SOPER P3 - Blanca Martín y Fernando Villar - P10

Generado por Doxygen 1.8.8

Miércoles, 5 de Abril de 2017 19:12:49

Índice general

Capítulo 1

Índice de clases

1.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

Capítulo 2

Indice de archivos

2.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

Capítulo 3

Documentación de las clases

3.1. Referencia de la Estructura buff

Estructura que se almacenara en la region de memoria compartida.

Atributos públicos

- char buffer [BUF_SIZE]
- unsigned short n_char
- unsigned short limite

3.1.1. Descripción detallada

Estructura que se almacenara en la region de memoria compartida.

Estructura que contiene una cadena de caracteres, y dos numeros que se almacenara en la region de memoria compartida para lectura y escritura de los procesos.

3.1.2. Documentación de los datos miembro

3.1.2.1. char buff::buffer[BUF_SIZE]

3.1.2.2. unsigned short buff::limite

3.1.2.3. unsigned short buff::n_char

La documentación para esta estructura fue generada a partir del siguiente fichero:

- ejercicio6.c

3.2. Referencia de la Estructura info

Estructura que se guardara en la region de memoria compartida.

Atributos públicos

- char nombre [MAX_CHAR]
- int id

3.2.1. Descripción detallada

Estructura que se guardara en la region de memoria compartida.

Estructura que contendra un array con un nombre y un int con un id, y que se almacenara en la memoria compartida.

3.2.2. Documentación de los datos miembro

3.2.2.1. int info::id

3.2.2.2. char info::nombre[MAX_CHAR]

La documentación para esta estructura fue generada a partir del siguiente fichero:

- ejercicio2.c

3.3. Referencia de la Unión semun

Estructura para manejo de semaforos.

`#include <semaforos.h>`

Atributos públicos

- int val
- struct semid_ds * semstat
- unsigned short * array

3.3.1. Descripción detallada

Estructura para manejo de semaforos.

Estructura que contiene aquellos componentes necesarios para que las funciones del sistema destinadas al manejo de los semaforos funcionen correctamente.

3.3.2. Documentación de los datos miembro

3.3.2.1. unsigned short* semun::array

3.3.2.2. struct semid_ds* semun::semstat

3.3.2.3. int semun::val

La documentación para esta unión fue generada a partir del siguiente fichero:

- semaforos.h

Capítulo 4

Documentación de archivos

4.1. Referencia del Archivo ejercicio2.c

Sistemas Operativos: Practica 3, ejercicio 2.

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

Dependencia gráfica adjunta para ejercicio2.c:

Clases

- struct info

  Estructura que se guardara en la region de memoria compartida.

'defines'

- #define MAX_CHAR 80
- #define FILEKEY "/bin/cat"
- #define KEY 1301

Funciones

- void manejador (int sig)

  Manejador para la señal SIGUSR1.
- int num_aleatorio (int inf, int sup)

  Generador de numeros aleatorios en un intervalo.
- int main (int argc, char *argv[])

  Funcion main del ejercicio2.

4.1.1. Descripción detallada

Sistemas Operativos: Practica 3, ejercicio 2.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del segundo ejercicio de la tercera practica, que consiste en la creacion de n hijos que compartiran una region de memoria en la que iran sobrescribiendo los datos y el padre leyendolos.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

07-04-2017

4.1.2. Documentación de los 'defines'

4.1.2.1. #define FILEKEY "/bin/cat"

Fichero para la generacion de la clave

4.1.2.2. #define KEY 1301

Numero para la generacion de la clave

4.1.2.3. #define MAX_CHAR 80

Maximo de caracteres

4.1.3. Documentación de las funciones

4.1.3.1. int main (int *argc,* char * *argv[]*)

Funcion main del ejercicio2.

El programa consiste en la creacion de n procesos secuencialmente que compartan, junto con el padre, una region de memoria. Una vez creados, los procesos pediran un nombre por pantalla, lo guardaran en la region compartida y actualizaran un contador. Cada vez que un hijo termine este proceso, mandara una señal SIGUSR1 al padre, que leera los datos, y terminara.

Parámetros

argc,numero	de argumentos
argv,array	de strings con los argumentos (numero de procesos).

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al introducir los argumentos de entrada, al asignar los manejadores, al ejecutar la funcion fork() o al establecer la region de memoria compartida.

4.1.3.2. void manejador (int *sig*)

Manejador para la señal SIGUSR1.

Este manejador se encarga de evitar la finalizacion del proceso una vez capturada una señal SIGUSR1.

Parámetros

sig,numero	de la señal.

4.1.3.3. int num_aleatorio (int *inf*, int *sup*)

Generador de numeros aleatorios en un intervalo.

num_aleatorio() genera un numero entero aleatorio comprendido entre un minimo y maximo.

Parámetros

inf,entero	limite inferior.
sup,entero	limite superior.

Devuelve

un entero aleatorio comprendido entre inf y sup.

4.2. Referencia del Archivo ejercicio5.c

Sistemas Operativos: Practica 3, ejercicio 5.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/types.h>
#include <sys/shm.h>
#include <errno.h>
#include "../semaforos.h"
```

Dependencia gráfica adjunta para ejercicio5.c:

'defines'

- #define SEMKEY 71458
- #define N_SEMAFOROS 5

Funciones

- int main (void)

  Funcion main del ejercicio5.

4.2.1. Descripción detallada

Sistemas Operativos: Practica 3, ejercicio 5.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del quinto ejercicio de la tercera practica, que consiste en la creacion de un cliente de pruebas para las funciones implementadas en el cuarto ejercicio.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

07-04-2017

4.2.2. Documentación de los 'defines'

4.2.2.1. #define N_SEMAFOROS 5

Numero de semaforos

4.2.2.2. #define SEMKEY 71458

Numero para la clave del semaforo

4.2.3. Documentación de las funciones

4.2.3.1. int main (void)

Funcion main del ejercicio5.

El programa consiste en la creacion de un numero de semaforos y su manipulacion para comprobar la correccion de las funciones implementadas en el ejercicio 4.

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al reservar memoria o al emplear las funciones implementadas sobre semaforos.

4.3. Referencia del Archivo ejercicio6.c

Sistemas Operativos: Practica 3, ejercicio 6.

```
#include <errno.h>
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include "../semaforos.h"
```

Dependencia gráfica adjunta para ejercicio6.c:

Clases

- struct buff

  Estructura que se almacenara en la region de memoria compartida.

'defines'

- #define BUF_SIZE 26
- #define FILEKEY "/bin/cat"

- #define KEY 1301
- #define N_SEMAFOROS 3
- #define SEMKEY 75846

Funciones

- int productor (int semid, struct buff *buffer)

  Productor.
- int consumidor (int semid, struct buff *buffer)

  Consumidor.
- int main (void)

  Funcion main del ejercicio6.

4.3.1. Descripción detallada

Sistemas Operativos: Practica 3, ejercicio 6.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del sexto ejercicio de la tercera practica, que consiste en la implementacion del problema del productor-consumidor.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

07-04-2017

4.3.2. Documentación de los 'defines'

4.3.2.1. #define BUF_SIZE 26

Maximo numero de caracteres

4.3.2.2. #define FILEKEY "/bin/cat"

Fichero para la generacion de la clave

4.3.2.3. #define KEY 1301

Numero para la generacion de la clave de la memoria compartida

4.3.2.4. #define N_SEMAFOROS 3

Numero de semaforos

4.3.2.5. #define SEMKEY 75846

Numero para la generacion de la clave de los semaforos

4.3.3. Documentación de las funciones

4.3.3.1. int consumidor (int *semid,* struct buff * *buffer*)

Consumidor.

Funcion que accede, haciendo uso de semaforos, a la region de memoria compartida y lee de ella los caracteres escritos por el productor y los imprime por pantalla.

Parámetros

semid,identificador	del semaforo.
buffer,estructura	de tipo buff.

Devuelve

OK si todas las operaciones se realizaron con exito, ERROR si se ha producido algun fallo en la manipulacion de los semaforos.

4.3.3.2. int main (void)

Funcion main del ejercicio6.

El programa consiste en la creacion e inicializacion de una serie de semaforos que regularan el acceso a una region de memoria compartida. A esta region accederan dos procesos, uno que escribira en ella y otro que leera.

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al reservar memoria, al crear e inicializar los semaforos, al ejecutar la funcion fork(), al establecer la region de memoria compartida o al desvincularse de la misma.

4.3.3.3. int productor (int *semid,* struct buff * *buffer*)

Productor.

Funcion que accede, haciendo uso de semaforos, a la region de memoria compartida y escribe en ella caracteres generados en orden alfabetico.

Parámetros

semid,identificador	del semaforo.
buffer,estructura	de tipo buff.

Devuelve

OK si todas las operaciones se realizaron con exito, ERROR si se ha producido algun fallo en la manipulacion de los semaforos.

4.4. Referencia del Archivo semaforos.c

Sistemas Operativos: Practica 3, ejercicio 4.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/types.h>
#include <errno.h>
#include <sys/shm.h>
#include "semaforos.h"
```

Dependencia gráfica adjunta para semaforos.c:

Funciones

- int Inicializar_Semaforo (int semid, unsigned short *array)

  Inicializa los semaforos indicados.
- int Borrar_Semaforo (int semid)

  Borra un semaforo.
- int Crear_Semaforo (key_t key, int size, int *semid)

  Crea un semaforo con la clave y el tamaño.
- int Down_Semaforo (int id, int num_sem, short undo)

  Baja el semaforo indicado.
- int DownMultiple_Semaforo (int id, int size, short undo, int *active)

  Baja todos los semaforos del array indicado por active.
- int Up_Semaforo (int id, int num_sem, short undo)

  Sube el semaforo indicado.
- int UpMultiple_Semaforo (int id, int size, short undo, int *active)

  Sube todos los semaforos del array indicado por active.

4.4.1. Descripción detallada

Sistemas Operativos: Practica 3, ejercicio 4.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del cuarto ejercicio de la tercera practica, que consiste en la creacion de una libreria que simplifique el uso de los semaforos.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

07-04-2017

4.4.2. Documentación de las funciones

4.4.2.1. int Borrar_Semaforo (int *semid*)

Borra un semaforo.

Parámetros

int	semid: Identificador del semaforo.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.4.2.2. int Crear_Semaforo (key_t *key,* int *size,* int * *semid*)

Crea un semaforo con la clave y el tamaño.

Parámetros

key_t	key: Clave precompartida del semaforo.
int	size: Tamaño del semaforo.
int	*semid: Nuevo identificador del semaforo.

Devuelve

int: ERROR en caso de error, 0 si ha creado el semaforo, 1 si ya estaba creado.

4.4.2.3. int Down_Semaforo (int *id,* int *num_sem,* short *undo*)

Baja el semaforo indicado.

Parámetros

int	semid: Identificador del semaforo.
int	num_sem: Semaforo dentro del array.
int	undo: Flag de modo persistente pese a finalización abrupta.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.4.2.4. int DownMultiple_Semaforo (int *id,* int *size,* short *undo,* int * *active*)

Baja todos los semaforos del array indicado por active.

Parámetros

int	semid: Identificador del semaforo.
int	size: Numero de semaforos del array.
int	undo: Flag de modo persistente pese a finalización abrupta.
int	*active: Semaforos involucrados.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.4.2.5. int Inicializar_Semaforo (int *semid,* unsigned short * *array*)

Inicializa los semaforos indicados.

Parámetros

int	semid: Identificador del semaforo.
unsigned	short ∗array: Valores iniciales.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.4.2.6. int Up_Semaforo (int *id,* int *num_sem,* short *undo*)

Sube el semaforo indicado.

Parámetros

int	semid: Identificador del semaforo.
int	num_sem: Semaforo dentro del array.
int	undo: Flag de modo persistente pese a finalización abrupta.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.4.2.7. int UpMultiple_Semaforo (int *id,* int *size,* short *undo,* int ∗ *active*)

Sube todos los semaforos del array indicado por active.

Parámetros

int	semid: Identificador del semaforo.
int	size: Numero de semaforos del array.
int	undo: Flag de modo persistente pese a finalización abrupta.
int	∗active: Semaforos involucrados.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.5. Referencia del Archivo semaforos.h

Sistemas Operativos: Practica 3, ejercicio 4.

Gráfico de los archivos que directa o indirectamente incluyen a este archivo:

Clases

- union semun

  Estructura para manejo de semaforos.

'defines'

- #define OK 2
- #define ERROR-1

Funciones

- int Inicializar_Semaforo (int semid, unsigned short *array)

  Inicializa los semaforos indicados.
- int Borrar_Semaforo (int semid)

  Borra un semaforo.
- int Crear_Semaforo (key_t key, int size, int *semid)

  Crea un semaforo con la clave y el tamaño.
- int Down_Semaforo (int id, int num_sem, short undo)

  Baja el semaforo indicado.
- int DownMultiple_Semaforo (int id, int size, short undo, int *active)

  Baja todos los semaforos del array indicado por active.
- int Up_Semaforo (int id, int num_sem, short undo)

  Sube el semaforo indicado.
- int UpMultiple_Semaforo (int id, int size, short undo, int *active)

  Sube todos los semaforos del array indicado por active.

4.5.1. Descripción detallada

Sistemas Operativos: Practica 3, ejercicio 4.

Grupo 2201, Pareja 10. En este modulo se ha implementado la cabecera para la implementacion del ejercicio 4 de la practica.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

07-04-2017

4.5.2. Documentación de los 'defines'

4.5.2.1. #define ERROR-1

Valor para incorrecto retorno de funciones

4.5.2.2. #define OK 2

Valor para correcto retorno de funciones

4.5.3. Documentación de las funciones

4.5.3.1. int Borrar_Semaforo (int *semid*)

Borra un semaforo.

Parámetros

int	semid: Identificador del semaforo.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.5.3.2. int Crear_Semaforo (key_t *key,* int *size,* int * *semid*)

Crea un semaforo con la clave y el tamaño.

Parámetros

key_t	key: Clave precompartida del semaforo.
int	size: Tamaño del semaforo.
int	*semid: Nuevo identificador del semaforo.

Devuelve

int: ERROR en caso de error, 0 si ha creado el semaforo, 1 si ya estaba creado.

4.5.3.3. int Down_Semaforo (int *id,* int *num_sem,* short *undo*)

Baja el semaforo indicado.

Parámetros

int	semid: Identificador del semaforo.
int	num_sem: Semaforo dentro del array.
int	undo: Flag de modo persistente pese a finalización abrupta.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.5.3.4. int DownMultiple_Semaforo (int *id,* int *size,* short *undo,* int * *active*)

Baja todos los semaforos del array indicado por active.

Parámetros

int	semid: Identificador del semaforo.
int	size: Numero de semaforos del array.
int	undo: Flag de modo persistente pese a finalización abrupta.
int	*active: Semaforos involucrados.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.5.3.5. int Inicializar_Semaforo (int *semid,* unsigned short * *array*)

Inicializa los semaforos indicados.

Parámetros

int	semid: Identificador del semaforo.
unsigned	short ∗array: Valores iniciales.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.5.3.6. int Up_Semaforo (int *id,* int *num_sem,* short *undo*)

Sube el semaforo indicado.

Parámetros

int	semid: Identificador del semaforo.
int	num_sem: Semaforo dentro del array.
int	undo: Flag de modo persistente pese a finalización abrupta.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

4.5.3.7. int UpMultiple_Semaforo (int *id,* int *size,* short *undo,* int ∗ *active*)

Sube todos los semaforos del array indicado por active.

Parámetros

int	semid: Identificador del semaforo.
int	size: Numero de semaforos del array.
int	undo: Flag de modo persistente pese a finalización abrupta.
int	∗active: Semaforos involucrados.

Devuelve

int: OK si todo fue correcto, ERROR en caso de error.

Índice alfabético