the purpose of a *corner-turn buffer*? How is it used? **(4)**

c. An *n x n* array can support the following operations:

```
MOVE  addr1,addr2        ; (addr1) → (addr2)
CSWAP addr1,addr2        ; if (addr1) > (addr2) then (addr1) ↔ (addr2)
BCAST addr               ; (addr) →N,S,E, and W link
IN    x,addr             ; x (=N,S,E, or W) → (addr)
```

An *n x n* array of data, *f*, is distributed across the array (being placed in memory location 0 of each processor). Write a program to calculate the value of *g* in the following equation:

$$g(x,y) = \min_{-1 \le i, j \le 1} f(x+i, y+j)$$

placing each result in memory location 1 of the corresponding processor in the array. **(10)**

NLS / NA