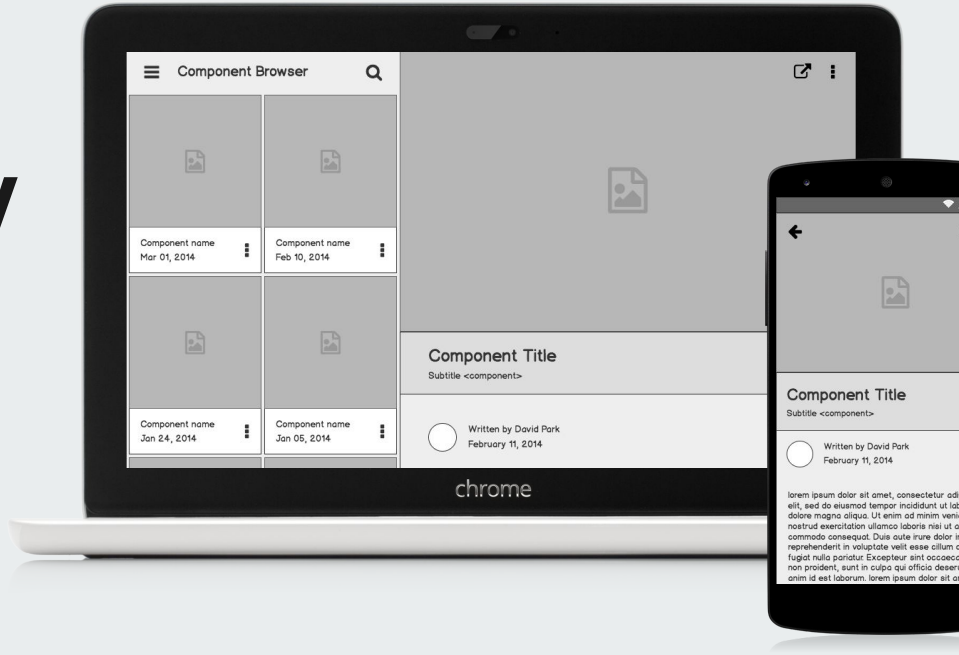
IPC Using Shared Memory

With C program that illustrates two processes communicating using Shared memory



Outline

- Introduction
- What is Inter-Process Communication?
- What is Shared Memory ?
- Functions Used
- How does IPC Using Shared Memory work ?
- Algorithms
- C Programs

Introduction

Hello and welcome to this presentation about Inter-process communication in Operating Systems. In this presentation, I will explain the basic concepts of Inter-process communication, Approaches to Interprocess Communication and Inter-process communication using Shared memory. Also, i will explain C programs with Algorithms that illustrates two processes communicating using Shared memory.

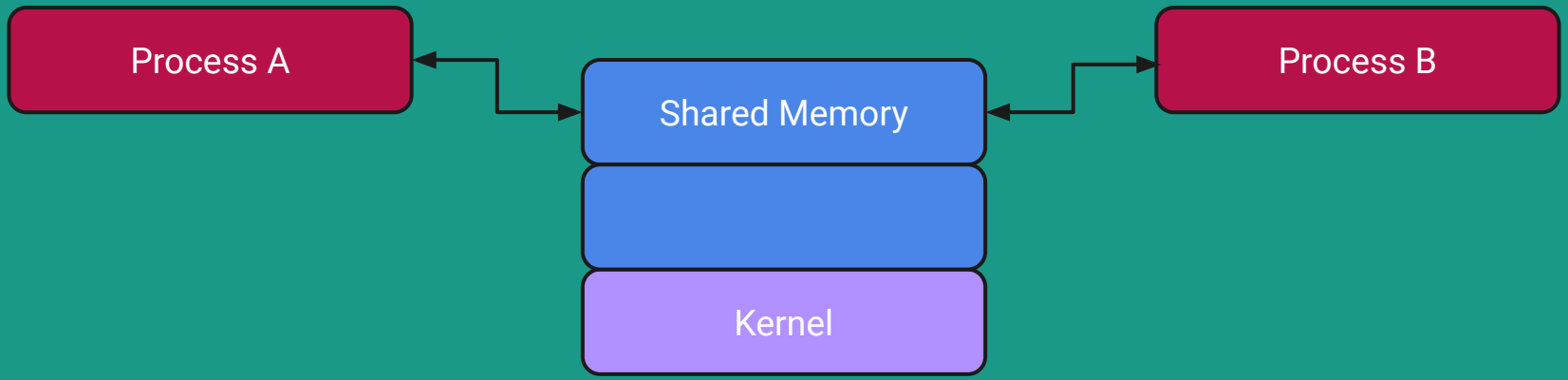
By the end of this presentation, you will have a better understanding of Inter-process communication and C program for Inter-process communication using Shared memory.

What is Inter-Process Communication?

Inter process communication in OS is the way by which multiple processes can communicate with each other. Shared memory in OS, message queues, FIFO, etc. are some of the ways to achieve IPC in OS.

What is Shared Memory ?

Shared memory is a memory shared between two or more processes.



Functions Used

Let us look at a few details of the system calls related to shared memory.

shmget() Function

It is used to create the shared memory segment.

Syntax :

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

#include <sys/shm.h>

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

shmat() Function

It is used to attach the created shared memory segment associated with the shared memory identifier specified by **shmid** to the calling process's address space.

Syntax :

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

#include <sys/shm.h>

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

shmdt() Function

When you're done with the shared memory segment, your program should detach itself from it using shmdt().

Syntax :

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

#include <sys/shm.h>

int shmdt(shmat());
```

shmctl() Function

When you detach from shared memory,it is not destroyed. So, to destroy shmctl() is used.

Syntax :

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

#include <sys/shm.h

shmctl(int shmid, IPC_RMID, NULL);
```

How does IPC Using Shared Memory work?

It can be either an **arbitrary value** or one that can be derived from the library function ftok().

This function is used to **create the shared memory** segment.

It is the **identifier of the shared memory.**

This function is used to **attach the shared memory** segment.

Pointer to the shared memory segment.

Algorithms

Let us take a look at the Algorithms to implement Shared memory.

Algorithm for Writer Process

1. **Create** a shared memory segment or use an already created shared memory segment using **shmget()**.

2. **shmat()** function is used to attach the created shared memory segment associated with the shared memory identifier specified by **shmid**.

3. Then, ask user to **write** some content into the shared memory segment.

4. After writing, the writer process **copy** written data into the shared memory.

5. **Exit** the program.

Algorithm for Reader Process

1. Using **shmget(),** attach itself to the shared memory created by writer Process.

2. Then, **reads** the content of the shared memory.

3. It will Detach itself from the already attached shared memory using **shmdt()**.

4. So, to **destroy** Shared memory shmctl() is used.

5. **Exit** the program.

C Programs

C Program for Writer Process :

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{
//Pointer
void *shared_memory;
char buff[100];
//Variable for Shared memory ID
int shmid;
```

```
shmid=shmget((key_t)2345, 1024, 0666|IPC_CREAT);
/*creates shared memory segment with key 2345, having size 1024 bytes.
IPC_CREAT is used to create the shared segment if it does not exist. 0666 are
the permissions on the shared segment*/

printf("Key of shared memory is %d\n", shmid);
//Printing   Shared memory ID

shared_memory=shmat(shmid,NULL,0);
//process attached to shared memory segment

printf("Process attached at %p\n",shared_memory);

//Printing the address where the segment is attached with this process
```

```
printf("Enter some data to write to shared memory : \n");
read(0,buff,100);
//get some input from user
strcpy(shared_memory,buff);
//data written to shared memory
printf("You wrote : %s\n",(char *)shared_memory);
shmdt(shared_memory);
}
```

Input

Input from the user :

Hello World!

Output

Key of shared memory is 0

Process attached at 0x7ffe040fb000

Enter some data to write to shared memory :

You wrote : Hello World!

C Program for Reader Process :

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{
//Pointer
void *shared_memory;
char buff[100];
//Variable for Shared memory ID
int shmid;
```

```
shmid=shmget((key_t)2345, 1024, 0666);
printf("Key of shared memory is %d\n", shmid);
//Printing   Shared memory ID

shared_memory=shmat(shmid,NULL,0);
//process attached to shared memory segment

printf("Process attached at %p\n",shared_memory);
printf("Data read from shared memory is : %s\n",(char *)shared_memory);
//Printing Data from Shared Memory
shmdt(shared_memory);

shmctl(shmid, IPC_RMID, NULL);

}
```

Output

Key of shared memory is 0

Process attached at 0x7f76b4292000

Data read from shared memory is : Hello World!

Questions?

References

Tutorialspoint

Javatpoint

geeksforgeeks

Thank You.