Politécnico Castelo Branco
Escola Superior de Tecnologia

Escola Superior de Tecnologia
Curso: Licenciatura em Engenharia Informática
2º Semestre do 2º Ano
Unidade Curricular: Sistemas Operativos

Ano Letivo 2024/2025

Ficha de Trabalho nº 5 – Memória Partilhada (Linux)

Índice

1 Objectivos

- Conhecer os mecanismos de gestão e manipulação de segmentos de memória partilhada.
- Compreender a construção de programas multi-processo que envolvam a utilização de segmentos de memória partilhada.

2 Memória Partilhada

A partilha de um segmento (zona ou região) de memória partilhada entre processos é a forma mais eficiente de realizar troca de informação entre eles. Um segmento de memória partilhada é criado no espaço de memória do sistema operativo e pode depois ser utilizado por cada um dos processos como se pertencesse ao seu espaço de memória. Deste modo, os processos podem utilizar uma zona de memória comum para realizar troca de informação. Este mecanismo depende da gestão de memória do sistema operativo, o que implica que as funcionalidades estão directamente associadas com o tipo de arquitectura sobre a qual a implementação é realizada.

2.1 Determinar o estado de um segmento memória partilhada

Tal como foi estudado para os semáforos, os comandos **ipcs** e **ipcrm** são bastante úteis aos programadores durante o desenvolvimento de aplicações que utilizem memória partilhada. Assim, para obter informações relativas aos segmentos de memória partilhada: **ipcs –m**. Utilizando o comando **ipcrm –m ID** é possível remover um segmento de memória partilhada, identificado pelo seu ID, que exista no sistema.

2.2 Conceitos essenciais sobre memória partilhada

Um processo pode criar um segmento de memória partilhada e as respectivas estruturas de controlo. Durante a criação de um segmento de memória partilhada, devem ser definidas as permissões de acesso, o tamanho do segmento (em bytes) e as opções sobre a criação desse segmento. Para consultar e alterar um segmento de memória partilhada, é necessário ter acesso a um identificador desse segmento. Este identificador é fornecido pelo sistema operativo aos processos que pretendam aceder a esse segmento, mediante uma chave associada. Concluída a criação ou o acesso ao segmento de memória partilhada, é possível efectuar duas operações por um processo:

1. Ligar (*attach*) a região de memória partilhada ao espaço virtual de endereçamento do processo invocador
2. Desligar (*detach*) a região de memória partilhada ao espaço virtual de endereçamento do processo invocador

2.3 Criar um segmento de memória partilhada

A função ***shmget()*** permite a criação de um segmento de memória partilhada. O valor retornado pela função é o identificador do segmento de memória partilhada que acaba de ser criado. Este valor será usado por outras funções sempre que quiserem operar sobre este segmento de memória.

A função ***shmget()*** é definida pelo seguinte protótipo:

```
#include <sys/ipc.h>
#include <sys/shm.h>

int shmget(key_t key, size_t size, int shmflg);
```

Os argumentos envolvidos na criação de um segmento de memória partilhada são:

1. ***key*** – chave que ficará associada ao segmento de memória partilhada;
2. ***size*** – tamanho do segmento de memória partilhada em bytes;
3. ***shmflg*** – acessos desejados para o segmento de memória partilhada e a constante **IPC_CREAT** (criar a chave, se não existir). A utilização da constante **IPC_EXCL** irá provocar a falha, no caso da existência de um segmento associado à chave. Por exemplo, o valor **0600** garante apenas ao utilizador as permissões de escrita e leitura da memória partilhada.

```
/* Mode bits for `msgget', `semget', and `shmget'.  */
#define IPC_CREAT       01000           /* Create key if key does not exist. */
#define IPC_EXCL        02000           /* Fail if key exists.  */
#define IPC_NOWAIT      04000           /* Return error on wait.  */
```

A função ***shmget()*** retorna o identificador do segmento de memória partilhada (valor não negativo) em caso de sucesso. Em situação de erro retorna -1 e o *errno* identifica o erro.

Ficheiro **criar_shm.c**:

```
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <errno.h>

#define KEY 123
#define SHM_SIZE 1024

int main()
{
  /* identificador do segmento de memoria partilhada*/
  int shmid;

  /* criar o segmento de memoria partilhada*/
  if((shmid = shmget(KEY, SHM_SIZE, IPC_CREAT|IPC_EXCL|0600)) == -1){
    perror("Erro ao criar zona de memoria partilhada");
    return 1;
  }
  printf("ID do segmento de memoria partilhada: %d\n", shmid);
  printf("Identificado pela chave unica : %d\n", KEY);
  return 0;
}
```

Exercício: Teste o programa.

2.4 Ligar/Desligar segmentos de memória partilhada a processos

2.4.1 Função shmat()

A função ***shmat()*** permite ligar um segmento de memória partilhada ao espaço de endereçamento do processo invocador. A função tem o seguinte protótipo:

```
#include <sys/types.h>
#include <sys/shm.h>

void *shmat(int shmid, const void *shmaddr, int shmflg);
```

Os argumentos da função ***shmat()*** são:

1. ***shmid*** – identificador do segmento de memória partilhada;
2. ***shmaddr*** – identifica o endereço virtual, pertencente ao espaço de endereçamento do processo, onde deve ser feita a correspondência. **Se for 0 o endereço é escolhido pelo sistema operativo.** Se for >0 a correspondência é feita a partir do endereço especificado, se a *shmflg* tiver SHM_RND a ligação ocorre no endereço especificado pelo menor inteiro resultante da divisão de *shmaddr* por SHMLBA, uma constante pré-definida do sistema em <sys/shm.h> associada ao tamanho da página na memória física
3. ***shmflg*** – caso o argumento anterior (*shmaddr*) seja 0, este argumento não é processado e **deverá ter também o valor 0**. Outras opções possíveis da ligação ao segmento de memória partilhada (no caso do argumento anterior não ser 0): SHM_RND (ver ponto anterior); SHM_RDONLY para a ligação ser apenas de leitura.

A função pode retornar vários valores, dependendo da acção realizada. Em caso de erro, retorna -1. O erro é identificado por *errno*.

Quando a função ***shmat*** é invocada, o sistema operativo verifica se existe espaço suficiente no espaço de endereçamento da memória virtual do processo ao qual deve ser ligado o segmento de memória partilhada. Caso seja possível fazer a ligação, o processo ficará com a indexação do endereço de memória do segmento de memória e não com uma cópia do segmento de memória. Se não for possível, será retornado um código de erro.

2.4.2 Função shmdt()

A função ***shmdt()*** é responsável por desligar o segmento de memória partilhada do espaço de endereçamento do processo invocador. A função tem o seguinte protótipo:

```
#include <sys/types.h>
#include <sys/shm.h>

int shmdt(const void *shmaddr);
```

Os argumentos da função ***shmat()*** são:

1. ***shmaddr*** – o segmento a desassociar deve estar neste parâmetro que tem valor igual ao valor retornado pela função ***shmat()***;

A função retorna 0 em caso de sucesso e -1 em caso de erro, onde *errno* indica o erro que ocorreu.

2.4.3 Exemplos

Ficheiro escrever_shm.c:

```
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>

#define KEY 123
#define SHM_SIZE 1024

int main()
{
  /* identificador do segmento de memoria partilhada*/
  int shmid;
  /* ponteiro para a zona de memória comum*/
  char *mem;

  /* aceder ao segmento de memoria partilhada*/
  if((shmid = shmget(KEY, SHM_SIZE, 0)) == -1){
    perror("Erro ao aceder 'a zona de memoria partilhada");
    return 1;
  }
  printf("Sou o processo: %d\n", getpid());
  printf("Vou aceder ao segmento de mem partilhada identificado por: %d\n", shmid);

  /* ligação do processo a zona de memoria
   * obter o pornteiro para o segmento de memoria partilhada*/
  if((mem = shmat(shmid, 0, 0)) == (void*)-1){
    perror("Erro shmat()");
    return 1;
  }

  /* escrever na memoria partilhada */
  strcpy(mem, "SISTEMAS OPERATIVOS");

  printf("MENSAGEM ESCRITA NO SEGMENTO DE MEMORIA PARTILHADA\n");

  /* desligar do segmento de memoria partilhada */
  if(shmdt(mem) == -1){
    perror("Erro shmdt()");
    return 1;
  }

  return 0;
}
```

Exercício: Teste o programa.

Ficheiro ler_shm.c:

```
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <unistd.h>
#include <errno.h>

#define KEY 123
#define SHM_SIZE 1024

int main()
{
  /* identificador do segmento de memoria partilhada*/
  int shmid;
```

```
  /* ponteiro para a zona de memória comum*/
  char *mem;


  /* aceder ao segmento de memoria partilhada*/
  if((shmid = shmget(KEY, SHM_SIZE, 0)) == -1){
    perror("Erro ao aceder a zona de memoria partilhada");
    return 1;
  }

  printf("Sou o processo: %d\n", getpid());
  printf("Vou aceder ao segmento de mem partilhada identificado por: %d\n", shmid);



  /* ligação do processo a zona de memoria
   * obter o pornteiro para o segmento de memoria partilhada*/
  if((mem = shmat(shmid, 0, 0)) == (void*)-1){
    perror("Erro shmat()");
    return 1;
  }

  printf("LEITURA DA MENSAGEM NO SEGMENTO DE MEMORIA PARTILHADA\n");

  /* ler da memoria partilhada */
  printf("%s\n", mem);

  /* desligar do segmento de memoria partilhada */
  if(shmdt(mem) == -1){
    perror("Erro shmdt()");
    return 1;
  }

  return 0;
}
```

Exercício: Teste o programa.

2.5 Remover um segmento de memória partilhada e outros comandos auxiliares

A função *shmctl()* permite realizar comandos auxiliares sobre um segmento de memória partilhada. A função é definida pelo seguinte protótipo:

```
#include <sys/ipc.h>
#include <sys/shm.h>

int shmctl(int shmid, int cmd, struct shmid_ds *buf);
```

Os argumentos da função *shmctl()* são:

1. *shmid* – identificador do segmento de memória partilhada;
2. *cmd* – comando a realizar sobre o segmento de memória partilhda;

```
/* Comandos (cmd) para shmctl */
#define IPC_RMID             /* Remove identifier.  */
#define IPC_SET              /* Set `ipc_perm' options.  */
#define IPC_STAT             /* Get `ipc_perm' options.  */
#define IPC_INFO             /* See ipcs.  */

/* Comandos (cmd) do shmctl:  */
#define SHM_LOCK             /* Prevent swapping of the shared memory segment */
#define SHM_UNLOCK           /* Unlock the segment, allowing it to be swapped out */
```

3. *buf* – estrutura de dados relacionada com o segmento de memória partilhada, caso não seja necessária esta estrutura, pode ser usado o valor NULL neste parâmetro.

A função retorna 0 em caso de sucesso. Em caso de erro, retorna -1. O erro é identificado por *errno*.

3 Exercícios

Construa um programa que crie dois processos filho. Um desses processos deve aceitar sequências de caracteres introduzidas pelo utilizador e enviá-las ao segundo processo. O segundo processo deve esperar o enviado de sequências de caracteres pelo primeiro processo e imprimi-las no ecrã assim que cheguem. O programa deve terminar quando o utilizador digitar a sequência de caracteres *"sair"*.