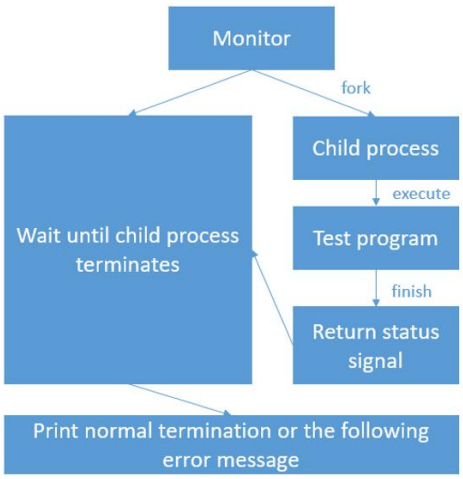
CSC3150 Project1 Report

Name: Jiaqi Li
Student ID: 120090545

Design

Design of program 1:



In Program1, we need to implement a program to do the following step:

(1) fork a child process:

Call function fork(), which will create a child process and return a pid. If the pid is 0, then it is now parent process, otherwise it is child process.

(2) execute child process and send parent a signal:

First get the filename from para argv, then use the function

int **execve**(const char ***filename**, char *const **argv**[], char *const **envp**[])

to execute the file in child process.

(3) parent process wait for the child process to finish

Use the function **pid_t waitpid**(**pid_t pid**, int *status, int **options**) in parent process to wait for the child process to end. Moreover, the options should include WUNTRACED to report on stopped child process.

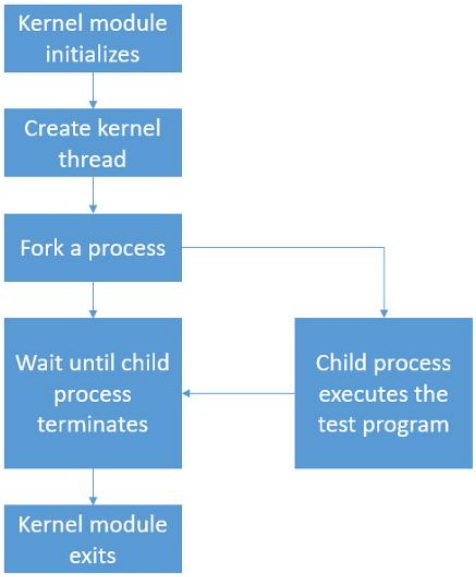
(4) parent process prints the signal from child process

By the function

```
waitpid(pid, &state, WUNTRACED|WCONTINUED);
```

We can get the signal from child process and we use printf() to output the signal in different cases.

Design of program 2:



In the program 2, we should create a kernel thread and fork a child process, and catch the signals in parent process and report them. Steps as follows:

(1) create a kernel thread and run my_fork function.

In program 2, we need to initialize and exit a kernel thread. And in initialization, we create the kernel thread using function kthread_create()

```
task = kthread_create(&my_fork, NULL, "Mythread");
```

And then wake up the process

```
        printk("[program2] : Module_init kthread starts\n");
        wake_up_process(task);
```

(2)Within my_fork, fork a process to execute the test program.

To fork a process, I use kernel_clone which will return the pid of the process.

```
    pid = kernel_clone(&args);
```

And in kernel_clone, function my_exec() is executed to run the test program.

In my_exec(), function

```
struct filename *getname_kernel(const char * filename)
```

is used to get the filename from absolute path, and

```
static int do_execve(struct filename *filename,
            const char __user *const __user *__argv,
            const char __user *const __user *__envp)
```

Is used to execute the test program:

```
result =
do_execve(getname_kernel("/home/vagrant/csc3150/template_source/source/program2/test"), NULL, NULL);
```

(3)The parent process will wait until child process terminates.
To wait until the child process terminates, function
static long **do_wait**(struct **wait_opts** *wo)
is used ” to wait for child process’ termination status. By adding WSTOPPED to wo.wo_flags, SIGSTOP will also be report.

```
    a = do_wait(&wo);
```

(4)Catch and print out the signal in child process in kernel log
The do_wait() will change the information in wo and we can get the wait option status in wo.wo_stat to retrieve the signal from child process.
For different status, we will print different information to kernel log. Besides,

```
    int my_WIFSTOPPED(int status)
```

is used to check if the child process get SIGSTOP

Setting up development environment

(1) Set up VM

Install vitrualbox and vagrant.After installation, **reboot computer.**
Set up a directory for csc3150 (D:\CSC3150).
Go to Powershell and change to D:\CSC3150 and execute:
vagrant init cyzhu/csc3150
vagrant up

(2) Set up Vscode

Set up Remote SSH plugin in VS Code by installing Remote-SSH
Go back to powershell Copy everything but the last line (Loglevel) to the ssh config and save the file
find SSH Target to connect to the VM and launch a new window.
install essential dependencies and libraries:
sudo apt update && sudo apt install -y build-essential

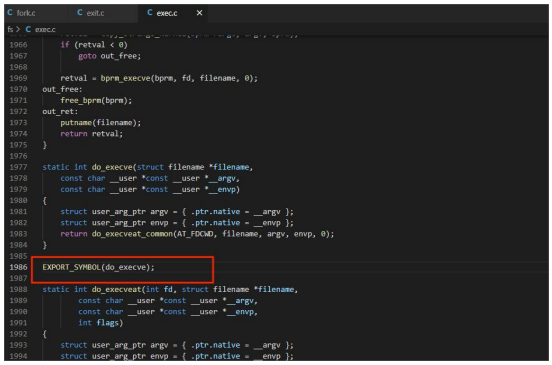
(3) Compile the kernel

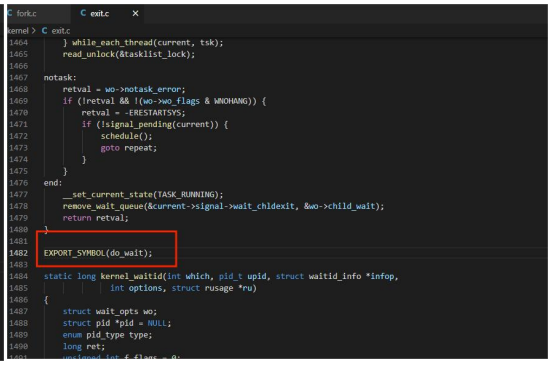
Download source code from https://mirror.tuna.tsinghua.edu.cn/kernel/v5.x/
Install Dependency and development tools:
Extract the source file:
$sudo tar xvf Linux-5-10-27.tar.xz
Copy .config from /boot to ~/Linux-5-10-27
Login root account: $sudo su
Clean previous setting and start configuration:
$make mrproper

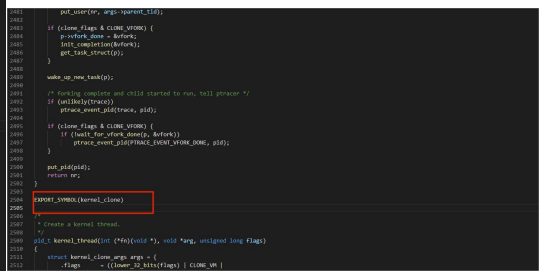
$make clean
$make menuconfig
Then save the config and exit.
Build kernel Image and modules: $make –j$(nproc)
Install kernel modules : $make modules_install
Install kernel : $make install
Reboot to load new kernel : $reboot

(4) Modify the kernel:

use EXPORT_SYMBOL() to provides API for function that are used in program2.c







Then rebuilt the kernel module and install the updated kernel:
$make bzImage -j$(nproc)
$make modules -j$(nproc)
$make modules_install
$make install
$reboot

Program execution

Program 1:

Change the terminal to folder of program1: cd program1
Makefile: make

Program 2:

Change the terminal to folder of program1: cd program2
Makefile: make
Login into root: sudo su
Compile the test file: gcc test.c -o test
Insert the module: insmod program2.ko
Remove the module: rmmod program2.ko
Show the kernel log: dmesg

Note:
For testing several test file, you can simply copy the content of file to test.c,
Then execute the commands without makefile again:
gcc test.c -o test
insmod program2.ko
rmmod program2.ko
dmesg

Screenshot of the Result

Program 1:

```
vagrant@csc3150:~/csc3150/template_source/source$ cd program1
vagrant@csc3150:~/csc3150/template_source/source/program1$ make
cc -o program1 program1.c
./program1 ./normal
I'm the Parent Process, my pid = 24562
I'm the Child Process, my pid = 24563
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the normal program

------------CHILD PROCESS END------------
Parent process receives SIGCHLD signal
Normal termination with EXIT STATUS = 0

./program1 ./abort
I'm the Parent Process, my pid = 24564
I'm the Child Process, my pid = 24565
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGABRT program

Parent process receives SIGCHLD signal
child process get SIGABRT signal

./program1 ./alarm
I'm the Parent Process, my pid = 24567
I'm the Child Process, my pid = 24568
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGALRM program

Parent process receives SIGCHLD signal
child process get SIGALRM signal

./program1 ./bus
I'm the Parent Process, my pid = 24596
I'm the Child Process, my pid = 24597
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGBUS program

Parent process receives SIGCHLD signal
child process get SIGBUS signal

./program1 ./floating
I'm the Parent Process, my pid = 24599
I'm the Child Process, my pid = 24600
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGFPE program

Parent process receives SIGCHLD signal
child process get SIGFPE signal
```

```
./program1 ./hangup
I'm the Parent Process, my pid = 24602
I'm the Child Process, my pid = 24603
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGHUP program

Parent process receives SIGCHLD signal
child process get SIGHUP signal

./program1 ./illegal_instr
I'm the Parent Process, my pid = 24604
I'm the Child Process, my pid = 24605
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGILL program

Parent process receives SIGCHLD signal
child process get SIGILL signal

./program1 ./interrupt
I'm the Parent Process, my pid = 24607
I'm the Child Process, my pid = 24608
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGINT program

Parent process receives SIGCHLD signal
child process get SIGINT signal

./program1 ./kill
I'm the Parent Process, my pid = 24609
I'm the Child Process, my pid = 24610
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGKILL program

Parent process receives SIGCHLD signal
child process get SIGKILL signal

./program1 ./pipe
I'm the Parent Process, my pid = 24611
I'm the Child Process, my pid = 24612
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGPIPE program

Parent process receives SIGCHLD signal
child process get SIGPIPE signal
```

```
./program1 ./quit
I'm the Parent Process, my pid = 24613
I'm the Child Process, my pid = 24614
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGQUIT program

Parent process receives SIGCHLD signal
child process get SIGQUIT signal

./program1 ./segment_fault
I'm the Parent Process, my pid = 24616
I'm the Child Process, my pid = 24617
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSEGV program

Parent process receives SIGCHLD signal
child process get SIGSEGV signal

./program1 ./stop
I'm the Parent Process, my pid = 24619
I'm the Child Process, my pid = 24620
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSTOP program

Parent process receives SIGCHLD signal
child process get SIGSTOP signal

./program1 ./terminate
I'm the Parent Process, my pid = 24621
I'm the Child Process, my pid = 24622
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTERM program

Parent process receives SIGCHLD signal
child process get SIGTERM signal

./program1 ./trap
I'm the Child Process, my pid = 24624
Child process start to execute test program:
I'm the Parent Process, my pid = 24623
------------CHILD PROCESS START------------
This is the SIGTRAP program

Parent process receives SIGCHLD signal
child process get SIGTRAP signal
```

Program 2:

Test.c:

```
[ 1847.614159] [program2] : Module_init 李佳齐 120090545
[ 1847.614160] [program2] : module_init create kthread start
[ 1847.614373] [program2] : Module_init kthread starts
[ 1847.614416] [program2] : The child process has pid = 7874
[ 1847.614417] [program2] : This is the parent process, pid = 7873
[ 1847.614417] [program2] : child process
[ 1847.696192] [program2] : get SIGBUS signal
[ 1847.696194] [program2] : child process terminated
[ 1847.696194] [program2] : The return signal is 7
[ 1854.171449] [program2] : Module_exit
```

abort.c

```
[ 1946.559493] [program2] : Module_init 李佳齐 120090545
[ 1946.559495] [program2] : module_init create kthread start
[ 1946.559557] [program2] : Module_init kthread starts
[ 1946.559597] [program2] : The child process has pid = 8348
[ 1946.559598] [program2] : This is the parent process, pid = 8347
[ 1946.559599] [program2] : child process
[ 1946.632177] [program2] : get SIGABRT signal
[ 1946.632178] [program2] : child process terminated
[ 1946.632179] [program2] : The return signal is 6
[ 1951.355998] [program2] : Module_exit
```

alarm.c

```
[ 2054.127392] [program2] : Module_init 李佳齐 120090545
[ 2054.127394] [program2] : module_init create kthread start
[ 2054.127445] [program2] : Module_init kthread starts
[ 2054.127484] [program2] : The child process has pid = 8794
[ 2054.127485] [program2] : This is the parent process, pid = 8793
[ 2054.127486] [program2] : child process
[ 2056.127835] [program2] : get SIGALRM signal
[ 2056.127837] [program2] : child process terminated
[ 2056.127838] [program2] : The return signal is 14
[ 2066.548593] [program2] : Module_exit
```

bus.c

```
[ 2240.704900] [program2] : Module_init 李佳齐 120090545
[ 2240.704901] [program2] : module_init create kthread start
[ 2240.704959] [program2] : Module_init kthread starts
[ 2240.704987] [program2] : The child process has pid = 9561
[ 2240.704988] [program2] : This is the parent process, pid = 9560
[ 2240.704988] [program2] : child process
[ 2240.781170] [program2] : get SIGBUS signal
[ 2240.781171] [program2] : child process terminated
[ 2240.781172] [program2] : The return signal is 7
[ 2258.969375] [program2] : Module_exit
```

floating.c

```
[ 2314.747898] [program2] : Module_init 李佳齐 120090545
[ 2314.747899] [program2] : module_init create kthread start
[ 2314.747952] [program2] : Module_init kthread starts
[ 2314.748017] [program2] : The child process has pid = 9961
[ 2314.748018] [program2] : This is the parent process, pid = 9960
[ 2314.748019] [program2] : child process
[ 2314.844287] [program2] : get SIGFPE signal
[ 2314.844289] [program2] : child process terminated
[ 2314.844290] [program2] : The return signal is 8
[ 2320.208063] [program2] : Module_exit
```

hangup.c

```
[ 2401.748958] [program2] : Module_init 李佳齐 120090545
[ 2401.748960] [program2] : module_init create kthread start
[ 2401.749007] [program2] : Module_init kthread starts
[ 2401.749045] [program2] : The child process has pid = 10343
[ 2401.749046] [program2] : This is the parent process, pid = 10342
[ 2401.749046] [program2] : child process
[ 2401.749295] [program2] : get SIGHUP signal
[ 2401.749296] [program2] : child process terminated
[ 2401.749297] [program2] : The return signal is 1
[ 2409.104542] [program2] : Module_exit
```

illegal_instr.c

```
[ 2474.466372] [program2] : Module_init 李佳齐 120090545
[ 2474.466373] [program2] : module_init create kthread start
[ 2474.466428] [program2] : Module_init kthread starts
[ 2474.466462] [program2] : The child process has pid = 10713
[ 2474.466463] [program2] : This is the parent process, pid = 10712
[ 2474.466464] [program2] : child process
[ 2474.538595] [program2] : get SIGILL signal
[ 2474.538597] [program2] : child process terminated
[ 2474.538598] [program2] : The return signal is 4
[ 2481.416109] [program2] : Module_exit
```

interrupt.c

```
[ 2543.312842] [program2] : Module_init 李佳齐 120090545
[ 2543.312843] [program2] : module_init create kthread start
[ 2543.312892] [program2] : Module_init kthread starts
[ 2543.312927] [program2] : The child process has pid = 11078
[ 2543.312927] [program2] : This is the parent process, pid = 11077
[ 2543.312928] [program2] : child process
[ 2543.313179] [program2] : get SIGINT signal
[ 2543.313179] [program2] : child process terminated
[ 2543.313180] [program2] : The return signal is 2
[ 2551.677055] [program2] : Module_exit
```

kill.c

```
[ 2614.289165] [program2] : Module_init 李佳齐 120090545
[ 2614.289166] [program2] : module_init create kthread start
[ 2614.289219] [program2] : Module_init kthread starts
[ 2614.289254] [program2] : The child process has pid = 11463
[ 2614.289255] [program2] : This is the parent process, pid = 11462
[ 2614.289256] [program2] : child process
[ 2614.289744] [program2] : get SIGKILL signal
[ 2614.289745] [program2] : child process terminated
[ 2614.289746] [program2] : The return signal is 9
[ 2619.748890] [program2] : Module_exit
```

normal.c

```
[ 2669.113107] [program2] : Module_init 李佳齐 120090545
[ 2669.113109] [program2] : module_init create kthread start
[ 2669.113169] [program2] : Module_init kthread starts
[ 2669.113201] [program2] : The child process has pid = 11748
[ 2669.113202] [program2] : This is the parent process, pid = 11747
[ 2669.113203] [program2] : child process
[ 2669.113543] [program2] : child process exit normally
[ 2669.113545] [program2] : child process terminated
[ 2669.113545] [program2] : The return signal is 0
[ 2683.022367] [program2] : Module_exit
```

pipe.c

```
[ 2736.007282] [program2] : Module_init 李佳齐 120090545
[ 2736.007284] [program2] : module_init create kthread start
[ 2736.007338] [program2] : Module_init kthread starts
[ 2736.007360] [program2] : The child process has pid = 12031
[ 2736.007369] [program2] : This is the parent process, pid = 12030
[ 2736.007373] [program2] : child process
[ 2736.007588] [program2] : get SIGPIPE signal
[ 2736.007589] [program2] : child process terminated
[ 2736.007590] [program2] : The return signal is 13
[ 2742.894545] [program2] : Module_exit
```

quit.c

```
[ 2798.787610] [program2] : Module_init 李佳齐 120090545
[ 2798.787612] [program2] : module_init create kthread start
[ 2798.787666] [program2] : Module_init kthread starts
[ 2798.787697] [program2] : The child process has pid = 12361
[ 2798.787702] [program2] : This is the parent process, pid = 12360
[ 2798.787702] [program2] : child process
[ 2798.863561] [program2] : get SIGQUIT signal
[ 2798.863562] [program2] : child process terminated
[ 2798.863563] [program2] : The return signal is 3
[ 2805.008854] [program2] : Module_exit
```

segment_fault.c

```
[ 2881.188986] [program2] : Module_init 李佳齐 120090545
[ 2881.188987] [program2] : module_init create kthread start
[ 2881.189035] [program2] : Module_init kthread starts
[ 2881.189073] [program2] : The child process has pid = 12783
[ 2881.189074] [program2] : This is the parent process, pid = 12782
[ 2881.189075] [program2] : child process
[ 2881.266248] [program2] : get SIGSEGV signal
[ 2881.266250] [program2] : child process terminated
[ 2881.266251] [program2] : The return signal is 11
[ 2884.619817] [program2] : Module_exit
```

stop.c

```
[  421.471589] [program2] : Module_init 李佳齐 120090545
[  421.471590] [program2] : module_init create kthread start
[  421.471644] [program2] : Module_init kthread starts
[  421.471679] [program2] : The child process has pid = 3594
[  421.471680] [program2] : This is the parent process, pid = 3593
[  421.471681] [program2] : child process
[  421.471850] [program2] : child process get SIGSTOP signal
[  421.471851] [program2] : child process stopped
[  421.471851] [program2] : The return signal is 19
[  425.053545] [program2] : Module_exit
```

terminate.c

```
[ 2930.293778] [program2] : Module_init 李佳齐 120090545
[ 2930.293780] [program2] : module_init create kthread start
[ 2930.293837] [program2] : Module_init kthread starts
[ 2930.293874] [program2] : The child process has pid = 13051
[ 2930.293875] [program2] : This is the parent process, pid = 13050
[ 2930.293875] [program2] : child process
[ 2930.294164] [program2] : get SIGTERM signal
[ 2930.294165] [program2] : child process terminated
[ 2930.294165] [program2] : The return signal is 15
[ 2935.527920] [program2] : Module_exit
```

trap.c

```
[ 2987.447369] [program2] : Module_init 李佳齐 120090545
[ 2987.447371] [program2] : module_init create kthread start
[ 2987.447427] [program2] : Module_init kthread starts
[ 2987.447457] [program2] : The child process has pid = 13346
[ 2987.447458] [program2] : This is the parent process, pid = 13345
[ 2987.447458] [program2] : child process
[ 2987.517658] [program2] : get SIGTRAP signal
[ 2987.517660] [program2] : child process terminated
[ 2987.517661] [program2] : The return signal is 5
[ 2994.452925] [program2] : Module_exit
```

What I learned in this task

In project 1, I get a better understanding of fork(), child and parent process.
In project 2, I learn about syscall from executing child process and waiting for child process, and I learn more about kernel and thread while creating kthread and building kernel module.
Finally, the ability to learn from google and ability to work on a big project is greatly improved!