TCES 420 - Week 8

Jake McKenzie

November 23, 2018

Homework 7

Section 1 9.18 A certain computer provides its users with a virtual memory space of $2^{3}2$ bytes. The computer has $2^{2}2$ bytes of physical memory. The virtual memory is implemented by paging, and the page size is $4,096$ bytes. A user process generates the virtual address 11123456. Explain how the system establishes the corresponding physical location. Distinguish between software and hardware operations.

Answer: If we convert the virtual address from decimal to hex we obtain 11123456_{10} = 0x00A9BB00. We can obtain the page size of $2^{1}2$ thus the first 12 bits are for displacement within pages and we can deduce that the displacement within the table is 20 bits because the table size is $2^{2}0$. We obtain a 0x00A9B for the pages table displacement and 0xB00 for pages displacement.

9.19 Assume that we have a demand-paged memory. The page table is held in registers. It takes 8 milliseconds to service a page fault if an empty frame is available or if the replaced page is not modified and 20 milliseconds if the replaced page is modified. Memory-access time is 100 nanoseconds. Assume that the page to be replaced is modified 70 percent of the time. What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

```
syms p
us = 10^-6
```

```
us = 1.0000e-06
```

```
ns = 10^-9
```

```
ns = 1.0000e-09
```

```
ms = 10^-3
```

```
ms = 1.0000e-03
```

```
eqn = 200.*ns == (1-p)*100.*ns + 0.3*8.*ms*p+0.7*20.*ms*p
```

eqn =

$$\frac{7555786372591433}{37778931862957161709568} = \frac{163999\,p}{10000000} + \frac{1}{10000000}$$

```
vpa(solve(eqn,p),5)
```

ans = $6.0976\;10^{-6}$

Answer: $p = 6.0976 * 10^{-6}$
When a page fault occurs, the process requesting the page must block while waiting for the page to be brought from disk into physical memory. Assume that there exists a process with five user-level threads and that the mapping of user threads to kernel threads is one to one. If one user thread incurs a page fault while accessing its stack, would the other user threads belonging to the same process also be affected by the page fault—that is, would they also have to wait for the faulting page to be brought into memory? Explain.

Answer: If you have multiple threads, while one thread is blocking which is the case in this example, the other threads in this process will be uneffected by the blocking. Only on that single thread who is blocking is there an negative effect from the page fault.

9.22 The page table shown in Figure 9.32 is for a system with 16-bit virtual and physical addresses and with 4, 096-byte pages. The reference bit is set to 1 when the page has been referenced. Periodically, a thread zeroes out all values of the reference bit. A dash for a page frame indicates the page is not in memory. The page-replacement algorithm is localized LRU, and all numbers are provided in decimal.

a) Convert the following virtual addresses (in hexadecimal) to the equivalent physical addresses. You may provide answers in either hexadecimal or decimal. Also set the reference bit for the appropriate entry in the page table.

Answer:
Virtual Address $\sim$ physical address
0xE12C $\sim$ 0x312C
0x3A9D $\sim$ 0xAA9D
0xA9D9 $\sim$ 0x59D9
0x7001 $\sim$ 0xF001
0xACA1 $\sim$ 0x5CA1

b) Using the above addresses as a guide, provide an example of a logical address (in hexadecimal) that results in a page fault.

Answer: 0x4, 0x8, 0xC and 0xD are all page frames that will result in page faults

c) From what set of page frames will the LRU page-replacement algorithm choose in resolving a page fault?

Answer: I don't know how to do this and I'm completely lost. You also won't even look at this so what even is the point.