CSC/ECE 506: Architecture of Parallel Computers
Problem Set 1
Due Wednesday, June 9, 2004

Problems 1, 2, and 4 will be graded. There are 50 points on these problems. Note: You must do all the problems, even the non-graded ones. If you do not do some of them, half as many points as they are worth will be subtracted from your score on the graded problems.

Problem 1. (20 points)

Above is a network of IBM 2Pu processor units.
- Each 2Pu unit has two processor cores that communicate across a 64-bit internal bus. Each core shares a 1MB cache and each core operates at 2 GHz with an aggregate operation speed of 4 GHz.
- The 2PUs in the network share DDRII memory operating at 533 Mhz. Shared memory is implemented utilizing a message-passing system.
- The DDRII modules have a 64-bit wide data bus.
- It takes 15ns to set up communication to the memory, and 3 clock cycles to arbitrate for the bus. No data is transferred within the setup time and the arbitration time.
- A page consists of 8192 bits for each memory module, which results in a total page length of 4 x 8192 (32,768 bits).
- Once a page of data has been transferred, the 2Pu has to relinquish the bus.
- The arbiter scheme is round robin.

(a) Processor 1 is requesting one page of data and has gained access to the bus and set up communication to the memory. How long will it take Processor 2 to transfer its first four 64-bit words if it started arbitrating for the bus when Processor 1 had completed its 100th 64-bit transfer?
(b) How much additional time did it take Processor 2 to complete its full-page transfer?
(c) How long did it take the Processor 2 to transfer its entire page from the time it started arbitrating for the bus?
(d) If all five processors wanted to transfer an entire page of data, how long will it take Processor 5 to complete its page transfer? Assume that all processors were attempting arbitration, but that Processor 1 has been granted access to the memory.

Problem 2. (10 points) Consider the following statement from a conventional programming language to convert temperature from Fahrenheit to Celsius.

```c
double celsius(double fahrenheit)
{
    return (fahrenheit - 32.0) * 5.0 / 9.0;
}
```
\[ C = (F - 32) \times \frac{5}{9} \]

(a) Draw a dataflow graph to represent the above statement.

(b) Suppose a subtraction operation takes 1 cycle, and multiplication and division take 3 cycles. How many cycles will be saved by executing the above statement on a dataflow computer with two processors, compared to sequential execution using 1 processor?

(c) The message or token of a dataflow computer contains the name of the operation, receivers (the number of input arcs) and destinations. Would it be necessary to store whether the operand was a left to right or vice versa?

Problem 3. **(20 points)** For the interconnection topology shown below, each link is a 128-bit wide bus operating at 100MHz. There is a communication base cost of 10ns between links.

(a) (i) What is the time to send 512 bytes to an adjacent processor?
    (ii) What is the time to send 512 bytes to a processor three links (hops) away? (assuming no intermediate delays)
    (iii) What is the time to send 512 bytes to a processor on the opposite corner of a 3×3×3 generalized hypercube?

(b) For the shared-memory architecture shown below, there is a communication base cost of 10ns, and the communication network is a 128-bit wide bus operating at 100MHz. What is the transfer time to send 512 bytes to any other processor? (Ignore propagation time differences and assume an idle bus.)
(c) Compare the transfer times for the two architectures when there are three transmissions to be made: (i) Assume in the message-passing model that a processor is communicating with three of its direct neighbors, and (ii) in the shared-memory model all three transmissions go over the common bus. Each transmission transfers 512 bytes.

Problem 4. (20 points) Consider the following C program:

```c
for (i = 1; i <= 1024; i++) {
  sum[i] = 0;
  for (j = 1; j <= i; j++)
    sum[i] = sum[i] + i;
}
```

Assume statements 2 and 4 take two cycles and ignore the overhead due to software loop control (statements 1, 3, and 5). You may ignore and all other system overhead and resource conflicts.

(a) What is the total execution time of the program on a uni-processor system?

Now consider the parallel version of the program to be executed on a shared memory multiprocessor consisting of 32 processors.

(b) Using a simple partitioning scheme where i-loop iterations are divided among 32 processors (i.e. processor 1 executes iterations i = 1 to 32, processor 2 executes iterations i = 33 to 64, and so on), determine the total execution time and the speedup factor compared to the results of part (a). Note that this leads to an unbalanced computational workload among the processors.

(c) Modify the partitioning scheme to facilitate a balanced parallel execution of all the computational workload over 32 processors. What is the minimum execution time and speedup resulting from the balanced parallel execution on 32 processors?

Problem 5. (20 points) Translate the following code for multiplying a rectangular portion of a larger array into the assembly code that was discussed in lecture 3.

```
ADNX          k, i, j
// Adds the contents of index registers i and j, leaving the result in index register k.
SHLNX         j, i, 3
// Shifts index register i left by three bits, effectively multiplying its contents by 8, and leaves this quantity in index register j.
```
procedure Mat_Mult (matrix A, B, C) {
    for (k = 0; k < 768; k += 256) {
        for (i = 0; i < 255; i++) {
            p = i + k;
            C[p, j] = 0, (0 ≤ j ≤ N−1);
        }
    }
    for (j = 0; j < 192; j += 64) {
        q = i + 4j;
        r = j + k;
        for (j = 0; j < 63; j++) {
            s := r + j;
            C[p, k] := C[p, k] + A[q, j] x B[s, k], (0 ≤ k ≤ N−1);
        }
    }
}