In this question we have maded a module and compiled it using the makefile and its commands such that it may show us the desired outputs of the struct task and there respective informations sucha as pid,gid,command line and uid and etc if any else is required

To implement a kernel system call as a module, you will need to follow these steps:

Define the system call function. This function should have the prototype long sys\_my\_syscall(int pid). The function should return 0 on success and a negative error code on failure.

Define the system call wrapper function. This function should have the prototype asmlinkage long my\_syscall(int pid). It should simply invoke the system call function and return the result.

Add the system call wrapper function to the system call table. You can do this by modifying the sys\_call\_table array in the sys\_call\_table.h header file.

Implement the system call function. Inside the function, you can use the find\_task\_by\_pid() function to get a pointer to the task\_struct of the process with the given PID. Then, you can use the pid, uid, pgid, and comm fields of the task\_struct to get the values you want to print.

Write the module initialization and cleanup functions. The initialization function should register the system call using the register\_syscall() function. The cleanup function should unregister the system call using the unregister\_syscall() function.