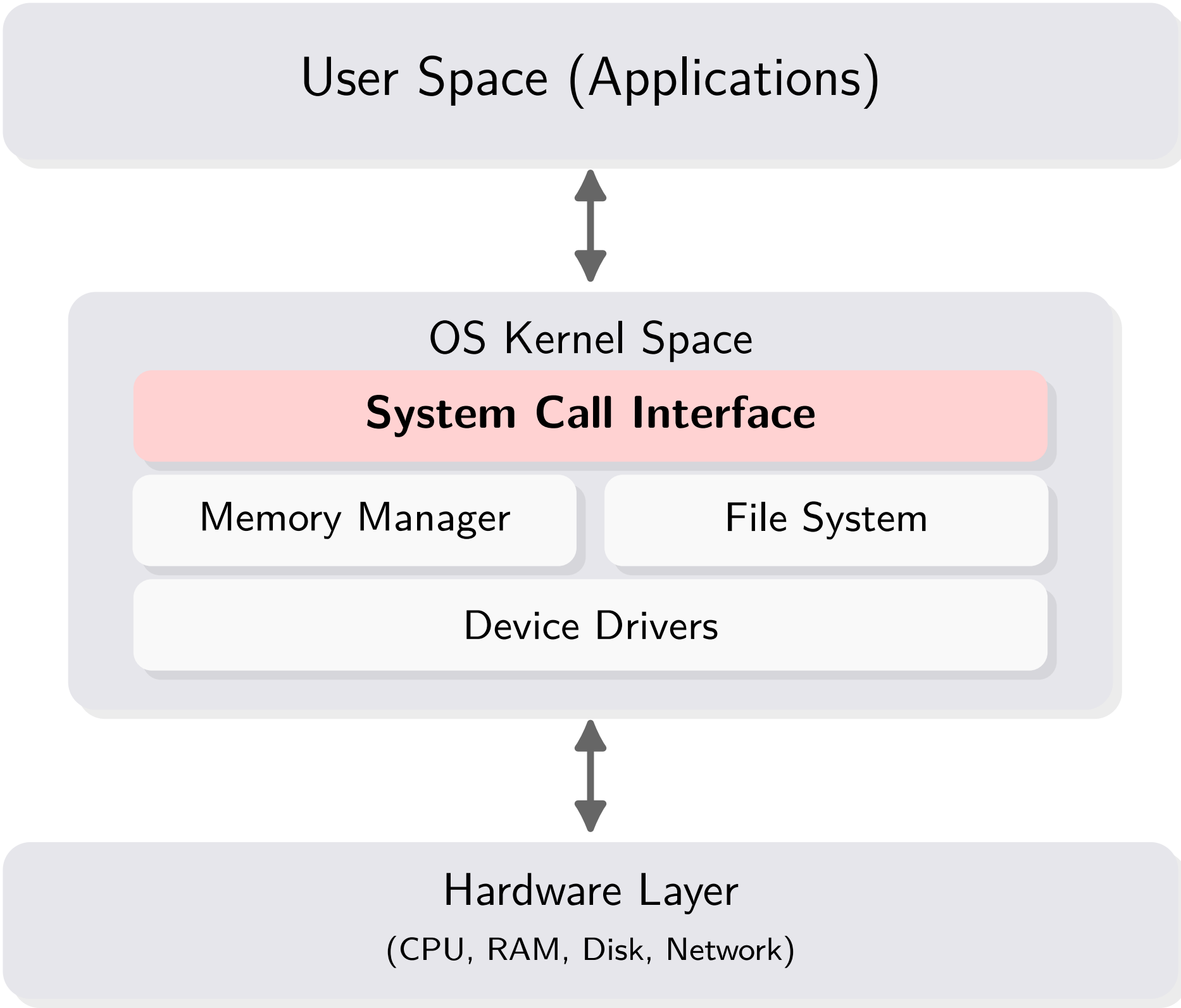
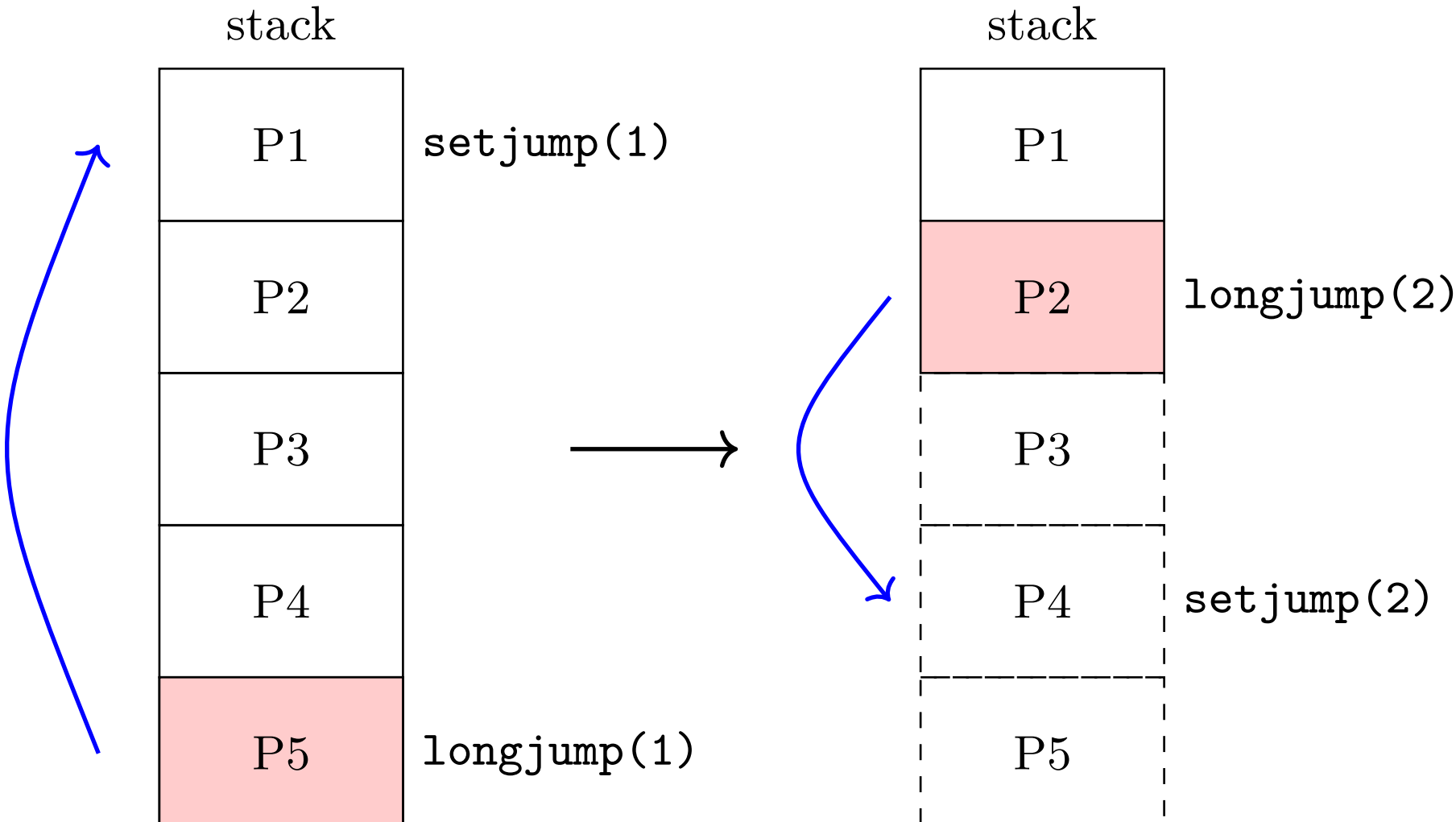
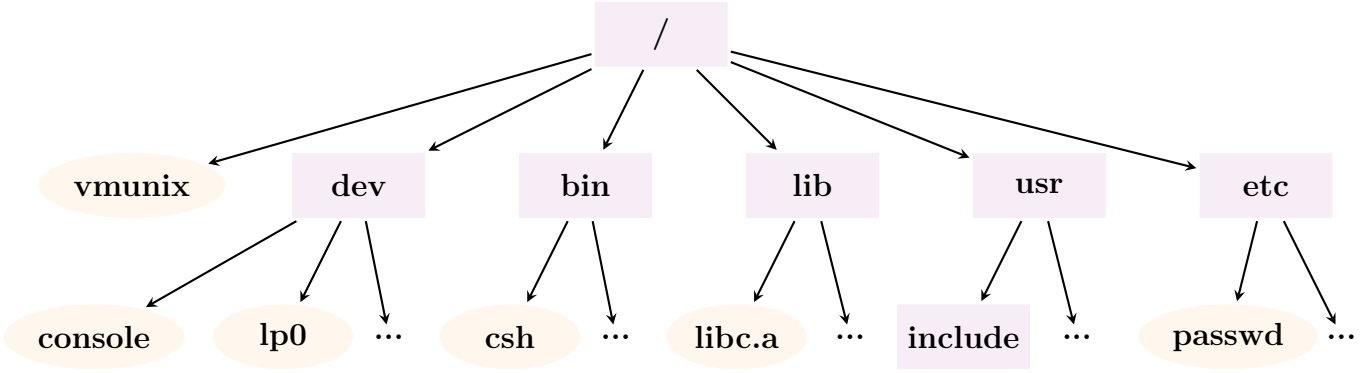
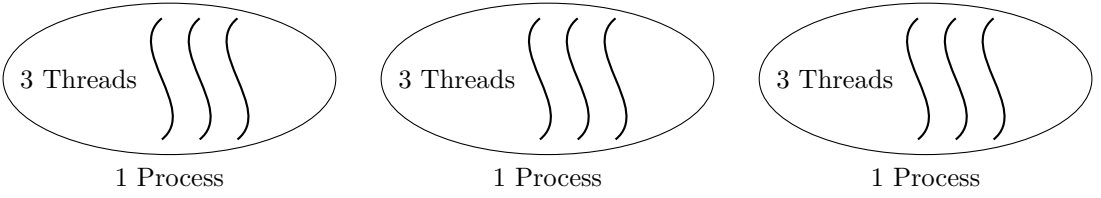

User Space (Applications)
OS Kernel Space
System Call Interface
Memory Manager
File System
Device Drivers
Hardware Layer
(CPU, RAM, Disk, Network)

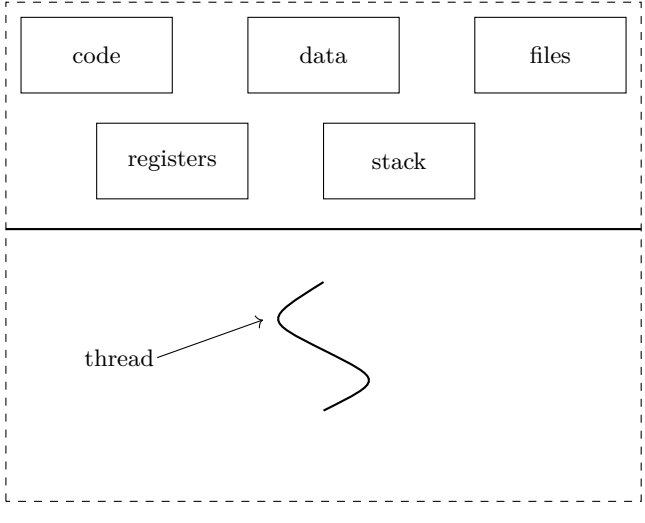

stack
P1
P2
P3
P4
P5
setjump(1)
longjump(1)
stack
P1
P2
P3
P4
P5
longjump(2)
setjump(2)


/
vmunix
dev
bin
lib
usr
etc
console
lp0
...
csh
...
libc.a
...
include
...
passwd
...

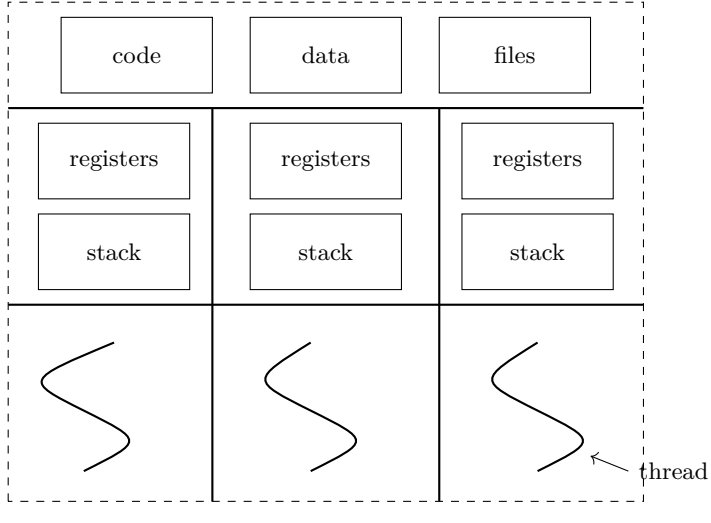
Many Process

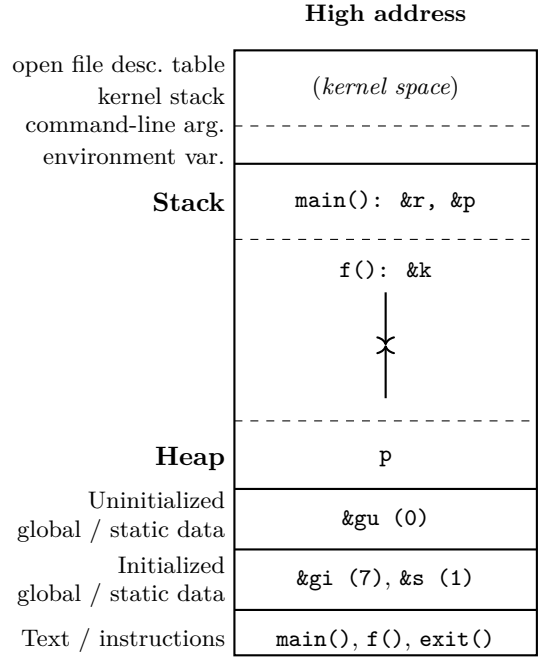


single-threaded process

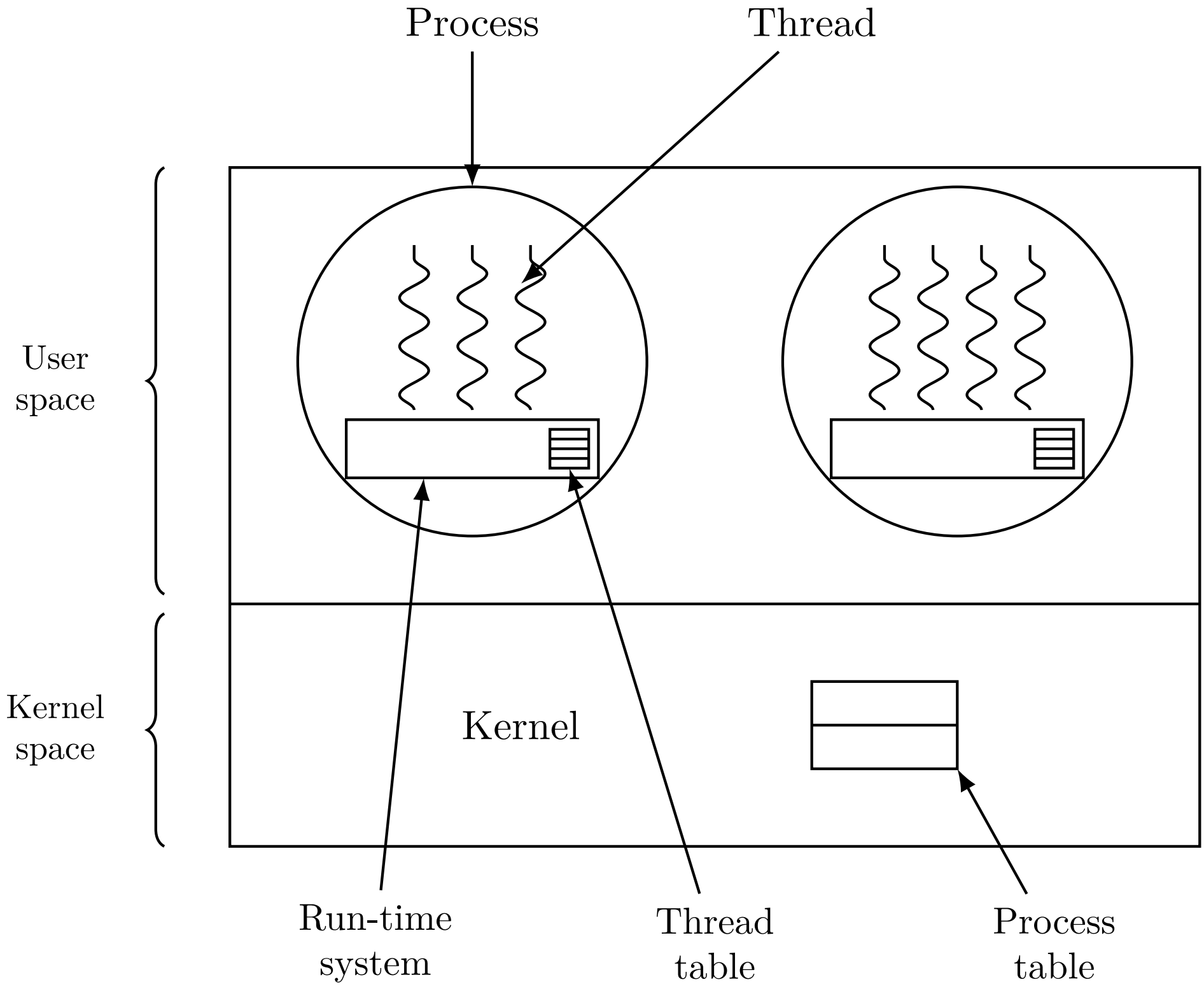


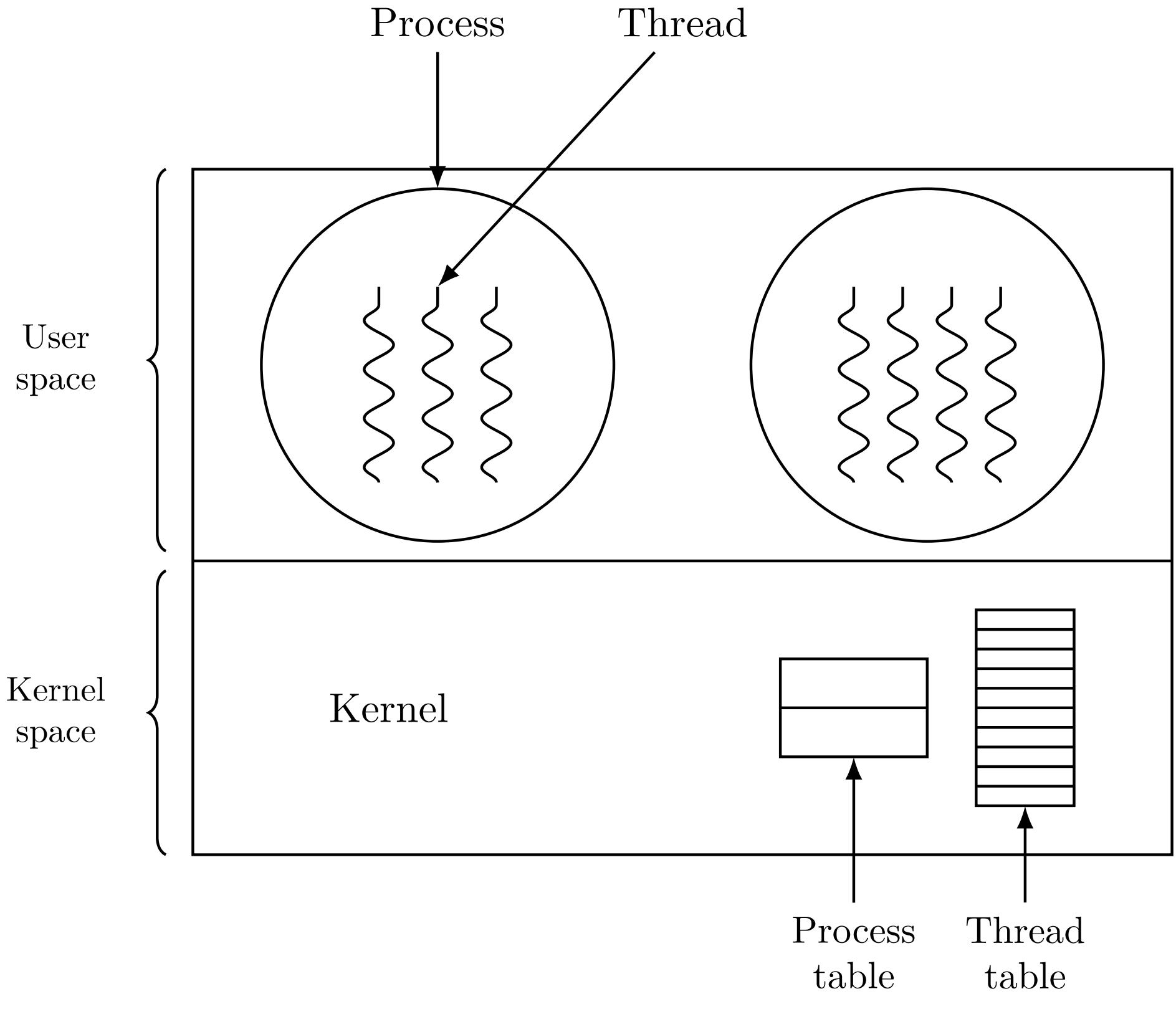
multithreaded process

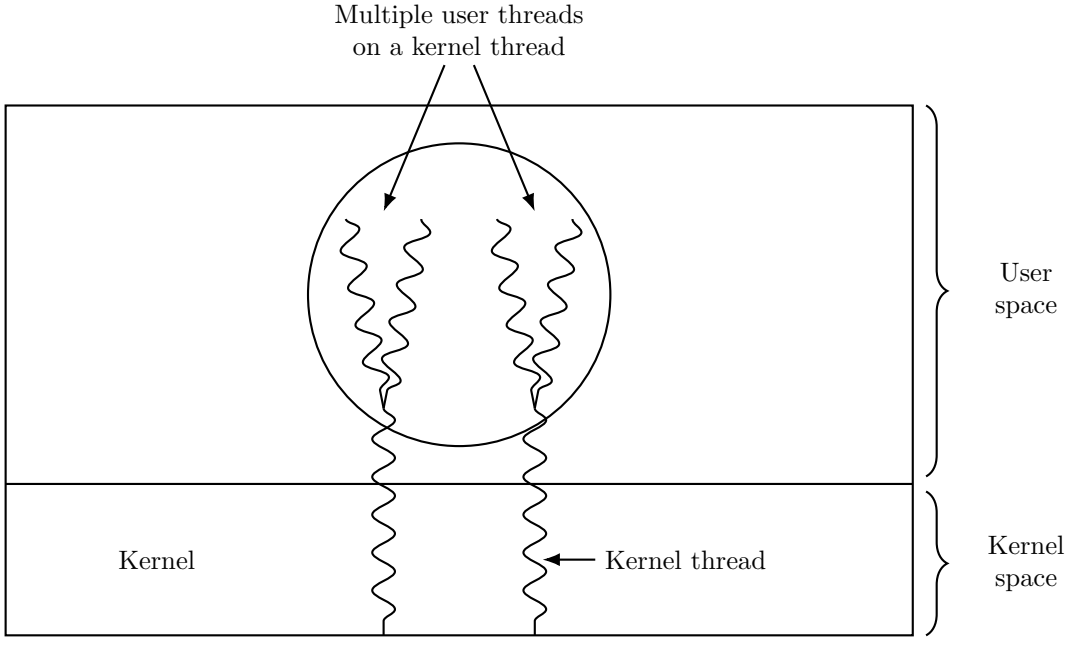


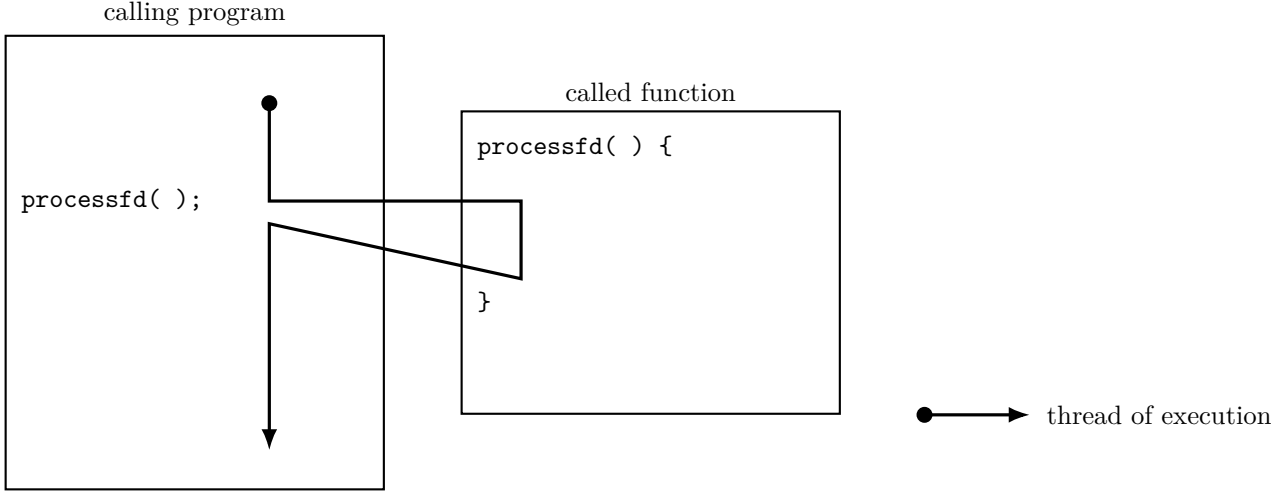


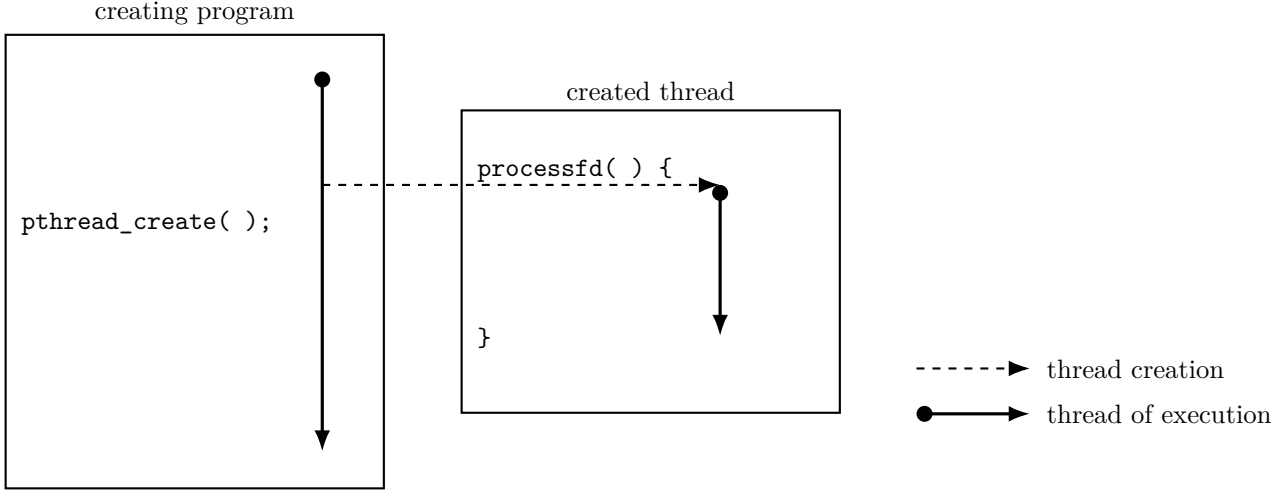

```
int gi = 7;
int gu;

int f(int k){
  static int s = 1;
}

int main(){
  int r = f();
  char *p = malloc();
  exit(0);
}
```

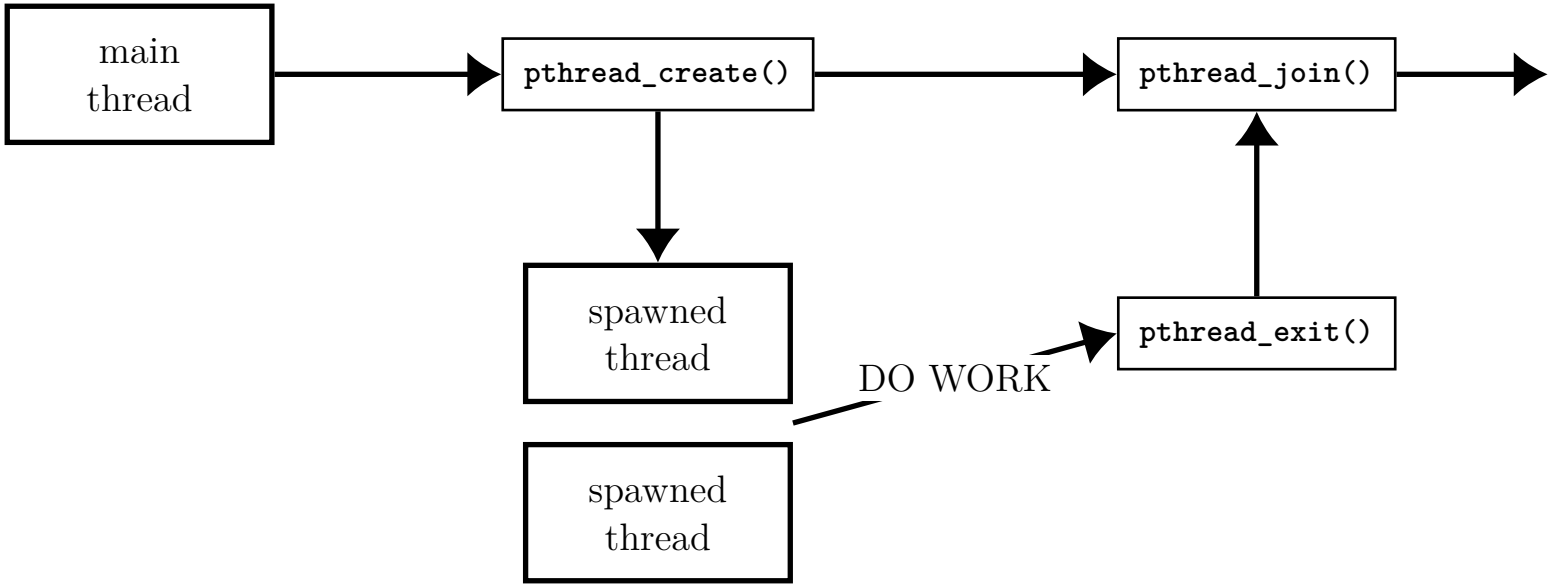

Process
Thread
User space
Kernel space
Kernel
Run-time system
Thread table
Process table


Process
Thread
User space
Kernel space
Kernel
Process table
Thread table


Multiple user threads
on a kernel thread
User
space
Kernel
Kernel thread
Kernel
space

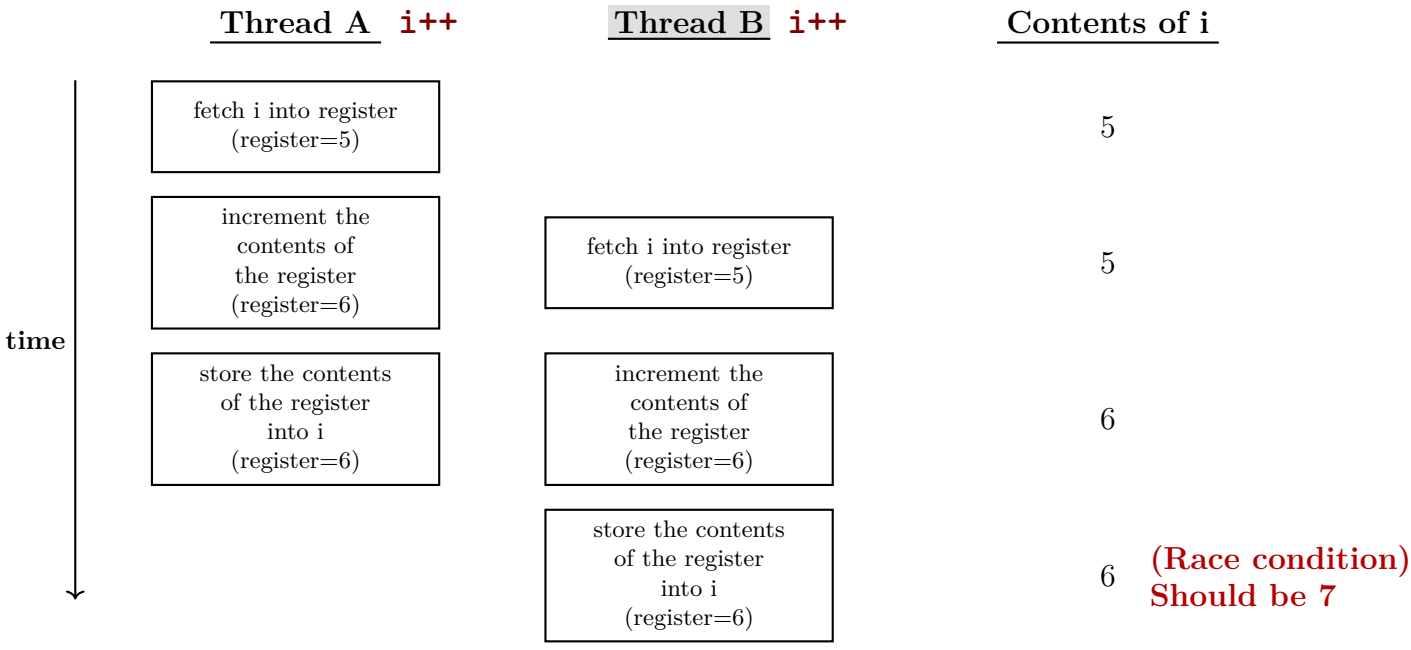

calling program
processfd( );
called function
processfd( ) {
}
thread of execution

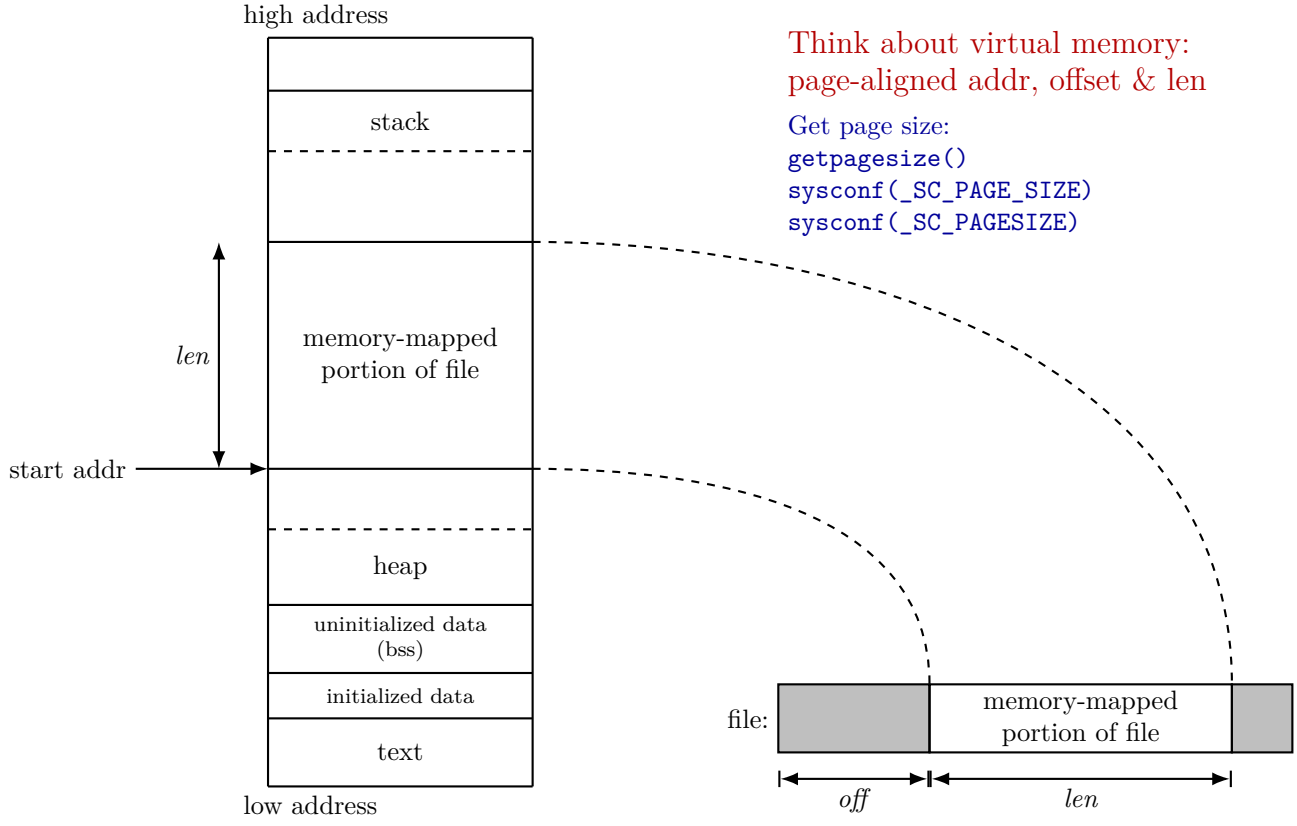

creating program
pthread_create( );
created thread
processfd( ) {
}
thread creation
thread of execution


main
thread
pthread_create()
pthread_join()
spawned
thread
pthread_exit()
DO WORK
spawned
thread

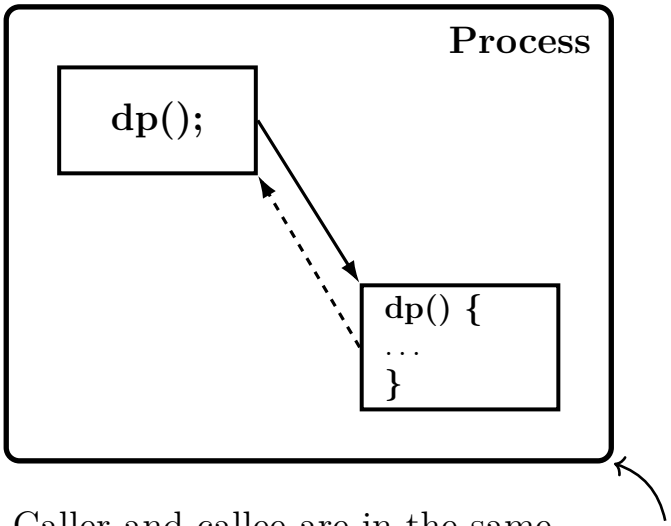
Two unsynchronized threads incrementing the same variable

1. Read the memory location into a register.
2. Increment the value in the register.
3. Write the new value back to the memory location.




high address
stack
memory-mapped
portion of file
len
start addr
heap
uninitialized data
(bss)
initialized data
text
low address
Think about virtual memory:
page-aligned addr, offset & len
Get page size:
getpagesize()
sysconf(_SC_PAGE_SIZE)
sysconf(_SC_PAGESIZE)
file:
memory-mapped
portion of file
off
len

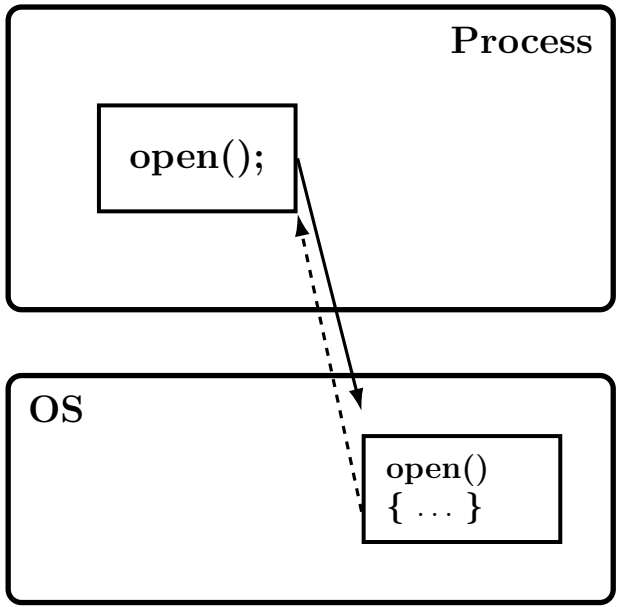
Function Call



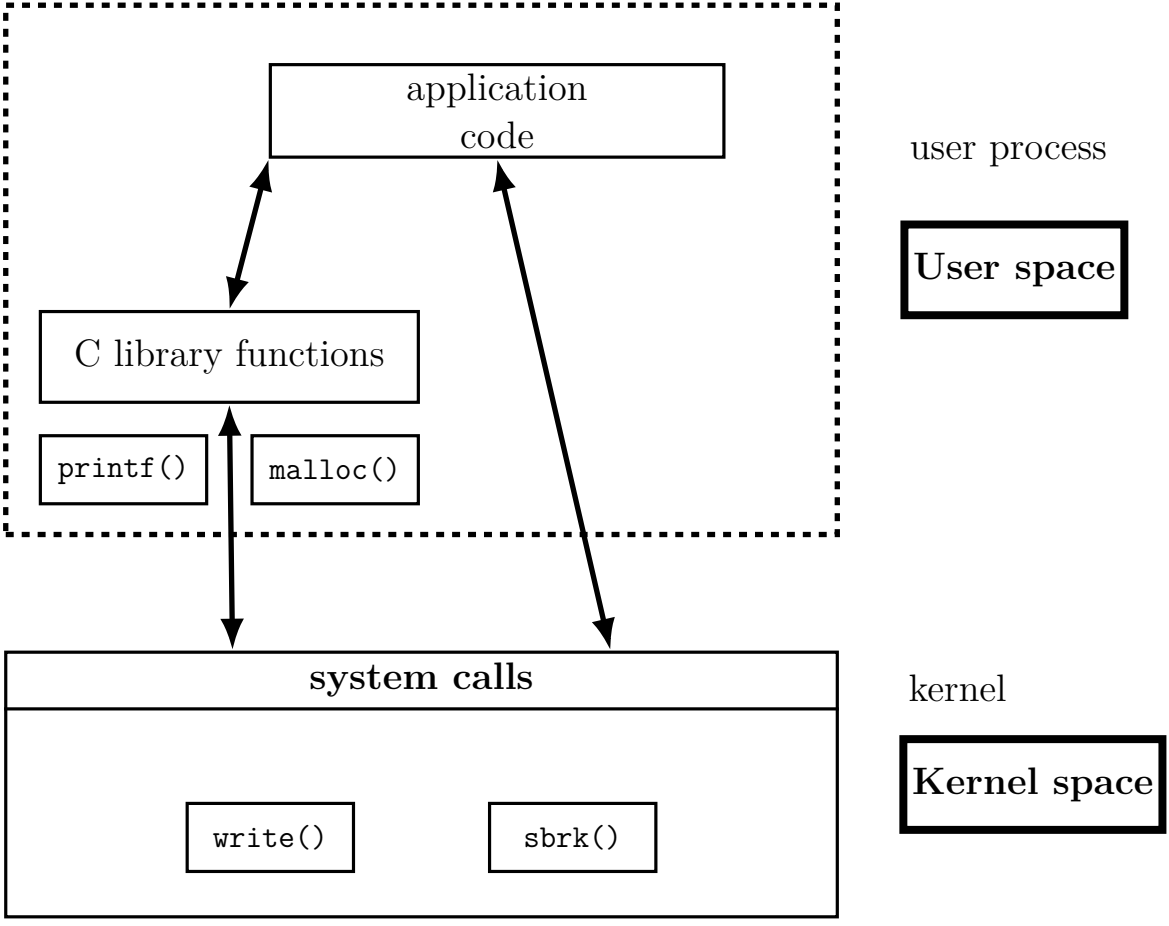
Caller and callee are in the same process

- Same user
- Same "domain of trust"

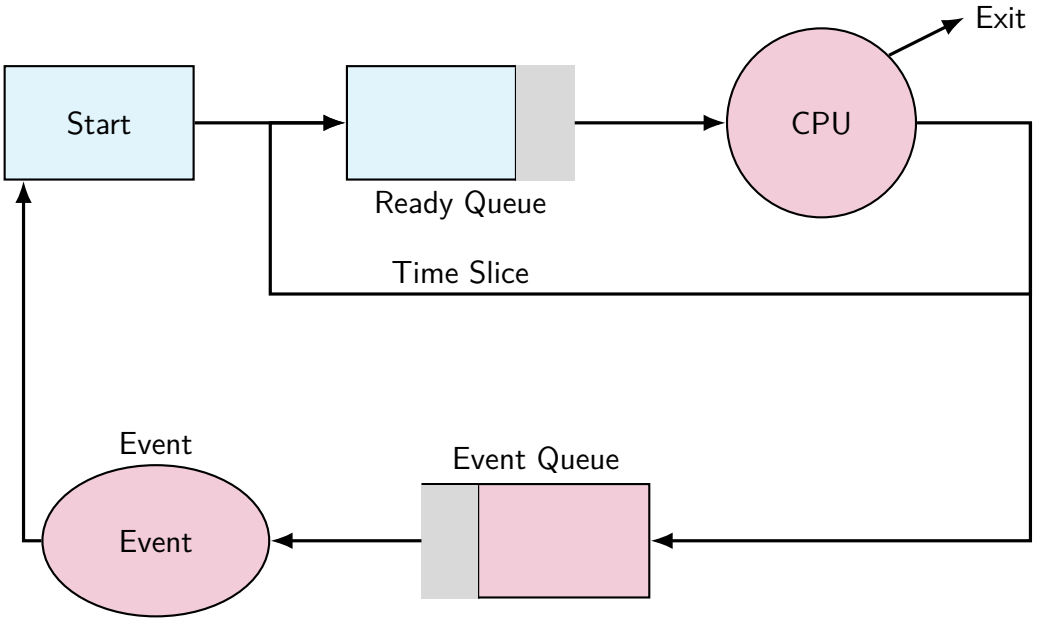
System Call

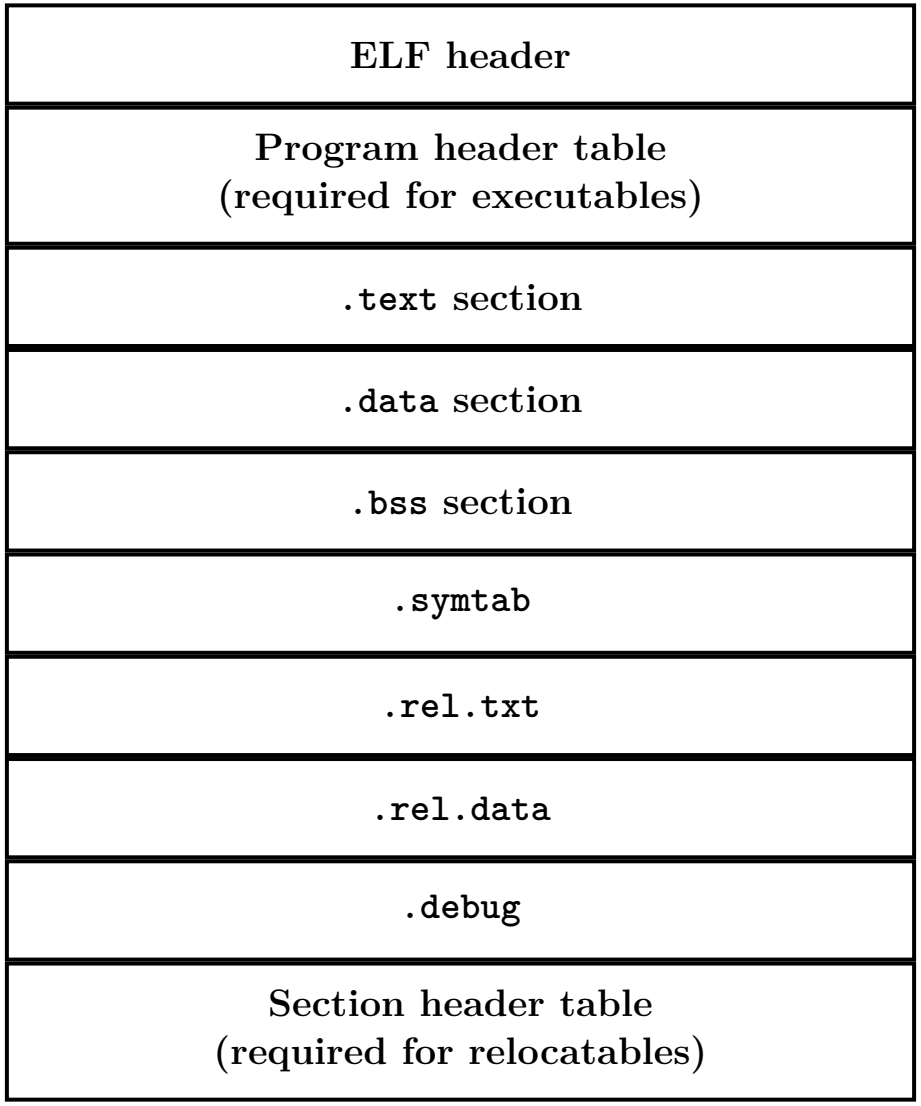


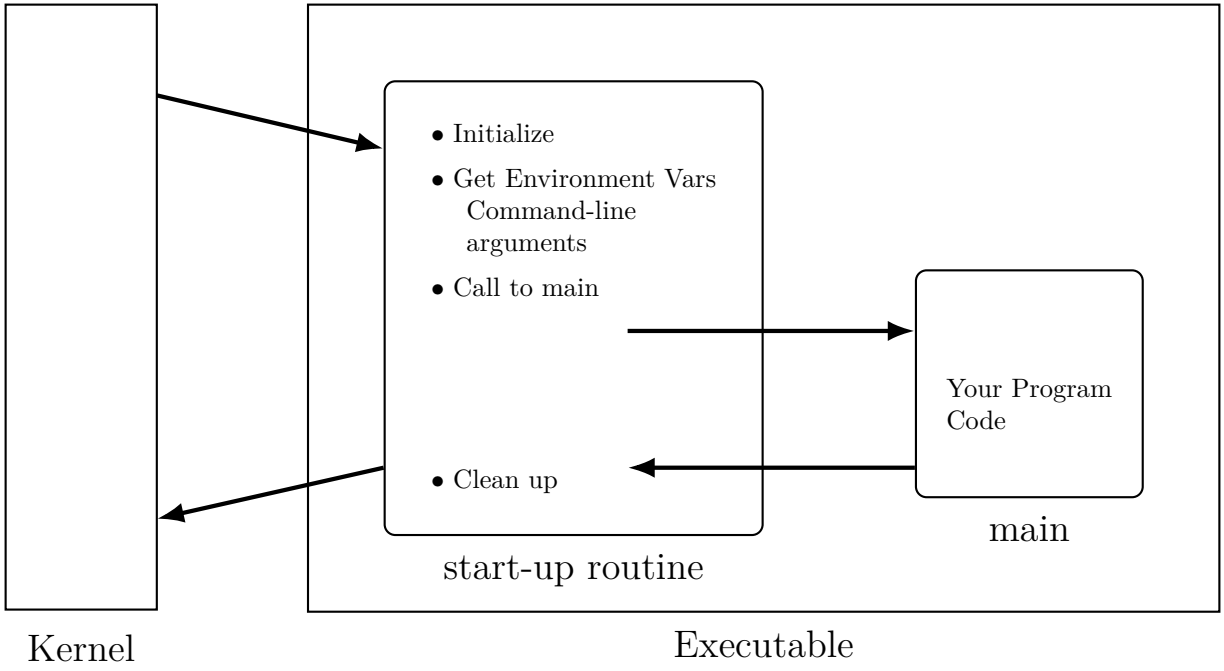
- OS is trusted; user is not.
- OS has super-privileges; user does not
- Must take measures to prevent abuse

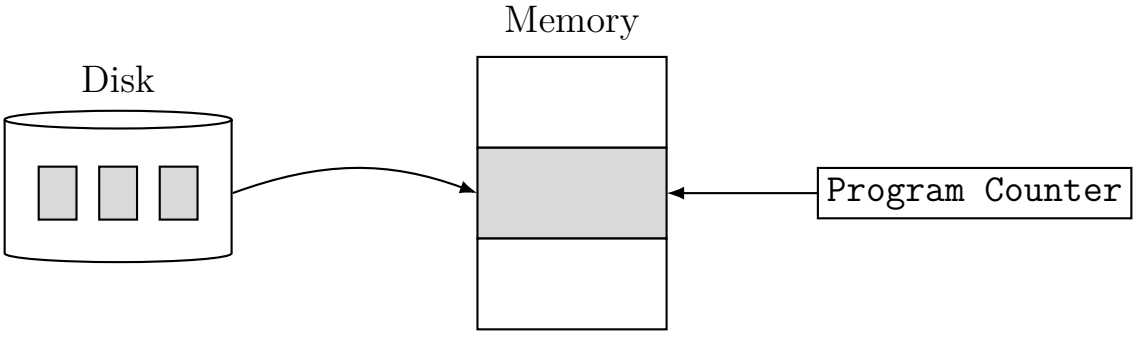

application
code
C library functions
printf()
malloc()
system calls
write()
sbrk()
user process
User space
kernel
Kernel space

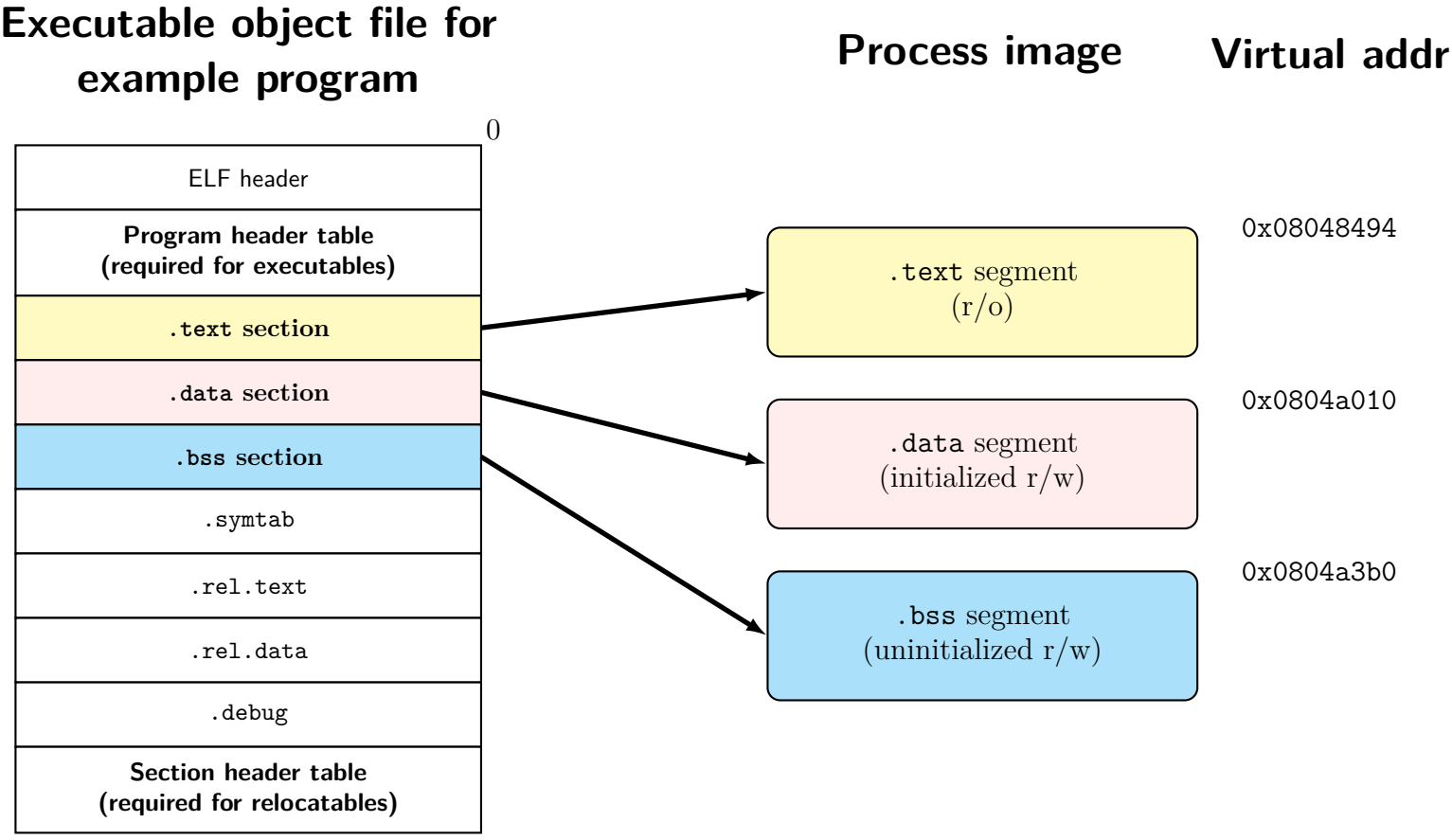
Time sharing: CPU's time is shared among multiple tasks simultaneously




ELF header
Program header table
(required for executables)
.text section
.data section
.bss section
.symtab
.rel.txt
.rel.data
.debug
Section header table
(required for relocatables)


• Initialize
• Get Environment Vars
Command-line
arguments
• Call to main
• Clean up
start-up routine
Your Program
Code
main
Kernel
Executable


Memory
Disk
Program Counter


Executable object file for example program
Process image
Virtual addr
0
ELF header
Program header table (required for executables)
.text section
.data section
.bss section
.symtab
.rel.text
.rel.data
.debug
Section header table (required for relocatables)
.text segment (r/o)
.data segment (initialized r/w)
.bss segment (uninitialized r/w)
0x08048494
0x0804a010
0x0804a3b0