NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING
EXAMINATION FOR
Semester 1 AY2002/2003

CS2271: EMBEDDED SYSTEMS

NOVEMBER 2002 Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This examination paper contains **FOUR (4)** questions and comprises **TWELVE (12)** printed pages, including this page.
2. Answer **ALL** questions.
3. Answer **ALL** questions within the space provided in this booklet.
4. This is an OPEN BOOK examination.
5. Please write your Matriculation Number Below.

MATRICULATION NO: ______________________________

This portion is for examiner's use only

Question	Marks	Remarks
Q1		
Q2		
Q3		
Q4		
Total		

QUESTION 1

A) A 3 x 8 x 3 PLA can implement any logic function that can be implemented by 8 x 3 PROM: True or false? Justify your answer. 2%

B) Explain with an example why "prialt" construct cannot be emulated using "switch" or "par" construct in Handel-C. 4%

C) Following is a Handel-C code fragment using multi-ported memory. Is it legal? In case it is not legal, why not? 2%

```
mpram memstruct{
        rom <unsigned int 2> ro[16];
        ram <unsigned int 2> rw[16];
}tom;
par{ tom.rw[15] = 0; tom.ro[14] =  1; }
```

D) Why is a flash A/D converter faster than successive approximation A/D converter? 4%

E) A hard-disk transfers data in 4-word (each word is 4 bytes) chunks and can transfer at 4MB/sec. This hard disk has to be interfaced with a 500MHz processor. Determine the fraction of CPU time consumed in polling overhead. Assume that you poll often enough so that no data is ever lost and that the devices are always potentially busy. Also assume that the number of clock cycles for a polling operation (including transferring to the polling routine, accessing the device, and restarting the user program) is 400. 4%

F) On I2C bus for a master transmit function, the master outputs data bits to the SDL line. Is it necessary for the master to read the SDL signal after it writes a bit? Explain why or why not. 2%

G) A linker/loader can place code anywhere in memory by taking a set of code pieces, and a table showing all locations in the code where address constants exist. The linker/loader then computes the correct values for those address constants, loads the code into memory and sets the values of the address constants correctly. Suppose that we have a program X that has been linked/loaded and has run for a while. We have preserved all of the tables used by the linker/loader. Can we stop the program and have the linker/loader move it to some other location in memory? If yes, explain how. If no, give as many reasons as you can. 4%

H) Describe a situation where a variable should be declared as volatile. 4%

I) What is the problem in setting a breakpoint if processor has instruction cache? What is the problem if the code is in flash memory? What are the solutions to these problems?

4%

QUESTION 2

4 tasks have the following timings. Assume all the tasks are ready at the beginning.

Task	**Execution Time**	**Deadline**	**Period**
T1	3	5	20
T2	3	7	15
T3	4	10	10
T4	3	20	20

A) Demonstrate that a feasible schedule based on rate monotonic scheduling (RMS) exists for the 4 tasks or show that one cannot exist. 10%

B) Deadline Monotonic Scheduling (DMS) is a fixed priority preemptive scheduling algorithm that is similar to RMS. In this case, priorities assigned to tasks are inversely proportional to the length of the deadline. Thus the task with shortest deadline is assigned the highest priority and the longest deadline task is assigned the lowest priority. Demonstrate that a feasible schedule based on DMS exists for the 4 tasks in the previous page or show that one cannot exist. 10%

C) RMS is optimal fixed priority scheduling algorithm. However, for a set of tasks, it is possible that RMS does not meet all the deadlines, but DMS can. Explain the contradiction. 2%

QUESTION 3

Write ARM assembly language code for the following to calculate fibonacci number. 14%

```
unsigned long fib (int n) {
        if (n <= 1) return 1;
        return fib(n-1) + fib(n-2); }

 void main(void){
    x = fib(10); }
```

[THIS IS A BLANK PAGE]

QUESTION 4

The following is a code fragment from a popular embedded benchmark.

```
q = c[0];
for (i = 0; i < 5; i++) {
   k = c[2*i];
   left = top;
   right = state[i];
   state[i] = bottom;
   top = q * left - k* right;
   bottom = q * right + k* left;
   q = c[2*(i+1)];
 }
```

A) Can you execute all the loop iterations in parallel? Explain your answer. 2%

B) Convert the code fragment to Handel-C so that it executes in minimum number of clock cycles. Exploit as much parallelism as you can but make sure that your code fragment produces the same result as the original code fragment. 12%

[THIS IS A BLANK PAGE]

END OF PAPER