National University of Computer & Emerging Sciences
(FAST-NUCES)

Operating Systems Project Report

Project Name: Dining Philosophers Problem

Group Members:

Ali Nayab Nathani (19K-0301)
Tuaha Ajaz(19K-0257)
Hamamd Jabbar(19K-1367)

Course: Operating Systems

Coding Platform: gedit & nano for user level code & terminal for compilation

Operating System: Ubuntu

Version: 18.04

Dining Philosopher's Problem

1. Project Description

In computer science, the dining philosopher's problem is an example problem often used in concurrent algorithm design to illustrate synchronization issues and techniques for resolving them. The dining-philosophers problem is considered a classic synchronization problem neither because of its practical importance nor because computer scientists dislike philosophers but because it is an example of a large class of concurrency-control problems. It was originally formulated in 1965 by Edsger Dijkstra as a student exam exercise, presented in terms of computers competing for access to tape drive peripherals. Soon after, Tony Hoare gave the problem its present formulation

2. Problem Statement

Five silent philosophers sit at a round table around a bowl of spaghetti. Chopsticks are placed between each pair of adjacent philosophers. Each philosopher must alternately think and eat. However, a philosopher can only eat spaghetti when they have both left and right chopsticks. Each chopstick can be held by only one philosopher and so a philosopher can use the chopstick only if it is not being used by another philosopher. After an individual philosopher finishes eating, they need to put down both forks so that the forks become available to others. A philosopher can take the fork on their right or the one on their left as they become available, but cannot start eating before getting both chopsticks. Eating is not limited by the remaining amounts of spaghetti or stomach space; an infinite supply and an infinite demand are assumed. The problem is how to design a discipline of behavior (a concurrent algorithm) such that no philosopher will starve; i.e., each can forever continue to alternate between eating and thinking, assuming that no philosopher can know when others may want to eat or think.

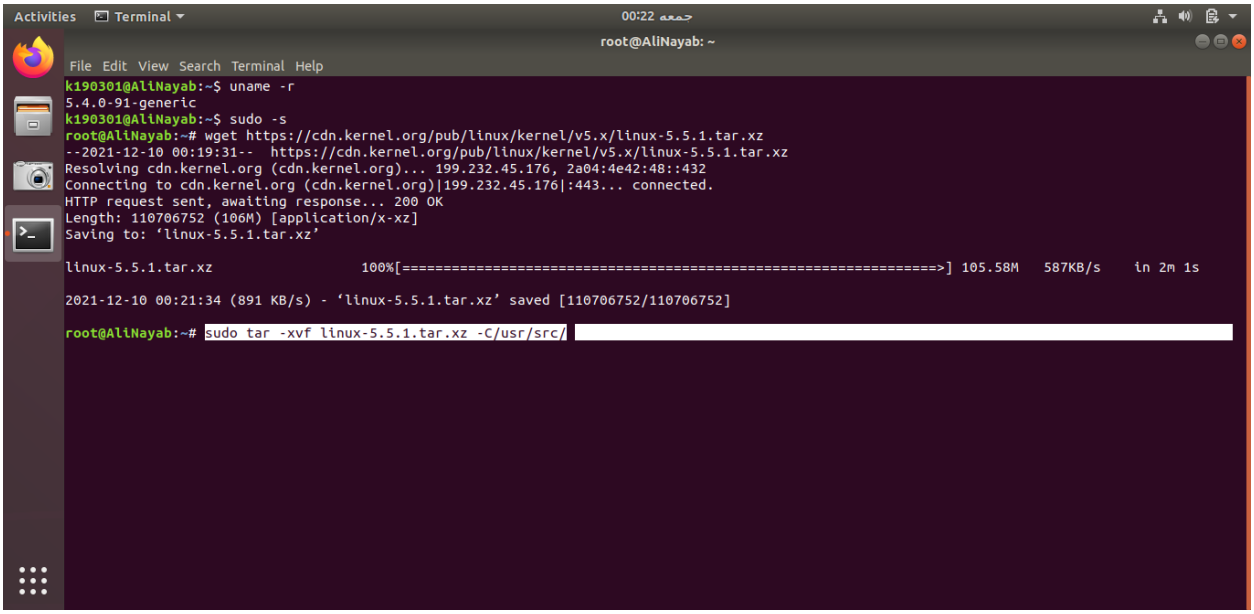
3. Operating System Implementation

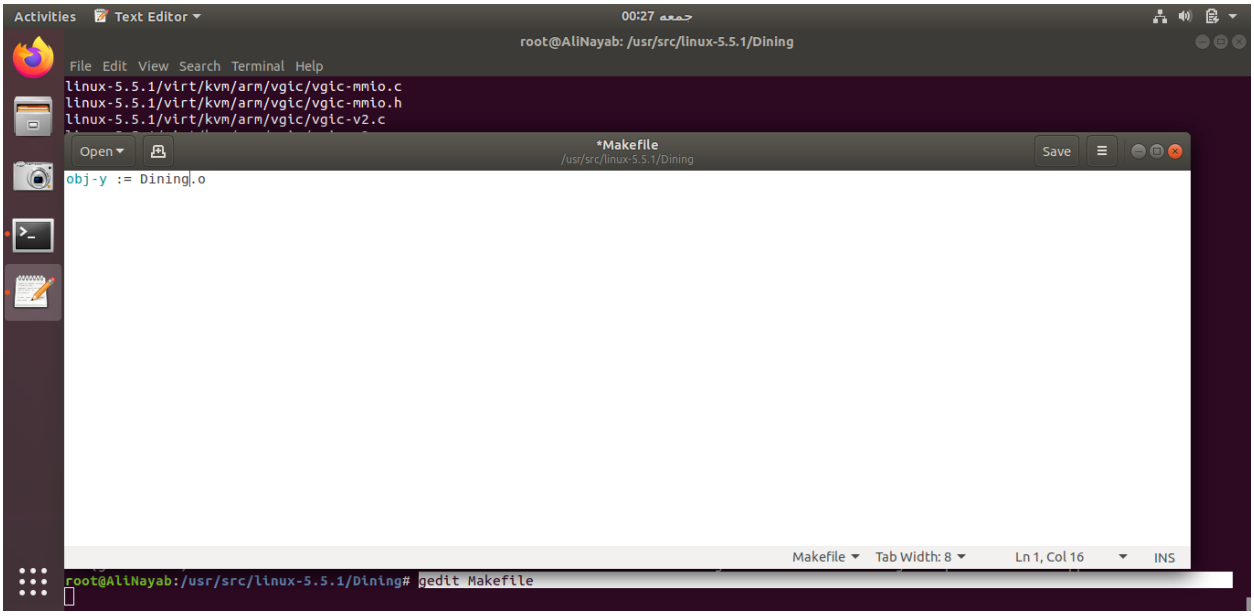
We will utilize Semaphore and Threads to exhibit and take care of the issue. One straightforward arrangement is to address every chopstick with a semaphore. A scholar attempts to get a chopstick by executing a wait () procedure on that semaphore. At the point when philosopher delivers his chopsticks by executing the signal () procedure on the fitting semaphores, our code will be depend on kernel level and there is semaphore

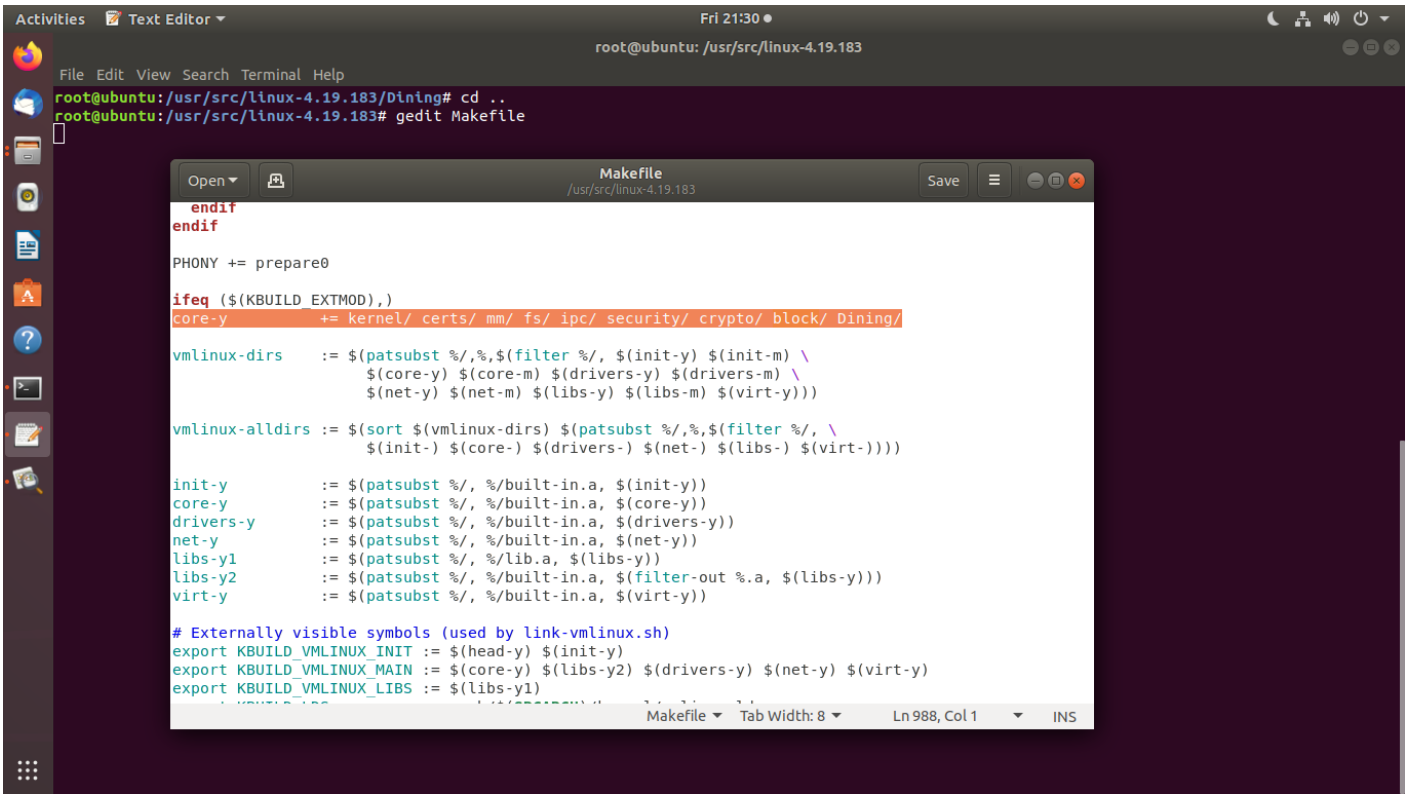
4. Project Result

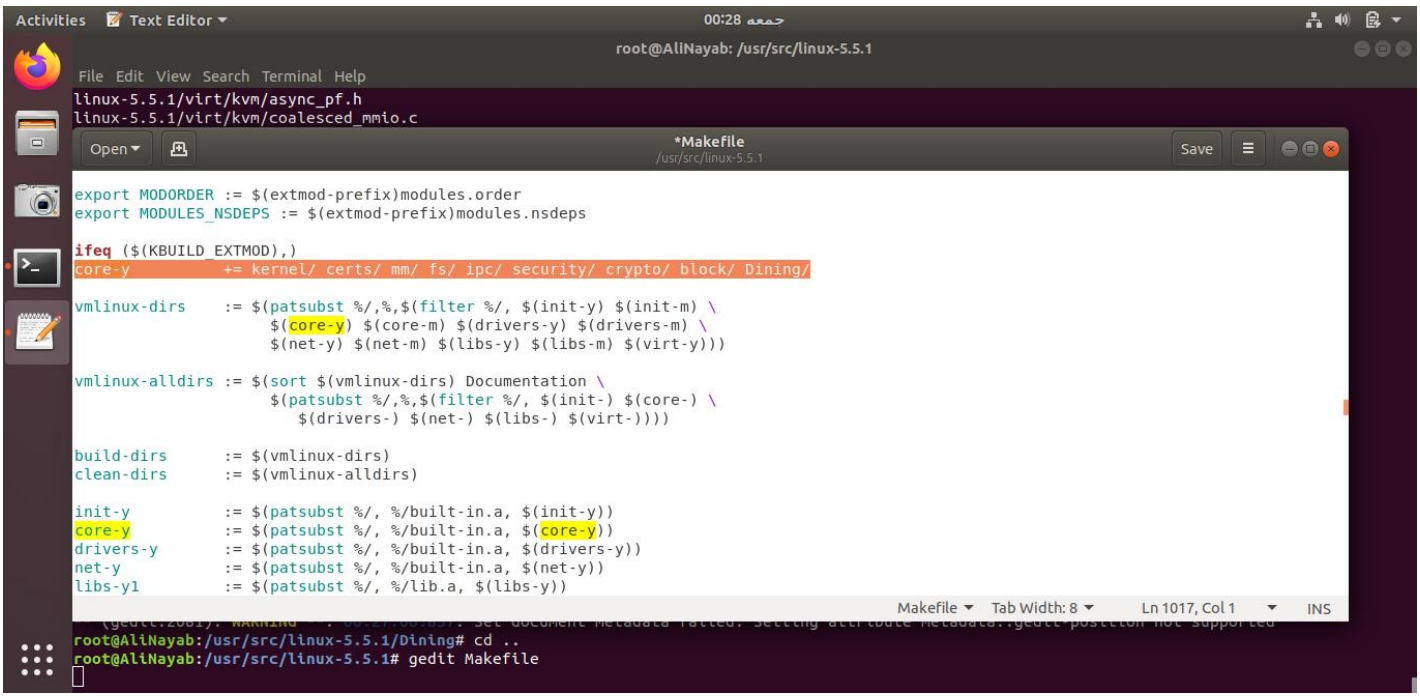
Getting rid of deadlock by properly synchronizing. Meeting need to allocate several resources among several processes in a deadlock-free and starvation-free manner. The main Objective of this project was to learn the use of semaphore and to understand the concept of deadlock and way to resolve this problem.

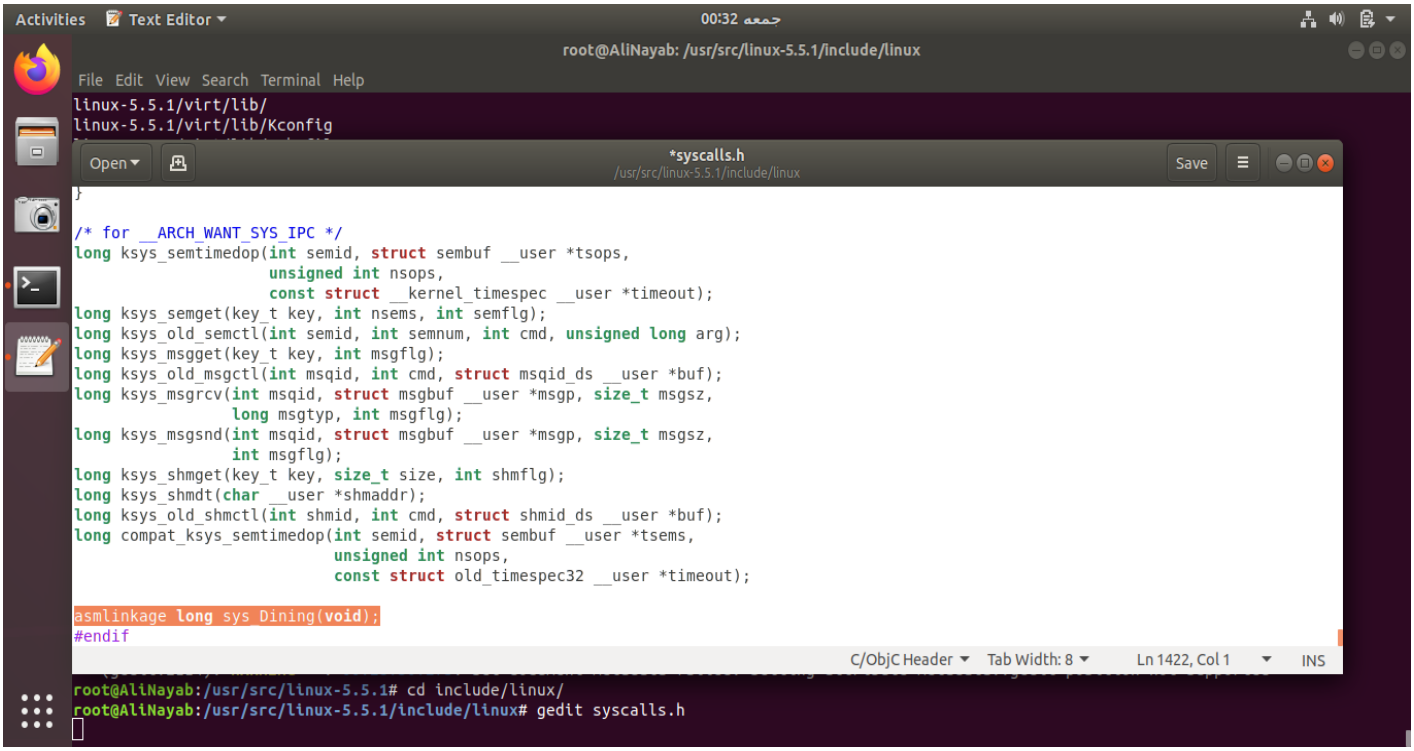
5. Project Screenshots

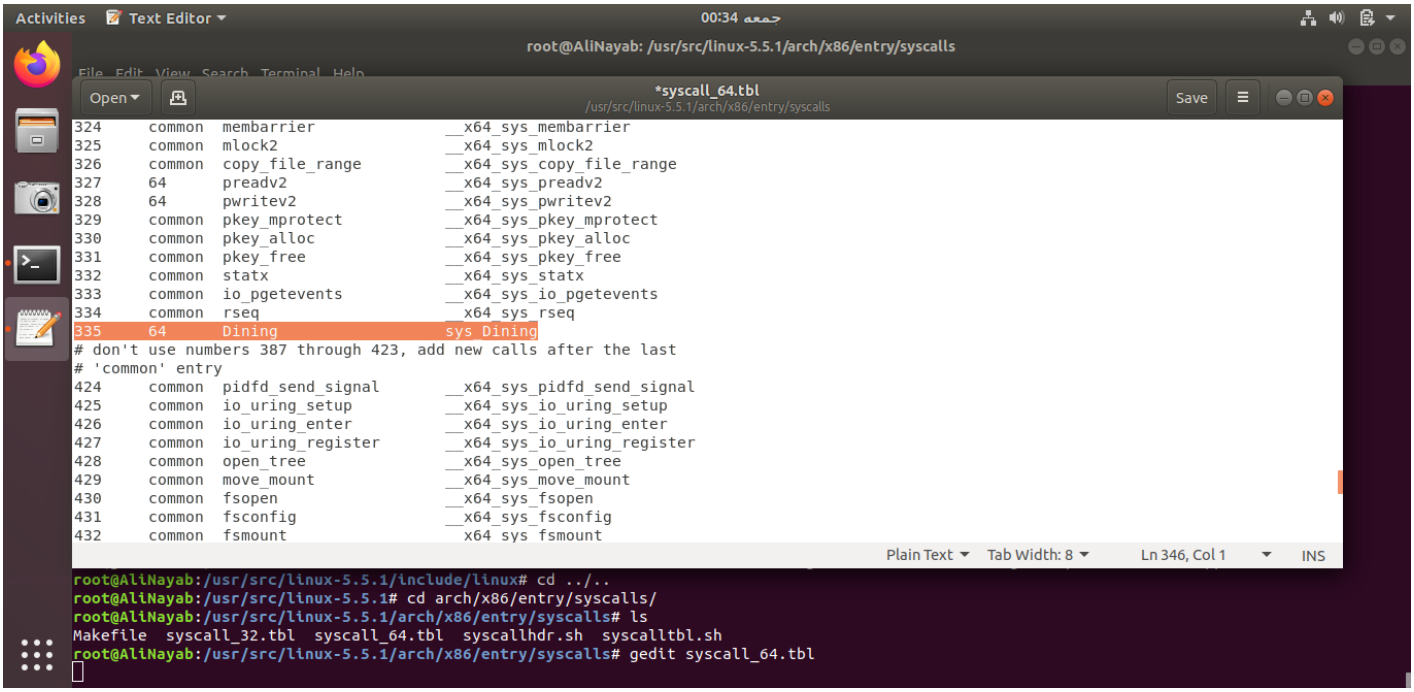


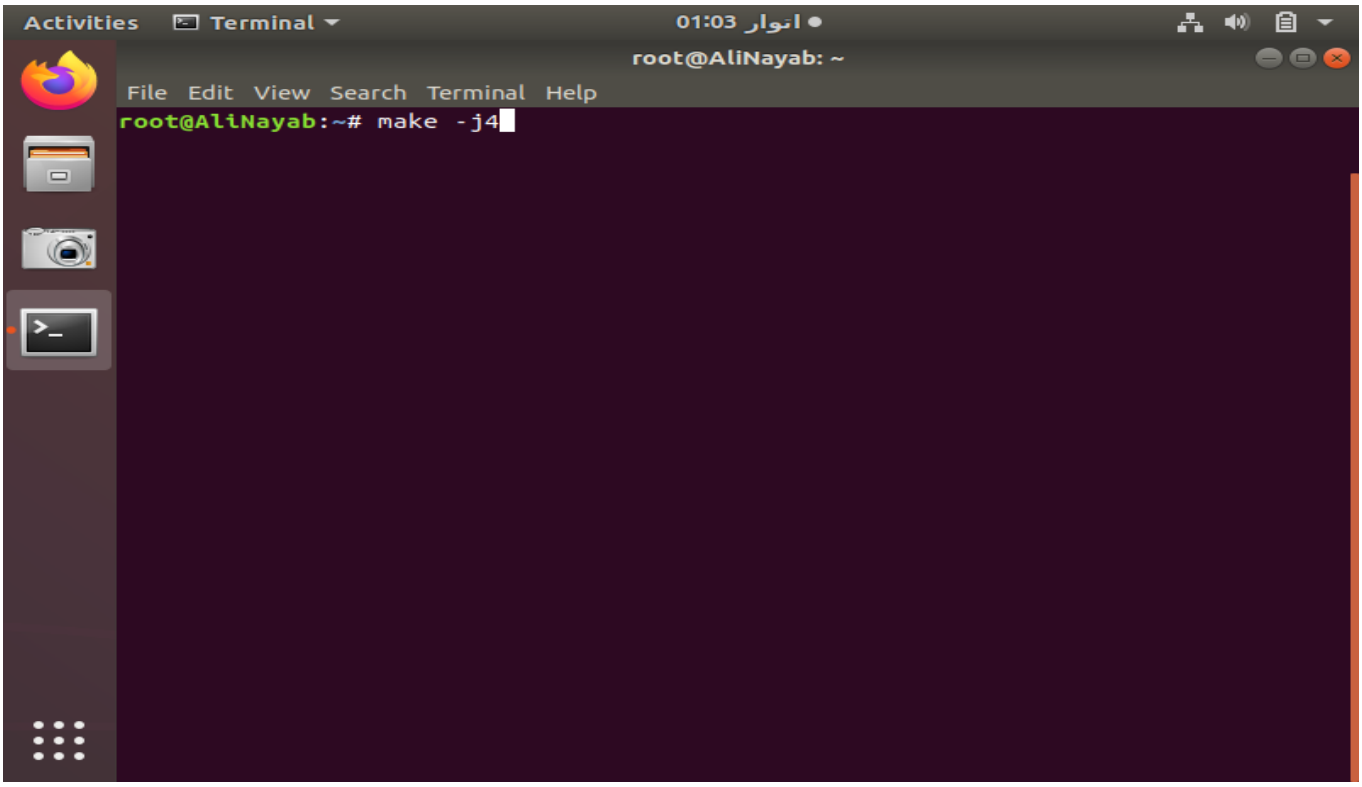



Activities
Text Editor
Fri 21:30
root@ubuntu: /usr/src/linux-4.19.183
File Edit View Search Terminal Help
root@ubuntu:/usr/src/linux-4.19.183/Dining# cd ..
root@ubuntu:/usr/src/linux-4.19.183# gedit Makefile
Open
Makefile
/usr/src/linux-4.19.183
Save
endif
endif
PHONY += prepare0
ifeq ($(KBUILD_EXTMOD),)
core-y += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ Dining/
vmlinux-dirs := $(patsubst %/,%,$(filter %/, $(init-y) $(init-m) \
$(core-y) $(core-m) $(drivers-y) $(drivers-m) \
$(net-y) $(net-m) $(libs-y) $(libs-m) $(virt-y)))
vmlinux-alldirs := $(sort $(vmlinux-dirs) $(patsubst %/,%,$(filter %/, \
$(init-) $(core-) $(drivers-) $(net-) $(libs-) $(virt-))))
init-y := $(patsubst %/, %/built-in.a, $(init-y))
core-y := $(patsubst %/, %/built-in.a, $(core-y))
drivers-y := $(patsubst %/, %/built-in.a, $(drivers-y))
net-y := $(patsubst %/, %/built-in.a, $(net-y))
libs-y1 := $(patsubst %/, %/lib.a, $(libs-y))
libs-y2 := $(patsubst %/, %/built-in.a, $(filter-out %.a, $(libs-y)))
virt-y := $(patsubst %/, %/built-in.a, $(virt-y))
Externally visible symbols (used by link-vmlinux.sh)
export KBUILD_VMLINUX_INIT := $(head-y) $(init-y)
export KBUILD_VMLINUX_MAIN := $(core-y) $(libs-y2) $(drivers-y) $(net-y) $(virt-y)
export KBUILD_VMLINUX_LIBS := $(libs-y1)
Makefile
Tab Width: 8
Ln 988, Col 1
INS

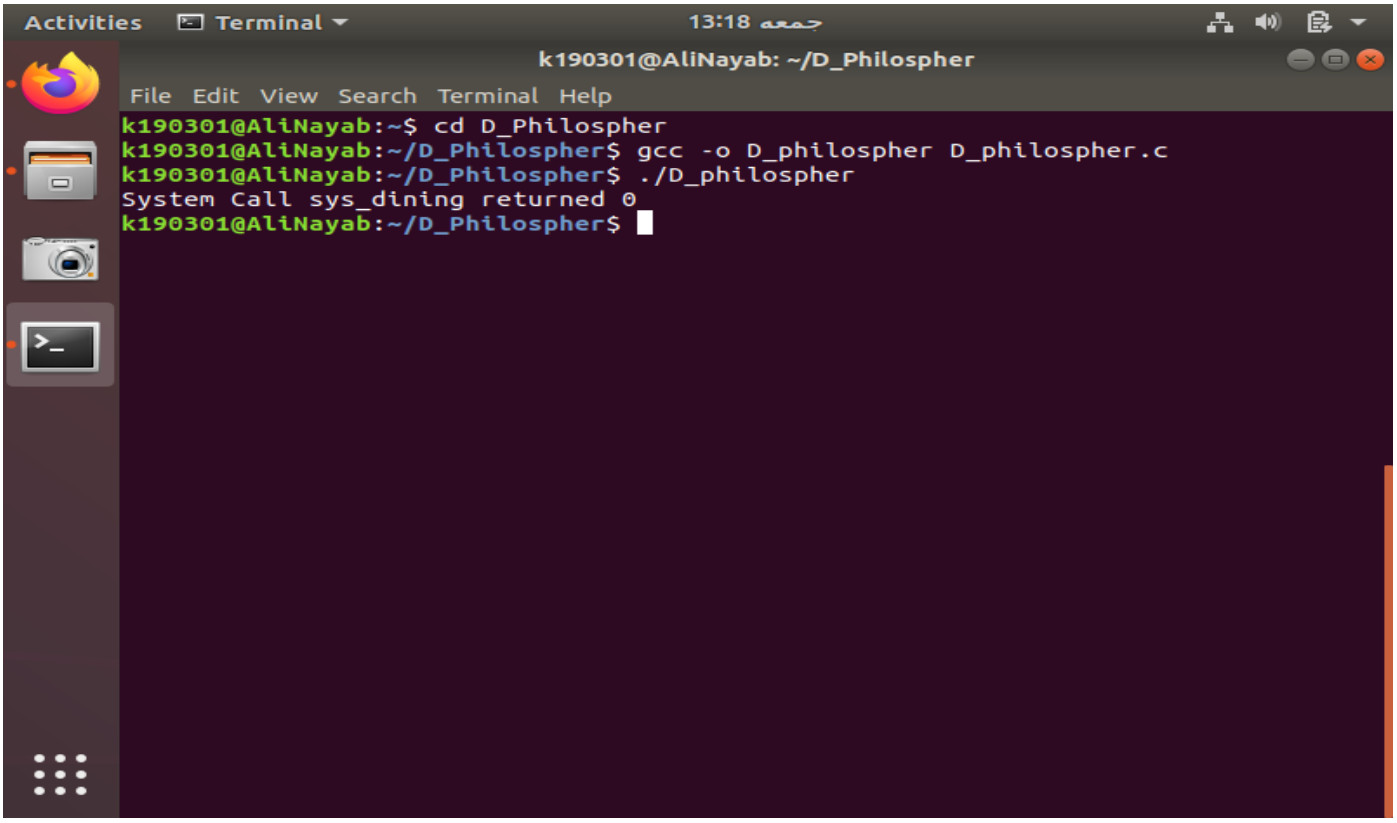

Activities
Text Editor
00:28 جمعه
root@AliNayab: /usr/src/linux-5.5.1
File Edit View Search Terminal Help
linux-5.5.1/virt/kvm/async_pf.h
linux-5.5.1/virt/kvm/coalesced_mmio.c
Open
*Makefile
/usr/src/linux-5.5.1
Save
export MODORDER := $(extmod-prefix)modules.order
export MODULES_NSDEPS := $(extmod-prefix)modules.nsdeps
ifeq ($(KBUILD_EXTMOD),)
core-y += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ Dining/
vmlinux-dirs := $(patsubst %/,%,$(filter %/, $(init-y) $(init-m) \
$(core-y) $(core-m) $(drivers-y) $(drivers-m) \
$(net-y) $(net-m) $(libs-y) $(libs-m) $(virt-y)))
vmlinux-alldirs := $(sort $(vmlinux-dirs) Documentation \
$(patsubst %/,%,$(filter %/, $(init-) $(core-) \
$(drivers-) $(net-) $(libs-) $(virt-))))
build-dirs := $(vmlinux-dirs)
clean-dirs := $(vmlinux-alldirs)
init-y := $(patsubst %/, %/built-in.a, $(init-y))
core-y := $(patsubst %/, %/built-in.a, $(core-y))
drivers-y := $(patsubst %/, %/built-in.a, $(drivers-y))
net-y := $(patsubst %/, %/built-in.a, $(net-y))
libs-y1 := $(patsubst %/, %/lib.a, $(libs-y))
Makefile
Tab Width: 8
Ln 1017, Col 1
INS
root@AliNayab:/usr/src/linux-5.5.1/Dining# cd ..
root@AliNayab:/usr/src/linux-5.5.1# gedit Makefile

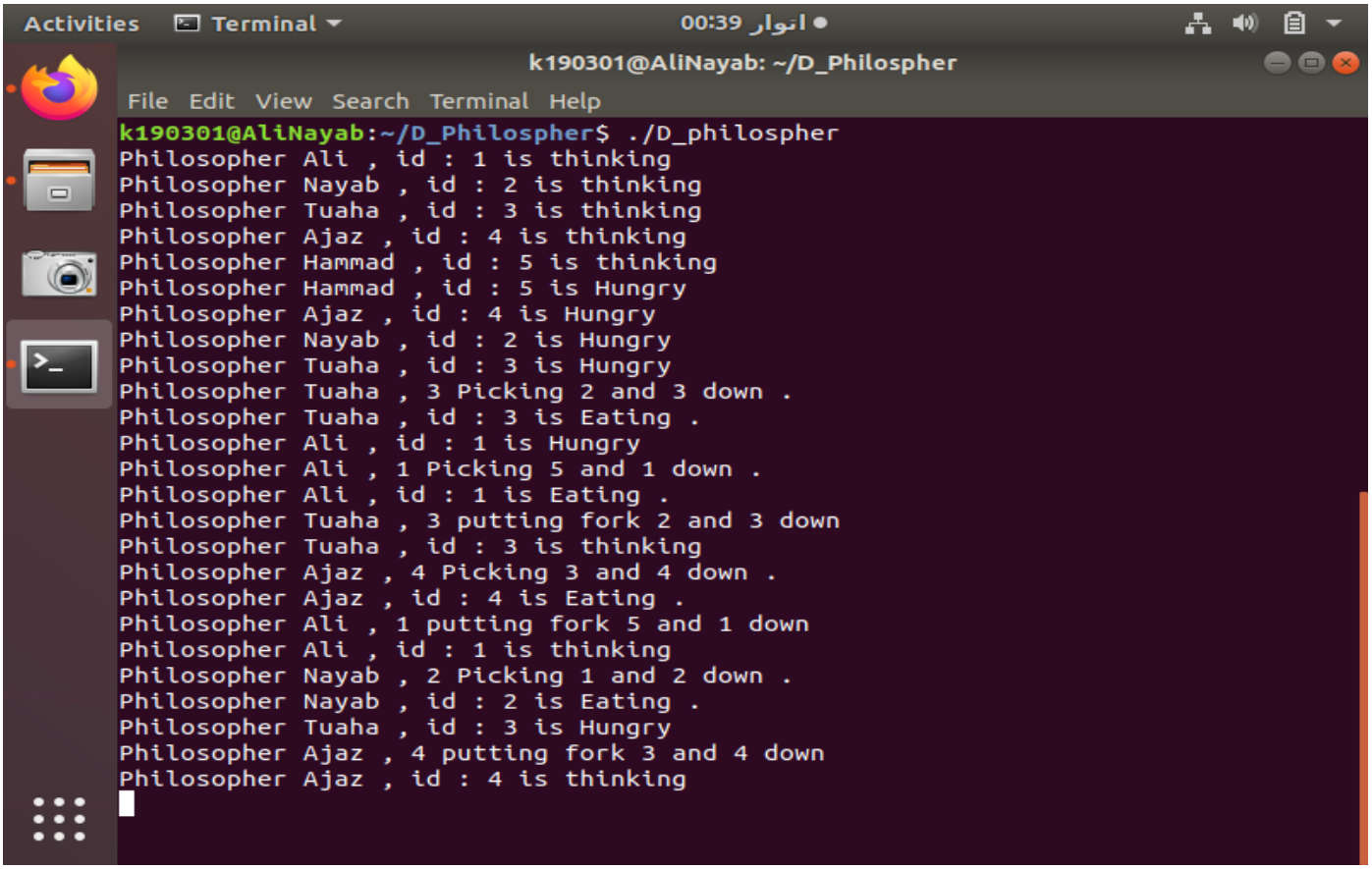

Activities
Text Editor
00:32 جمعه
root@AliNayab: /usr/src/linux-5.5.1/include/linux
File Edit View Search Terminal Help
linux-5.5.1/virt/lib/
linux-5.5.1/virt/lib/Kconfig
Open
*syscalls.h
/usr/src/linux-5.5.1/include/linux
Save
}
/* for __ARCH_WANT_SYS_IPC */
long ksys_semtimedop(int semid, struct sembuf __user *tsops,
unsigned int nsops,
const struct __kernel_timespec __user *timeout);
long ksys_semget(key_t key, int nsems, int semflg);
long ksys_old_semctl(int semid, int semnum, int cmd, unsigned long arg);
long ksys_msgget(key_t key, int msgflg);
long ksys_old_msgctl(int msqid, int cmd, struct msqid_ds __user *buf);
long ksys_msgrcv(int msqid, struct msgbuf __user *msgp, size_t msgsz,
long msgtyp, int msgflg);
long ksys_msgsnd(int msqid, struct msgbuf __user *msgp, size_t msgsz,
int msgflg);
long ksys_shmget(key_t key, size_t size, int shmflg);
long ksys_shmdt(char __user *shmaddr);
long ksys_old_shmctl(int shmid, int cmd, struct shmid_ds __user *buf);
long compat_ksys_semtimedop(int semid, struct sembuf __user *tsems,
unsigned int nsops,
const struct old_timespec32 __user *timeout);
asmlinkage long sys_Dining(void);
#endif
C/ObjC Header
Tab Width: 8
Ln 1422, Col 1
INS
root@AliNayab:/usr/src/linux-5.5.1# cd include/linux/
root@AliNayab:/usr/src/linux-5.5.1/include/linux# gedit syscalls.h


Activities
Text Editor
00:34 جمعه
root@AliNayab: /usr/src/linux-5.5.1/arch/x86/entry/syscalls
Open
*syscall_64.tbl
/usr/src/linux-5.5.1/arch/x86/entry/syscalls
Save
324 common membarrier __x64_sys_membarrier
325 common mlock2 __x64_sys_mlock2
326 common copy_file_range __x64_sys_copy_file_range
327 64 preadv2 __x64_sys_preadv2
328 64 pwritev2 __x64_sys_pwritev2
329 common pkey_mprotect __x64_sys_pkey_mprotect
330 common pkey_alloc __x64_sys_pkey_alloc
331 common pkey_free __x64_sys_pkey_free
332 common statx __x64_sys_statx
333 common io_pgetevents __x64_sys_io_pgetevents
334 common rseq __x64_sys_rseq
335 64 Dining sys_Dining
don't use numbers 387 through 423, add new calls after the last
'common' entry
424 common pidfd_send_signal __x64_sys_pidfd_send_signal
425 common io_uring_setup __x64_sys_io_uring_setup
426 common io_uring_enter __x64_sys_io_uring_enter
427 common io_uring_register __x64_sys_io_uring_register
428 common open_tree __x64_sys_open_tree
429 common move_mount __x64_sys_move_mount
430 common fsopen __x64_sys_fsopen
431 common fsconfig __x64_sys_fsconfig
432 common fsmount x64 sys fsmount
Plain Text
Tab Width: 8
Ln 346, Col 1
INS
root@AliNayab:/usr/src/linux-5.5.1/include/linux# cd ../..
root@AliNayab:/usr/src/linux-5.5.1# cd arch/x86/entry/syscalls/
root@AliNayab:/usr/src/linux-5.5.1/arch/x86/entry/syscalls# ls
Makefile syscall_32.tbl syscall_64.tbl syscallhdr.sh syscalltbl.sh
root@AliNayab:/usr/src/linux-5.5.1/arch/x86/entry/syscalls# gedit syscall_64.tbl


Activities
Terminal
01:03 انوار
root@AliNayab: ~
File Edit View Search Terminal Help
root@AliNayab:~# make -j4


Activities
Text Editor
Fri 23:27
Open
D_philospher.c
~/Desktop/D_philospher
Save
#include <stdio.h>
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <unistd.h>
int main()
{
long int amma = syscall(335);
printf("System Call sys_dining returned %ld\n", amma);
return 0;
}
Saving file "/home/ziyaanali/Desktop/D_philospher/D_philospher.c"...
C
Tab Width: 8
Ln 10, Col 18
INS


Activities
Terminal
13:18 جمعه
k190301@AliNayab: ~/D_Philospher
File Edit View Search Terminal Help
k190301@AliNayab:~$ cd D_Philospher
k190301@AliNayab:~/D_Philospher$ gcc -o D_philospher D_philospher.c
k190301@AliNayab:~/D_Philospher$ ./D_philospher
System Call sys_dining returned 0
k190301@AliNayab:~/D_Philospher$


Activities
Terminal
00:39 انوار
k190301@AliNayab: ~/D_Philospher
File Edit View Search Terminal Help
k190301@AliNayab:~/D_Philospher$ ./D_philospher
Philosopher Ali , id : 1 is thinking
Philosopher Nayab , id : 2 is thinking
Philosopher Tuaha , id : 3 is thinking
Philosopher Ajaz , id : 4 is thinking
Philosopher Hammad , id : 5 is thinking
Philosopher Hammad , id : 5 is Hungry
Philosopher Ajaz , id : 4 is Hungry
Philosopher Nayab , id : 2 is Hungry
Philosopher Tuaha , id : 3 is Hungry
Philosopher Tuaha , 3 Picking 2 and 3 down .
Philosopher Tuaha , id : 3 is Eating .
Philosopher Ali , id : 1 is Hungry
Philosopher Ali , 1 Picking 5 and 1 down .
Philosopher Ali , id : 1 is Eating .
Philosopher Tuaha , 3 putting fork 2 and 3 down
Philosopher Tuaha , id : 3 is thinking
Philosopher Ajaz , 4 Picking 3 and 4 down .
Philosopher Ajaz , id : 4 is Eating .
Philosopher Ali , 1 putting fork 5 and 1 down
Philosopher Ali , id : 1 is thinking
Philosopher Nayab , 2 Picking 1 and 2 down .
Philosopher Nayab , id : 2 is Eating .
Philosopher Tuaha , id : 3 is Hungry
Philosopher Ajaz , 4 putting fork 3 and 4 down
Philosopher Ajaz , id : 4 is thinking

6. Code File:


Dining.c