Betriebssysteme Praktikum Aufgabe 3 Virtuelle Speicherverwaltung

Einleitung

In Labor 3 war die Aufgabe eine virtuelle Speicherverwaltung (VSV) zu simulieren.
Dazu sollten Codefragmente und C-Datei Gerüste zusammengefügt und komplettiert bzw. implementiert werden.
In unserem Konstrukt soll ein sog. *Shared Memory* (anstelle von echtem, physikalischem Speicher) verwaltet werden, in den Daten aus der „pagefile.bin“ eingelesen, oder andersrum ausgelagert werden soll, unter Verwendung verschiedener Seitenersetzungsalgorithmen (FIFO, CLOCK, LRU).

Komponente

Die Hauptkomponenten der VSV sind der Shared Memory (*vmem*), der Memory-Manager (*mmanage*) und die Applikation (*vmappl* + *vmaccess* umfassend). Diese sind auf mehrere Dateien verteilt, die, wie erwähnt, komplettiert werden mussten. Die VSV wird über unser Makefile gebaut, in dem Compiler-Flags zu setzen sind, um den Seitenersetzungsalgorithmus zu wählen und Debug-Meldungen zu aktivieren, bzw. zu deaktivieren.

vmappl.c

Die Hauptanwendung, bestehend aus Funktionen zur Generierung von randomisierten Daten zur Befüllung des Shared Memory’s (SMs).

vmaccess.c

Stellt die Schnittstelle zum virtuellen Speicher dar, die virtuelle in physikalische Adressen übersetzt und ggf. einen *PageFault* auslöst, der an den Memory-Manager gesendet wird.
Die physikalische Adresse berechnet sich wie folgt:

FRAME_INDEX * PAGESIZE + offset

Außerdem setzt sie beim Lesen und Schreiben das *Used-Bit*, zusätzlich beim Schreiben auch das *Dirty-Bit* und verwaltet für den LRU-Algorithmus die Zeitstempel.

mmanage.c

Der Memory-Manager erhält bei einem PageFault, ein Kill-Signal (SIGUSR1) von vmaccess, wodurch er dazu veranlasst wird, die angeforderte Seite aus der Auslagerungsdatei (pagefile.bin) in einen Frame, im physikalischen Speicher zu laden.
Hierbei können drei verschiedene Seitenersetzungsalgorithmen verwendet werden:

FIFO

Beim FIFO-Algorithmus wird, wenn kein freier Frame vorhanden ist, immer der Frame ausgetauscht, der zuerst in den Speicher geschrieben wurde. Hierbei gehen wir folgendermaßen vor:
Wir ersetzen immer den Bereich auf den der Zeiger, der in Admin-Struktur hinterlegt ist, zeigt und schieben diesen Zeiger um eine Position weiter. Ist der Wert des Zeigers größer, als die Anzahl unserer Speicherplätze, wird dieser auf 0 zurückgesetzt.

CLOCK

Beim CLOCK-Algorithmus werden sich der Reihe nach, beginnend bei der in der Admin-Struktur gespeicherten Adresse, die Frames angeschaut. Ist das Used-Bit nicht gesetzt, wird der aktuell selektierte Frame ersetzt. Ist das Used-Bit wiederum gesetzt, wird das Used-Bit zurückgesetzt und der Zeiger auf den nächsten Frame gesetzt. Dieser Vorgang wird wiederholt, bis ein Frame ersetzt wird.

LRU

Zu jedem Frame wird gespeichert, wann der letzte Zugriff auf diesen erfolgte. Sind bei Anforderung einer Seite, keine freien Frames vorhanden, wird der Frame ersetzt, dessen Zugriff am weitesten in der Vergangenheit liegt.
(In unserer Lösung wird bei jedem Zugriff auf einen Frame, eine Zählvariable *g_count* inkrementiert und als Zeitstempel verwendet)

vmem.h

Der Shared Memory stellt den Ersatz zum reellen, physikalischen Hauptspeicher dar, wohin angeforderte Seiten aus der Auslagerungsdatei in Frames (intern das data[]-Array) geladen und bei Entfernung aus dem SM wieder zurückgeschrieben werden. In ihm werden global Informationen zur Steuerung der Verwaltung, sowie die Seitentabelle zur Verfügung gestellt.

```
VMEM_VIRTMEMSIZE        1024   /* Process address space / items */
VMEM_PHYSMEMSIZE         128   /* Physical memory / items */
VMEM_PAGESIZE              8   /* Items per page */
VMEM_NPAGES             (VMEM_VIRTMEMSIZE / VMEM_PAGESIZE)    /* Total number of pages */
VMEM_NFRAMES            (VMEM_PHYSMEMSIZE / VMEM_PAGESIZE)   /* Number of available frames */
VMEM_LASTBMMASK         (~0U << (VMEM_NFRAMES % (sizeof(Bmword) * 8)))
VMEM_BITS_PER_BMWORD    (sizeof(Bmword) * 8)
VMEM_BMSIZE             ((VMEM_NFRAMES - 1) / VMEM_BITS_PER_BMWORD + 1)
SHMKEY                   "./vmem.h"
SHMPROCID               'C'
typedef unsigned int Bmword
```