**Name:Rohan Nivas Chavan**

**PRN:2019BTEIT00084**

**UOS LAB**

**3.5** Write a program to lock the file using flock system call

**Objectives:**

1. To learn about File locking-mandatory and advisory locking.

**Theory:**

1.Name:-

flock - apply or remove an advisory lock on an open file

2. Syntax:-

#include <sys/file.h>

int flock(int fd, int operation);

3.Description:-

Apply or remove an advisory lock on the open file specified by fd. The argument operation is one of the following:

1.LOCK\_SH

Place a shared lock. More than one process may hold a shared lock for a given file at a given time.

2.LOCK\_EX

Place an exclusive lock. Only one process may hold an exclusive lock for a given file at a given time.

3.LOCK\_UN

Remove an existing lock held by this process.

A call to flock() may block if an incompatible lock is held by another process. To make a nonblocking request, include LOCK\_NB (by ORing) with any of the above operations.

A single file may not simultaneously have both shared and exclusive locks.

4.Return Value:-

On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

**Data Dictionary:**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr Number | Variable/Function | Datatype | Use |
|  |  |  |  |
| 1 | fd | int | Open a file and store file descriptor. |
|  |  |  |  |

Table 3.5 Data Dictonary

**Program:**

#include <fcntl.h>

#include <stdio.h>

#include <string.h>

#include <unistd.h>

int main (int argc, char\* argv[])

{

char\* file = argv[1];

int fd;

struct flock lock;

printf ("opening %s\n", file);

/\* Open a file descriptor to the file. \*/

fd = open (file, O\_WRONLY);

printf ("locking\n");

/\* Initialize the flock structure. \*/

memset (&lock, 0, sizeof(lock));

lock.l\_type = F\_WRLCK;

/\* Place a write lock on the file. \*/

fcntl (fd, F\_SETLKW, &lock);

printf ("locked; hit Enter to unlock... ");

/\* Wait for the user to hit Enter. \*/

getchar ();

printf ("unlocking\n");

/\* Release the lock. \*/

lock.l\_type = F\_UNLCK;

fcntl (fd, F\_SETLKW, &lock);

close (fd);

return 0;

}

**Conclusion:**

1.Functions of flock() to use shared, exclusive locks and unlock them are studied.

**Refrences :**

www.tutorialspoint.com/unix\_system\_calls

OUTPUT:

