**3.6** write a program to lock file using fnctl system call

**Objectives:**

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

**Theory:**

1.The fcntl system call is the access point for several advanced operations on file descriptors.

2.The first argument to fcntl is an open file descriptor, and the second is a value that

indicates which operation is to be performed. For some operations, fcntl takes an additional argument.

3.The fcntl system call allows a program to place a read lock or a write lock on a file.

4.A read lock is placed on a readable file descriptor, and a write lock is placed on a

writable file descriptor. More than one process may hold a read lock on the same file at the same time, but only one process may hold a write lock, and the same file

may not be both locked for read and locked for write. Note that placing a lock does not actually

prevent other processes from opening the file, reading from it, or writing to it, unless they acquire locks with fcntl as well.

5.To place a lock on a file, first create and zero out a struct flock variable. Set the l\_type field of the structure to F\_RDLCK for a read lock or F\_WRLCK for a write lock. Then call fcntl, passing a file descriptor to the file, the F\_SETLCKW operation

code, and a pointer to the struct flock variable. If another process holds a lock that prevents a new lock from being acquired, fcntl blocks until that lock is released.

6.It returns -1 if lock is not acquired on a file.

**Data Dictionary:**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr Number | Variable/Function | Datatype | Use |
|  |  |  |  |
| 1 | file | char\* | File name. |
|  |  |  |  |
| 2 | fd | int | File descriptor. |
|  |  |  |  |
| 3 | lock | struct flock | Structure to describe a file lock. |
|  |  |  |  |

Table 3.6 Data Dictonary

**Program:**

#include <fcntl.h>

#include <stdio.h>

#include <string.h>

#include <unistd.h>

#include <sys/file.h>

int main()

{

int fd = open("File1.txt", O\_WRONLY);

if(fd==-1)

printf("open file fails\n");

}

else

{

if(flock(fd, LOCK\_EX)==0)

printf("Lock is acquired on file\n");

else

printf("Lock is not acquired on a file\n");

}

close(fd);

return 0;

}

**Conclusion:**

1.Funcltions of fcntl() to use read lock, write lock are studied.

**References:**

[1] https://gavv.github.io/blog/file-locks/