PROGRAM 6

Description

The goal of this assignment is to simulate a virtual memory system.

The virtual memory is a sequence of 32-bit words. The physical memory is a shorter sequence of 32-bit words. Virtual memory addresses are 32 bits. Virtual and physical memory is managed in fixed-size pages.

A page table, which is logically in physical memory, maps virtual memory addresses to physical memory addresses. Since physical memory is smaller than virtual memory, the page table might indicate that a particular page is not currently present in physical memory, and that it is instead logically stored on disk.

However, this is a simulation, so all memory words are actually stored in the memory of this program, and disk is not used. Similarly, the page table does not actually exist in the simulated physical memory, and rather exists only in the data structures of the simulation.

A translation lookaside buffer (TLB) is simulated. The TLB stores recent virtual-to-physical address translations.

The goal of the simulation is to report the number of page misses, the number of TLB misses, and the number of disk writes.

The following properties of the virtual memory are set when it is created:

1. the size of the virtual memory, in pages
2. the size of the physical memory, in pages
3. the size of a page in words
4. the number of TLB entries
5. the page replacement algorithm
6. the TLB replacement algorithm

These properties must conform to the following constraints:

- the size of the virtual memory must be larger than the size of the physical memory.
- the size of the physical memory must be greater than zero.
- the size of a page must be a power of two.
- the size of the virtual memory times the size of a page must be less than or equal to 2^32.
- the size of the TLB should be less than or equal to the size of physical memory.
- the size of the TLB must be greater than zero.
- the replacement algorithms are either round-robin replacement or LRU replacement.

A virtual memory system is initialized to have the first K pages of virtual memory loaded into physical memory, where K is the number of pages in the physical memory. Initially the virtual memory system contains arbitrary values (i.e. not necessarily zeros).

The TLB is initialized to have the VM to PM mapping for the first N pages loaded into physical pages (i.e. starting at physical page 0), where N is the number of TLB entries.

A round-robin replacement algorithm will start by replacing the element (physical page or TLB entry) at index 0, then the element at index 1, and so on, wrapping back to 0 when the end of physical memory or the TLB is reached.

LRU replacement is implemented by using a 32-bit timestamp, with time being measured in the number of read/write operations performed since the simulation began. The simulator does not need to worry about the time counter rolling over (i.e. overflowing 32 bits).

When implementing LRU replacement, if two or more pages (or TLB entries) have the same timestamp, then replace the page with the smaller physical page number (or the TLB entry with the smaller index). This can only happen when replacing a page (or TLB entry) that was filled at the simulation startup.

Pages being replaced in memory are only written to disk if they have been modified since they were loaded into memory.

The TLB must be kept in sync with the page table. When a page in physical memory is replaced, its TLB entry, if there is one for it, must be invalidated. Update the TLB entry for the page being replaced to contain the virtual-to-physical mapping for the new page being loaded in. Do not advance the round-robin replacement index in this case, if round-robin replacement is being used for the TLB.

The efficiency of the virtual memory simulation is not critical. What is important is that it accurately count the number of page faults, TLB misses and disk writes.

You can assume the full virtual memory will fit in the memory of your program. That is, you can attempt to malloc space for the full virtual memory and, if malloc fails, you can simply print an error message to stderr and exit the program.

The mandatory interface (set of public functions) for the virtual memory simulation is defined in `~cs520/public/prog6/simVM.h`. (**Do not change this file!**) The documentation in this file specifies more details about the behavior of the simulated virtual memory systems, so be sure to read it carefully. Put your implementation of these functions in `simVM.c`.

In `~cs520/public/prog6/sum.c` is a simple test of the virtual memory system.

Your program will be graded primarily by testing it for correct functionality:

1. Counting page faults using round-robin replacement.
2. Counting TLB misses using round-robin replacement.
3. Counting page faults using LRU replacement.
4. Counting TLB misses using LRU replacement.
5. Counting disk writes.

In addition, however, you may lose points if your program is not properly structured and documented.

Please turn-off any debugging code before you submit your program.

Also be sure you do not include a "main" function in your submitted code.