University of Karlsruhe
System Architecture Group
Gerd Liefländer

Teaching Assistant:
Philipp Kupferschmied

System Architecture 2008/09
Assignment 13

Question 13.1: Terms and Definitions

1. What kind(s) of fragmentation can occur on systems that support paging?
2. Enumerate and explain the four orthogonal policies that are directly related to paging.
3. Discuss advantages and disadvantages of demand-paging vs. pre-paging.
4. Discuss advantages and disadvantages of local and global page replacement strategies.
5. The swap daemon of some systems tries to always offer a certain amount of free page frames to improve paging. What is the **basic idea** behind such a swap daemon?
6. What is the working set of a task? How does the working set replacement algorithm work?

Question 13.2: Comparison of Replacement Policies

A task has four page frames $(0, \ldots, 3)$ allocated to it. The virtual page number of each page frame, the time of the last loading of a page into each page frame, the time of the last access to the page frame, and the referenced (R) and modified (M) bits of each page frame are shown in the following table.

frame	virtual page	load time	access time	referenced	modified
0	2	60	161	0	1
1	1	130	160	0	0
2	0	26	162	1	0
3	3	20	163	1	1

A pagefault to virtual page 4 occurs. Which page frame will have its contents replaced for each of the following policies? Explain the reason in each case.

1. First-In First-Out (FIFO).
2. Least Recently Used (LRU).
3. Clock policy. Assume that the circular buffer is ordered ascending by load time and that the next-frame pointer refers to frame 3.
4. Not Recently Used (NRU).
5. Optimal algorithm with respect to number of page replacements using the following string for subsequent references: 4, 0, 0, 0, 2, 4, 2, 1, 0, 3, 2.

Question 13.3: FIFO Anomaly

Find a simple example of a system and a reference string showing anomalous behaviors of the FIFO-replacement algorithm, i.e., a behavior where you get worse results for a system with more memory.

Question 13.4: Device Management

1. Discuss the main objectives of an operating system's device management functionality.
2. What possibilities exist for an operating system to communicate with a device?

Question 13.5: I/O Techniques

What is the responsibility of the CPU for device-to-memory data transfers in each of the three primary I/O models?

1. programmed I/O (polling)
2. interrupt-driven I/O
3. DMA

Question 13.6: Device Classes

Operating systems categorize device interfaces to support modular structuring of the kernel code and to reduce the number of (driver) interfaces required. The kernel offers a small set of device abstractions, and each device (driver) must be categorized and associated with one of the abstractions.

1. Why categorize a disk as a block device rather than a character device? What are the consequences of permitting byte-level access to a disk device?
2. What are the reasons for characterizing a serial port as a character device? Would it be possible to treat a serial port as a block device?

Question 13.7: DMA

1. Write pseudo-code which expresses the functionality of the DMA engine for device-to-memory data transfer.
2. Why is it preferable to use a dedicated I/O bus, thus separating the devices from the CPU and memory bus?