DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PhD Comprehensive Examination, Paper II

Total time: 3 Hours **March 10, 2011** **Maximum Marks: 120**

Answer from ALL *the three parts*

Part A: Computer Organisation and Architecture

Answer FOUR *questions in this part*

A.1 i) List out the steps for performing $(-14) \times (-15)$ using any standard 2's complement binary number multiplication algorithm, specifying the algorithm used. 7

ii) Explain why *end correction* is not needed with the Booth's multiplication algorithm. 3

A.2 i) Consider the **`subleq`** instruction, **`a`**, **`b`**, **`c`** being direct memory addresses, defined as:

```
subleq a, b, c  ; Mem[b] = Mem[b] - Mem[a]
                ; if (Mem[b] ≤ 0) goto c
```

Indicate (with justification) whether this instruction can be used to realise the **`SUM a, b`** instruction, so that the content at location **`a`** is added to the content at location **`b`** and the result is stored at location **`b`**. (Hint: use a writable location **`Z`** initially containing 0.) 5

ii) Identify the write-read, write-write and read-write dependencies in the following instruction sequence: 5

```
I1: R1 = 100
I2: R1 = R2 + R4
I3: R2 = R4 - 25
I4: R4 = R1 + R3
I5: R1 = R1 + 30
```

A.3 i) A CPU has two interrupt lines. Assuming that each line has only one interrupt source, describe how the CPU can efficiently service the interrupts, distinguishing between their sources – no need to go into the elementary aspects of interrupt handling. 5

ii) A DMA module is transferring characters to memory using cycle stealing, from a device transmitting at 9600 bps. The processor is fetching instructions at the rate of 1 MIPS. By how much will the processor be slowed down due to the DMA activity? 5

A.4 A disk system is characterised by the following parameters:

t_s: seek time (average time to position head over a track)

r: rotation speed of the disk in rpm

n: number of bits per sector

N: capacity of a track, in bits

t_A: time to access a sector

Develop a formula for t_A in terms of the other parameters. 10

A.5 i) With the help of a diagram, describe the set associative cache mapping scheme. 5

ii) What stages of a typical 5-stage pipeline require accessing the register file? What hazards, if any, are involved – provide non-trivial examples to support your answer? 5

Part B: Operating Systems

Answer ALL *questions in this part*

B.1 In a typical memory organization, there is cache memory and there is also virtual memory mapping. Explain with the help of a schematic diagram, how logical to physical address mapping takes place and the requested memory word gets accessed from cache / main memory. 10

B.2 Consider the following code segment corresponding to a process **P**, which is executing on a machine with demand paging memory management with page size 1 Kbyte.

```
for (i=0;i<1000;i++)
   for (j=0;j<1000;j++)
        A[i][j] = A[i][j] + 1;
```

Assume that each element of the array **A** occupies 4 bytes in memory. If 4 pages are allocated to the process **P** for storing the array **A**, estimate the number of page faults while executing the above code segment assuming the array is stored in memory in column-major order. Assume first-in first-out (FIFO) page replacement strategy. 8

B.3 Consider the non-preemptive version of the Shortest Job Next (SJN) CPU scheduling algorithm. Prove that SJN algorithm results in the shortest average waiting time among all possible non-preemptive scheduling algorithms (Hint: Show that swapping of any two jobs in a SJN schedule increases the waiting time). 10

B.4 i) Suggest an implementation of binary semaphores that avoids busy waiting. 6

ii) Consider an operating system that allows multiple processes in the *Ready* state to reside in main memory at the same time. Suggest suitable memory protection schemes to prevent a process from accidentally accessing the address space of some other process, under the following two scenarios:

i. Each process occupies a contiguous physical memory segment

ii. Demand paging scheme

6

Part C: Programming

Answer ALL *questions in this part*

C.1 i) Define a 'C' datatype **GRAPH** to represent a weighted, undirected graph. You should use an adjacency list to represent the graph and your datatype should be able to accommodate any number of nodes and edges. You may define other intermediate datatypes if you want. 4

ii) Write a 'C' function **void GRAPH *ReadGraph(GRAPH *G)** that takes as parameter the pointer **G** to a variable of type **GRAPH**. It then does the following things in order: (*i*) read in the number of nodes and the number of edges, (*ii*) read in the edges one by one (for each edge read in the **id** of the two end nodes). Allocate any memory that you may need. The function returns the pointer to the graph created. You may assume that if there are n nodes, then the **id** of the nodes are from 0 to $n - 1$. 8

iii) Write a 'C' function `int FindMin(GRAPH G)` that takes as parameter a graph `G`, and returns the minimum weight among all edge weights of `G`. 8

C.2 Write a recursive 'C' function `void Permutation(char *s)` that takes as parameter a null-terminated string `s` and prints out all permutations of the string. Assume that all characters in `s` are distinct. 10

C.3 Write a 'C' program that reads in a sorted array of n integers and another integer k, and checks if there are any two numbers x and y in the array such that $x + y = k$. You should try to reduce the worst case time complexity of your program as much as possible 10