System and Device Programming

Examination Test - Programming Part
04 July 2013

Examination Time: lh 45min. Evaluation. 18 marks.
Textbooks and/or course material allowed.

A dictionary (e.g. from Italian to English) is split among several files Each file contains a section of the complete (much larger) dictionary. The candidate is asked to write a concurrent program to generate a unique dictionary file, appropriately ordered, and without duplications. Specifications are the following.

Each dictionary file starts with the number of translations contained in the file itself, and then stores couple of words, i.e the original one and its (single) translation. The following is a correct dictionary file:

5
casa home
balcone balcony
albero mast
casa house
albero tree

Notice that, in dictionary files, words in the source language have exactly one translation, but they can appear more than once within the same, or different, file(s) (e.g., albero and casa appear twice). Words are C strings, and for each string it is reserved a fixed length of 30 characters (adding enough white spaces). There is no ordering within the files. Files are written in binary form using 8 or 16 bit encoding at choice.

The concurrent program receives four command line arguments:
n sourceDirectory destinationDirectory fileName
where n is an integer, and the other three parameters are strings (of maximum 30 characters) indicating two directories and a file name, respectively.
The program has to run **n+1** threads. The first n threads are *ordering* threads. The last thread is a *merge* thread.

The ordering threads are in charge of ordering all dictionary files included in the **sourceDirectory** directory. For each file the lines of the file have to be sorted in alphabetic ascending order of the first string (only). When the source word is the same, lines have to be merged. For example the previously reported file has to become

albero 2 mast tree
balcone 1 balcony
casa 2 house home

The second field, an integer 32-bit value, indicates the number of translations the source words possesses in the generated file.
The destination file must have the same name of the source file, but has to be created in the **destinationDirectory**. Notice that, the **n** ordering threads have to order all the files included in the **sourceDirectory** without translating twice the same file.

As soon as two ordering threads have done their job, and two files have been created in the **destinationDirectory**, the last thread, i.e., the *merge* thread, has to merge them in a unique file. Again the generated file has to be ordered based on the source language term and the translation of duplicated terms have to be collated together. The merge thread can generate temporary files (with temporary names) to store partial results. The final result (all dictionary words, stored in a single file, ordered in ascending order, without duplication) has to be stored in the current directory with the **fileName** file name.

System and device programming
04 July 2013

(Theory: no textbooks and/or course material allowed)

(15 marks) The final mark is the sum of the I" > 8 and the 2nd part > 10

1. (3.0 marks) Explain the difference between the `canonical, raw` and `cbreak` modes of the terminal line disciplines Which is the system call that allows these modes to he set from a C program?

2. (3.0 marks) Explain the function of the system call **mmap** is and w rite its prototype, and an example of its use.

3. (3.0marks) Given this reference string

11223344441241221443225554 3

Compute the **page fault frequency and the mean resident set** for the working set strategy with control parameter tau = 3.

4. (3.0 marks) Given the architectural scheme of Windows CE represented below

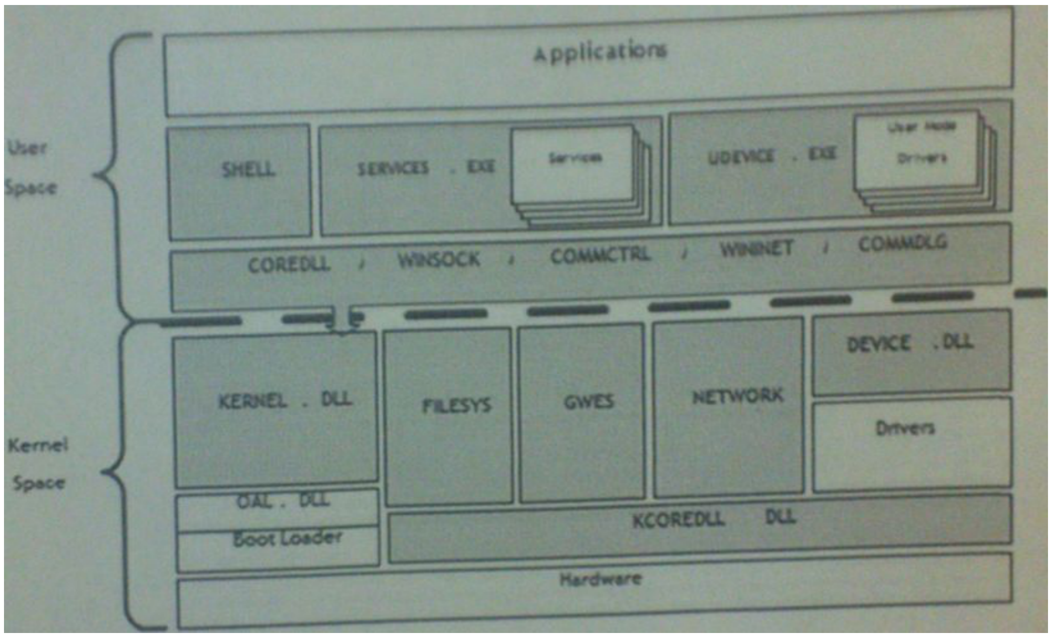

Figure 6.2 Windows Embedded CE 6 0 Architecture

Explain the difference between user-mode drivers and kernel drivers. Why (kernel) drivers are represented with a different colour, with respect to other kernel modules? What is the reflector module within the framework of user mode drivers? Is a user mode application, using a user mode driver, linked (dynamically or statically), with the user mode driver?

5. (3.0 marks) Which are the roles of files pointers and of the overlapped structures in direct file access on WIN32 systems. Briefly describe common aspects and differences. Given a file that contains a sequence of records, corresponding to the C type RECORD_T, write a function:

BOOL ReadRecord (HANDLE hFile, DWORD i, RECORD_T *pRec);

which reads the i-th record (i is the second parameter) in the file, using direct access. hFile is a handle to a file already open. pRec is the pointer to a record where to store the result. Provide two implementations (1) using pointers, (2) using an overlapped structure. The prototype of ReadFile is reported below

```
BOOL ReadFile (HANDLE hFile, LPVOID lpBuffer, DWORD nNumberOfBytesToRead.
                    LPDWORD lpNumberOfBytesRead. LPOVERLAPPED lpOverlapped);
```