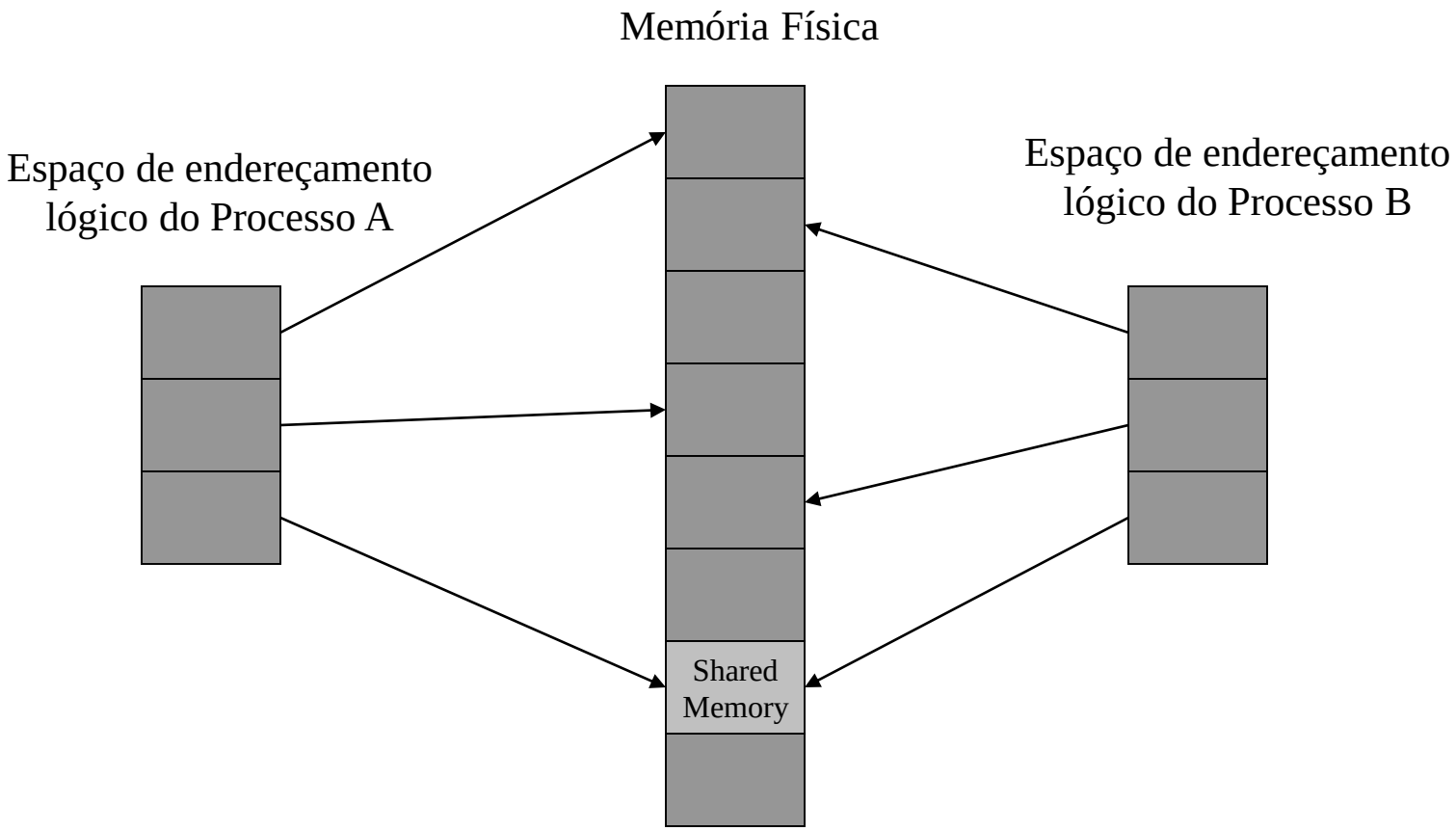
Sistemas Operacionais I

Memória Compartilhada

Memória Compartilhada

- Permite dois ou mais processos compartilhar uma determinada região de memória
- Forma mais simples e rápida de IPC
- Necessário sincronizar o acesso a região por múltiplos processos
  - Geralmente semáforos são usados

Memória Compartilhada



Memória Compartilhada

#include <sys/shm.h>

void * **shmat**(int shm_id, const void *shm_addr, int shmflg);

int **shmctl**(int shm_id, int cmd, struct shmid_ds *buf);

int **shmdt**(const void *shm_addr);

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

Memória Compartilhada

- Procedimento de uso:
  - obtém o identificador da memória compartilhada (shmget)
  - cada processo que deseja utilizar o segmento anexa o mesmo (shmat)
  - após o uso desanexa (shmdt)
  - finalmente um processo desaloca

Memória Compartilhada

- Procedimento de criação:
  - cria o segmento de memória compartilhada (shmget)
  - anexa o segmento para poder (shmat)
  - Após o uso desanexa (shmdt)
  - Finalmente um processo desaloca

Criação de Memória Compartilhada shmget

- shmget obtém um identificador de memória compartilhada:

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

  - Retorna ID se ok, -1 erro
  - key - identificador inteiro não negativo
  - size - tamanho em bytes da área solicitada
  - shmflg - define permissão de acesso inicial e flags de controle de criação(como IPC_CREAT).

Vinculação: shmat

- shmat anexa um segmento de memória compartilhada a um processo:

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

  - retorna ponteiro para segmento se ok, -1 em erro
  - shmid - identifica qual é o segmento
  - shmaddr - se NULL o sistema escolhe endereço não usado; ou especifica endereço
  - shmflg - define alguns flags como SHM_RND

Desvinculação: shmdt

- shmdt dexanexa um segmento de memória compartilhada a um processo:

void *shmdt(const void *shmaddr);

  - retorna 0 se ok, -1 em erro
  - shmaddr - endereço retornado pela operação anterior shmat
  - em caso de sucesso estrutura shmid_ds é atualizada

Controlando e obtendo informações

- shmctl permite executar alguns controles:

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

  - shmid - identifica qual é o segmento
  - cmd é um comando de controle:
    - SEM_LOCK - causa um lock / travamento
    - SEM_UNLOCK - causa um unlock / destrava
    - IPC_STAT - retorna alguns status
    - IPC_SET - altera acesso
    - IPC_RMID - remove o segmento do sistema
  - shmid_ds - estrutura que descreve segmentos

Estrutura shmid_ds

- O kernel mantém uma estrutura com pelo menos os seguintes campos para cada segmento de memória compartilhada:

```
struct shmid_ds {
    struct ipc_perm shm_perm; /* permissões da operação */
    size_t shm_segsz; /* tam. do segmento (bytes)*/
    pid_t shm_cpid; /* pid do criador */
    pid_t shm_lpid; /* pid do último operador */
    shmatt_t shm_nattch;/* no. de anexados atuais */
    time_t shm_atime; /* tempo último attach */
    time_t shm_dtime; /* tempo último detach */
    time_t shm_ctime; /* tempo última mudança*/
...
```

Exemplos

- Comunicação entre dois processos usando memória compartilhada
  - shm1.c
  - shm2.c

Exercícios

1. Digite, compile e execute os programas shm1.c e shm2.c. Verificar o status dos mecanismos de IPC criados usando o comando do shell ipcs
2. Implementar um buffer circular compartilhado entre dois processo. Um processo chamado produtor e outro consumidor. Seguir o modelo do problema do produtor e consumidor apresentado em sala de aula.