Department of Computer Science	**Operating Systems**
VU University Amsterdam	**30.06.2014**

This exam consists of two pages. No calculator and no open books allowed. Concise answers!

1 What is the difference between kernel mode and user mode? Why is the difference important to an operating system? *6pt*

2 MINIX 3 has adopted the client-server model to structure itself. Explain what this model entails and notably what the role of the kernel is. *6pt*

3 What is the relationship between C library calls, system calls, and message passing in a multiserver operating system such as MINIX 3? *6pt*

4 In a system with threads is there normally one stack per thread or one stack per process? Explain. *6pt*

5 Describe the *down* and *up* operations of a semaphore. *6pt*

6 Name two synchronization mechanisms that do not rely on busy waiting. Are there cases in which busy waiting is acceptable? Support your answer with an example. *6pt*

7 Explain what DMA is and why it is used. *6pt*

8 Describe the flow of control from the moment that a device generates an interrupt to the moment that the first instruction of an interrupt handler is executed. *6pt*

9 Describe the steps a device driver needs to perform to write a block of data to the disk. *6pt*

10 A machine has 48-bit virtual addresses and 32-bit physical addresses. Pages are 8 KB. How many entries are needed for the page table? *6pt*

11 What is the purpose of the translation lookaside buffer (TLB)? *6pt*

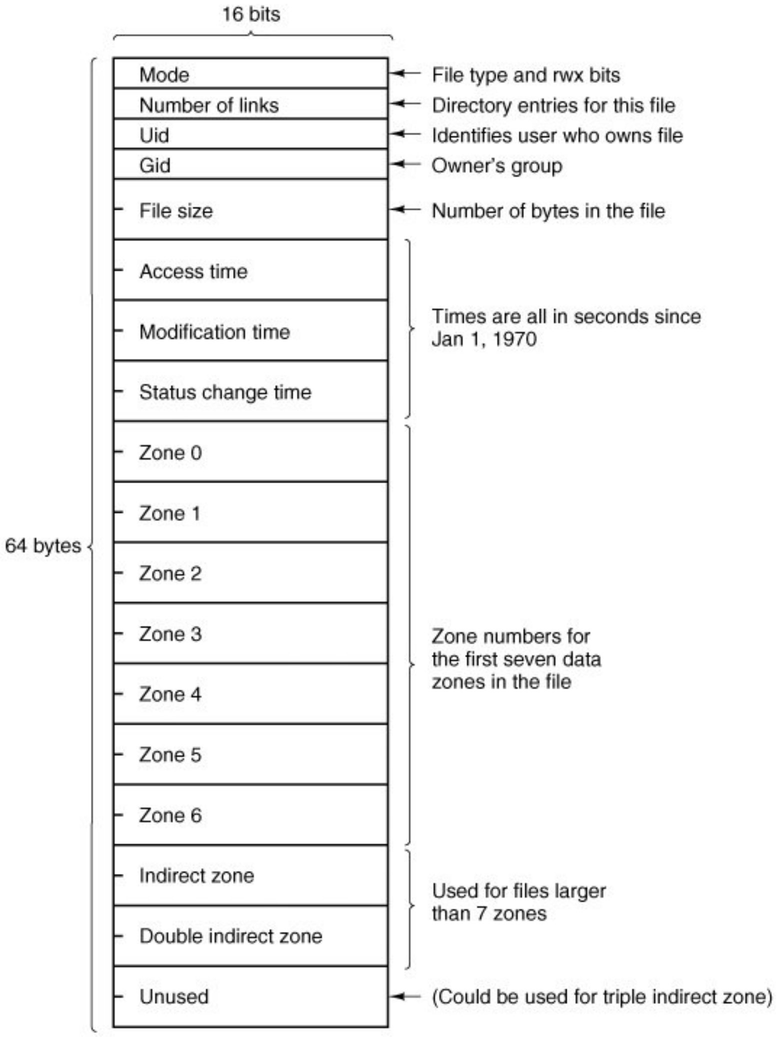
12 Explain what happens after line 4 of the C program below is executed until all the resources associated to the process are freed. *6pt*

```
1  int main(int argc, char **argv)
2  {
3          printf(”Hello,␣world!\n”);
4          return 1;
5  }
```

13 What is the difference between a hard link and a symbolic link? Give an advantage for each one. *6pt*

14 Consider the standard layout of a UNIX file system. Keeping track of available disk blocks and inodes is done through bitmaps. How do you determine the maximum number of files that a file system can handle? *6pt*

15 Describe the disk accesses needed, in the *best* and the *worst* case, for reading a byte from an open file in a UNIX file system. Assume a write-through block cache and the inode structure depicted in the figure below (each zone contains one block). *6pt*


16 bits
64 bytes
Mode
Number of links
Uid
Gid
File size
Access time
Modification time
Status change time
Zone 0
Zone 1
Zone 2
Zone 3
Zone 4
Zone 5
Zone 6
Indirect zone
Double indirect zone
Unused
File type and rwx bits
Directory entries for this file
Identifies user who owns file
Owner's group
Number of bytes in the file
Times are all in seconds since Jan 1, 1970
Zone numbers for the first seven data zones in the file
Used for files larger than 7 zones
(Could be used for triple indirect zone)