Instrukcja do laboratorium Systemów Operacyjnych

(semestr drugi)

Ćwiczenie szóste (*dwa zajęcia*)

Temat: Pamięć dzielona

Opracowanie:
mgr inż. Arkadiusz Chrobot
dr inż. Grzegorz Łukawski

Wprowadzenie

1. Pamięć dzielona

Pamięć dzielona (*ang. shared memory*), nazywana również w literaturze polskiej pamięcią wspólną lub pamięcią współdzieloną, służy do szybkiej komunikacji między procesami. Szybkość zyskiwana jest dzięki ograniczeniu do minimum roli jądra w obsłudze wymiany danych, co pozwala zaoszczędzić czas związany z kopiowaniem międzykontekstowym (proces użytkownika – jądro – proces użytkownika). Zasada działania tego środka komunikacji opiera się na pomyśle współdzielenia przez procesy pewnego obszaru pamięci w przestrzeni adresowej użytkownika. Zwykle przestrzenie adresowe procesów są dokładnie odseparowane, tak aby procesy nie mogły wzajemnie uszkodzić się nadpisując sobie dane lub kod. Dzięki odpowiednim wywołaniom systemowym procesy użytkownika mogą zażądać przydzielenia im przez jądro fragmentu pamięci, który będą mogły współdzielić. W takim przypadku system nie kontroluje operacji, które są wykonywane w tej pamięci, a dba jedynie aby żaden z komunikujących się procesów nie próbował odczytywać lub zapisywać pamięci leżącej poza wyznaczonym obszarem. Zaletą takiego rozwiązania jest szybkość działania i łatwość obsługi (procesy mogą korzystać z tej pamięci w taki sam sposób jak z pamięci która jest im przydzielana przez funkcję malloc()), natomiast wadą jest brak synchronizacji komunikacji, o którą musi zadbać programista aplikacji.

2. Funkcje i struktury danych

- funkcja *shmget*() - funkcja ta zwraca identyfikator pamięci dzielonej wygenerowany na podstawie klucza, który może być zwrócony przez funkcję ftok(). Jeśli obszar pamięci dzielonej istniał to zwracany jest tylko jego identyfikator, w przeciwnym razie najpierw jest tworzony obszar pamięci dzielonej o rozmiarze podanym w wywołaniu funkcji[1], a dopiero potem zwracany jest jego identyfikator. Funkcja ta pobiera również flagi związane z prawami dostępu do pamięci dzielonej oraz ze sposobem jej tworzenia i obsługi. Szczegóły: man shmget.

- funkcja *shmat*() - funkcja ta przyłącza obszar pamięci dzielonej określony poprzez podany jej w argumentach identyfikator do przestrzeni adresowej procesu, który ją wywołał. Jako drugi argument jest pobierany przez tę funkcję ad-

1 Dokładniej: o podanym w argumencie wywołania rozmiarze zaokrąglonym do wielokrotności rozmiaru strony.

res początkowy od którego ma być dołączona pamięć dzielona. Jeśli jest on równy NULL, to system wybierze dowolny nieużywany adres, natomiast, jeśli ma on określoną inną wartość, a argument określający flagi ma wartość SHM_RND, to adres pod który będzie przyłączona pamięć dzielona równy będzie adresowi podanemu w argumencie wywołania zaokrąglonemu w dół do wielokrotności wartości stałej SHMLBA (obecnie ta stała jest równa stałej PAGE_SIZE, czyli w przypadku platform sprzętowych Intela jest równa 4K). Oprócz opisanej powyżej wartości argument flag może również przyjąć wartość SHM_RDONLY, która oznacza, że przyłączony obszar pamięci dzielonej będzie przeznaczony tylko do czytania. Jeśli wywołanie funkcji się powiedzie to zwróci ona adres pod którym będzie dołączona pamięć dzielona, w przeciwnym przypadku zwróci wartość -1. Szczegóły: man shmat.

- funkcja *shmdt*() - funkcja ta odłącza obszar pamięci dzielonej o podanym w argumencie adresie od przestrzeni adresowej procesu, który ją wywołał. Jeśli wywołanie zakończy się sukcesem zwracane jest zero, w przeciwnym przypadku -1. Szczegóły: man shmdt.

- funkcja *shmctl*() - funkcja zarządzająca pamięcią dzieloną. Obszar pamięci dzielonej na którym ma zostać wykonana operacja jest określony poprzez identyfikator podany w argumencie wywołania funkcji. Drugim argumentem jest rodzaj operacji: IPC_STAT pozwala na uzyskanie informacji o podanym obszarze, IPC_SET pozwala zmodyfikować prawa dostępu związane z danym obszarem pamięci dzielonej, a IPC_RMID powoduje usunięcie podanego obszaru pamięci dzielonej. Pozostałe operacje albo są dostępne z poziomu użytkownika uprzywilejowanego i tylko w systemie Linux (SHM_LOCK i SHM_UNLOCK), albo mogą zostać w przyszłości zmodyfikowane lub usunięcie (SHM_STAT, SHM_INFO, IPC_INFO). Szczegóły: man shmctl.

Zadania

1. Zademonstruj użycie przez program prywatnego obszaru pamięci dzielonej.
2. Stwórz obszar pamięci dzielonej, z którego będą korzystać trzy procesy. Zorganizuj dostęp do tej pamięci tak, aby procesy miały do niej dostęp w ściśle określonej kolejności. Postaraj się nie używać semaforów.
3. Pokaż rozwiązanie problemu czytelników i pisarzy na przykładzie operacji zapisu i odczytu do pamięci dzielonej. Do synchronizacji komunikacji użyj semaforów. Do wypisywania na ekran komunikatów zamiast funkcji printf() użyj funkcji write()

(man 2 write).

4. Napisz dwa programy, które będą komunikować się poprzez pamięć dzieloną. W trakcie działania programów (np. po dwóch wysłanych i odebranych komunikatach) niech program będący właścicielem pamięci zamieni jej prawa dostępu, tak, aby tylko on mógł z niej korzystać. Pokaż co się wtedy stanie.
5. Zademonstruj działanie flagi SHM_RDONLY.
6. Stwórz trzy procesy, które będą wymieniały między sobą dane poprzez dwa obszary pamięci dzielonej (np. niech proces drugi pośredniczy w wymianie danych pomiędzy pierwszym i drugim obszarem pamięci dzielonej).
7. Użyj flagi IPC_STAT dla funkcji shmctl() celem uzyskania informacji na temat używanego przez proces obszaru pamięci dzielonej.
8. Napisz program, który stworzy dwa spokrewnione procesy, które prześlą między sobą 10 komunikatów przez kolejkę komunikatów, a następnie stworzy dwa kolejne procesy, które prześlą te same komunikaty za pomocą pamięci dzielonej. W obu przypadkach dokonaj pomiaru czasu przesyłania komunikatów i wyświetl otrzymane wyniki. Do pomiaru czasu użyj funkcji time() (man 2 time).

   Uwaga: We wszystkich programach po zakończeniu ich działania wszystkie obszary pamięci dzielonej muszą być usunięte. Jeśli program korzysta z innych zasobów IPC, to one również muszą zostać usunięte.