CSC3150 A1 Report

P.S.

- There is also a markdown file here, which is more beautiful, but the output screenshots are only available in this PDF file.
- To run this program in your computer, you need to first change the absolute path in program2.c to the path of your file. It is very strange that I cannot use relative path in my program.

Part1: Problem Brief

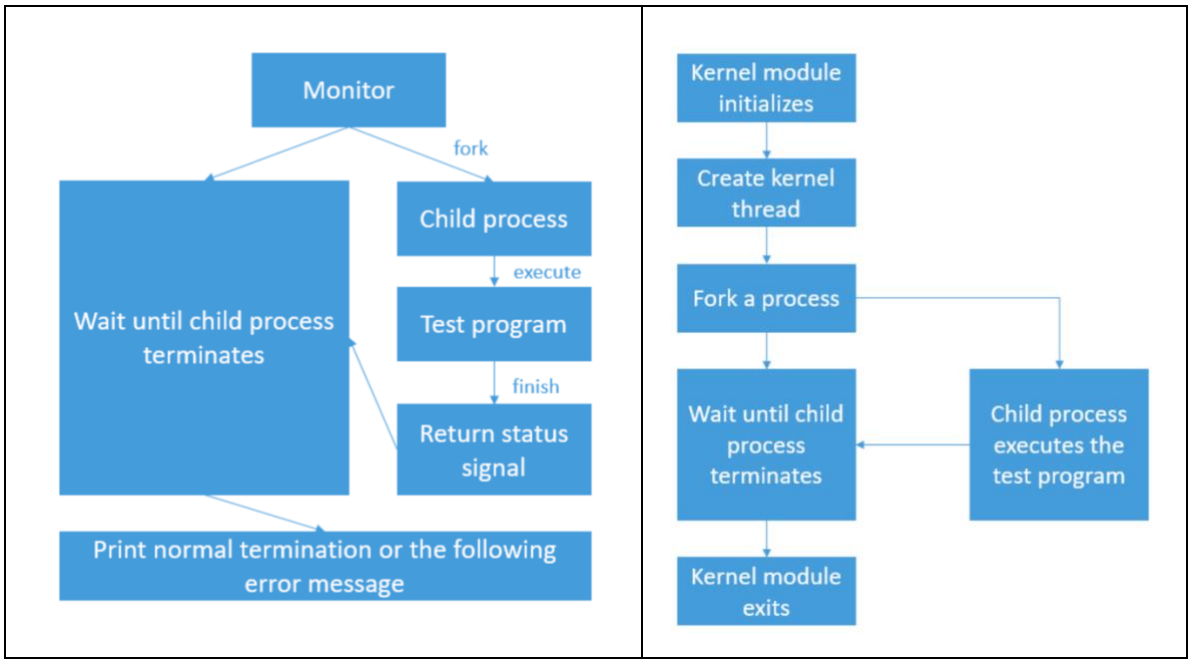
The project is divided into three parts. In part1, we are required to write a program(program1.c) to complete the tasks in part1.

The tasks in part1 includes: 1. Fork a child process to execute test programs (15 of them) 2. Use wait() to let the parent process receives the SIGCHLD signal 3. Print out the termination information of child process (normal or abnormal)

The tasks un part2 includes: 1. Create a kernel thread and run my_fork function 2. Fork a process to execute test.o 3. Use do_wait() to let the parent process wait for the child process 4. Print out pid of both parent and child processes 5. Catch the signal raised by the child process and print out related log 6. Recompile the Linux kernel source code to use its functions

Part2: Overall Project Structure

Program1	Program2



Part3: Function Explanation

Program1:

Fork the child process

```
pid_t pid;
printf("Process start to fork\n");
pid = fork();
```

wait for SIGCHLD signal

```
/* wait for child process terminates */
waitpid(-1, &status, WUNTRACED);
printf("Parent process receives the SIGCHLD signal\n");
```

Child process execute test programs

```
if (pid == 0) { // child process
    int i;
    char *arg[argc];

    for (i = 0; i < argc - 1; i++) {
        arg[i] = argv[i + 1];
    }
    arg[argc - 1] = NULL;

    /* execute test program */
    printf("I'm the child process, my pid = %d\n", getpid());
    printf("Child process start to execute test program:\n");
    // start execute the program
    execve(arg[0], arg, NULL);
```

```
   printf("Continue to run original child process!\n");

   perror("execve");
   exit(SIGCHLD);
```

Analyse exit status and print out info

```
/* check child process'  termination status */
if(WIFEXITED(status)){ // normal exit
   printf("Normal termination with EXIT STATUS =
%d\n",WEXITSTATUS(status));
}

else if(WIFSIGNALED(status)){ // abnormal exit
   int num = WTERMSIG(status);
   switch (num){
       case 6: // SIGABRT
           printf("child process get SIGABRT signal\n");
           printf("child process is abort by abort signal\n");
           printf("CHILD EXECUTION FAILED!!\n");
           break;
       case 14: // SIGALRM
           printf("child process get SIGALRM signal\n");
           printf("child process is abort by alarm signal\n");
           printf("CHILD EXECUTION FAILED!!\n");
           break;
       case 7: // SIGBUS
           printf("child process get SIGBUS signal\n");
           printf("child process is abort by bus error signal\n");
           printf("CHILD EXECUTION FAILED!!\n");
           break;
       ......
   }

}
else if(WIFSTOPPED(status)){ // stop signal
   printf("child process get SIGSTOP signal\n");
   printf("child process stopped\n");
   printf("CHILD EXECUTION STOPPED\n");            }
else{
   printf("CHILD PROCESS CONTINUED\n");
}
```

Program2

Create a kernel thread and run my_fork

```
task = kthread_create(&my_fork, NULL, "MyThread");
//wake up new thread if ok
if (!IS_ERR(task)) {
   printk("[program2] : Module_init kthread starts\n");
```

```
    wake_up_process(task);
}
```

Fork a process and print out pid

```
pid_t pid;
/* fork a process using do_fork */
pid = _do_fork(SIGCHLD, (unsigned long) &my_exec, 0, NULL, NULL, 0);

printk("[program2] : The child process has pid= %d\n", pid);
printk("[program2] : The parent process has pid= %d\n", (int) current-
>pid);
```

Execute the test program

```
int my_exec(void) {
  int result;
  const char path[] = "/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/
       source/program2/test";
  const char *const argv[] = {path, NULL, NULL};
  const char *const envp[] = {"HOME=/",
"PATH=/sbin:/user/sbin:/bin:/usr/bin", NULL};

  struct filename *my_filename = getname(path);

  /* execute a test program in child process */
  printk("[program2] : child process");

  result = do_execve(my_filename, argv, envp);

  if (!result) {
      return 0;
  } else {
      do_exit(result);
  }
}
```

Wait for child process termination

```
void my_wait(pid_t pid) {
    struct wait_opts wo;
    struct pid *wo_pid = NULL;
    enum pid_type type;
    type = PIDTYPE_PID;
    wo_pid = find_get_pid(pid);

    wo.wo_type = type;
    wo.wo_pid = wo_pid;
    wo.wo_flags = WEXITED;
    wo.wo_info = NULL;
    wo.wo_stat = (int __user*)&status;
    wo.wo_rusage = NULL;

    int a;
```

```
    a = do_wait(&wo);

    output_info(status);

    put_pid(wo_pid);
   return;
}
```

Catch the signal and printed out message

```
void output_info(int exit){
    switch (exit) {
        case 1:
            printk("[program2] : get SIGHUP signal\n");
            printk("[program2] : child process is hung up\n");
            printk("[program2] : The return signal is 1\n");
            break;
        case 2:
            printk("[program2] : get SIGINT signal\n");
            printk("[program2] : terminal interrupt\n");
            printk("[program2] : The return signal is 2\n");
            break;
        case 131:
            printk("[program2] : get SIGQUIT signal\n");
            printk("[program2] : terminal quit\n");
            printk("[program2] : The return signal is 3\n");
            break;
        ......
    }
    return;
}
```

Recompile the kernel in order to use kernel function

Since I am able to declare these extern functions and use them in my program, it proves that I have recompiled the kernel and exported these symbols.

```
extern long do_wait(struct wait_opts *wo);
extern struct filename * getname(const char __user * filename);
extern long _do_fork(unsigned long clone_flags,unsigned long
stack_start,
      unsigned long stack_size,int __user *parent_tidptr, int __user
*child_tidptr,
      unsigned long tls);
extern int do_execve(struct filename *filename,const char __user *const
__user *__argv,
      const char __user *const __user *__envp);
```

Part4: Program Environment

Virtual machine application: VM Ware fusion 11

The program is run on a Ubuntu 16.04 LTS operation system, with kernel version 4.10.14.

Compiler: gcc version 5.4.0

Part5: How to run my program

Program1:

```
cd ./program1
make
./program1 filename
```

Program2:

```
cd ./program2
gcc test.c -o test
make
insmod program2.ko
rmmod program2.ko
dmesg | tail -n 10
```

(You might need to export functions in linux kernel and recompile first)

Appendix

```
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 abort
Process start to fork
I'm the parent process, my pid = 30202
I'm the child process, my pid = 30203
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGABRT program

Parent process receives the SIGCHLD signal
child process get SIGABRT signal
child process is abort by abort signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 alarm
Process start to fork
I'm the parent process, my pid = 30205
I'm the child process, my pid = 30206
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGALRM program

Parent process receives the SIGCHLD signal
child process get SIGALRM signal
child process is abort by alarm signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 bus
Process start to fork
I'm the parent process, my pid = 30207
I'm the child process, my pid = 30208
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGBUS program

Parent process receives the SIGCHLD signal
child process get SIGBUS signal
child process is abort by bus error signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 floating
Process start to fork
I'm the parent process, my pid = 30214
I'm the child process, my pid = 30215
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGFPE program

Parent process receives the SIGCHLD signal
child process get SIGFPE signal
child process is abort by floating error signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 hangup
Process start to fork
I'm the parent process, my pid = 30219
I'm the child process, my pid = 30220
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGHUP program

Parent process receives the SIGCHLD signal
child process get SIGHUP signal
child process is abort by hung up signal
CHILD EXECUTION FAILED!!
```

```
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 illegal_instr
Process start to fork
I'm the parent process, my pid = 30226
I'm the child process, my pid = 30227
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGILL program

Parent process receives the SIGCHLD signal
child process get SIGILL signal
child process is abort by illegal signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 interrupt
Process start to fork
I'm the parent process, my pid = 30231
I'm the child process, my pid = 30232
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGINT program

Parent process receives the SIGCHLD signal
child process get SIGINT signal
child process is abort by keyboard interrupt signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 kill
Process start to fork
I'm the parent process, my pid = 30235
I'm the child process, my pid = 30236
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGKILL program

Parent process receives the SIGCHLD signal
child process get SIGKILL signal
child process is abort by kill signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 normal
Process start to fork
I'm the parent process, my pid = 30239
I'm the child process, my pid = 30240
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the normal program

------------CHILD PROCESS END------------
Parent process receives the SIGCHLD signal
Normal termination with EXIT STATUS = 0
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 pipe
Process start to fork
I'm the parent process, my pid = 30245
I'm the child process, my pid = 30246
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGPIPE program

Parent process receives the SIGCHLD signal
child process get SIGPIPE signal
child process is abort by broken pipe signal
CHILD EXECUTION FAILED!!
```

```
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 quit
Process start to fork
I'm the parent process, my pid = 30249
I'm the child process, my pid = 30250
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGQUIT program

Parent process receives the SIGCHLD signal
child process get SIGQUIT signal
child process is abort by quit signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 segment_fault
Process start to fork
I'm the parent process, my pid = 30256
I'm the child process, my pid = 30257
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSEGV program

Parent process receives the SIGCHLD signal
child process get SIGSEGV signal
child process is abort by quit signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 stop
Process start to fork
I'm the parent process, my pid = 30261
I'm the child process, my pid = 30262
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSTOP program

Parent process receives the SIGCHLD signal
child process get SIGSTOP signal
child process stopped
CHILD EXECUTION STOPPED
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 terminate
Process start to fork
I'm the parent process, my pid = 30267
I'm the child process, my pid = 30268
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTERM program

Parent process receives the SIGCHLD signal
child process get SIGTERM signal
child process is abort by terminate signal
CHILD EXECUTION FAILED!!
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program1$ ./program1 trap
Process start to fork
I'm the parent process, my pid = 30273
I'm the child process, my pid = 30274
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTRAP program

Parent process receives the SIGCHLD signal
child process get SIGTRAP signal
child process is abort by trap signal
CHILD EXECUTION FAILED!!
```

```
zhongkaining@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2$ sudo su
[sudo] password for zhongkaining:
root@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2# make
make -C /lib/modules/4.10.14/build M=/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2 modules
make[1]: Entering directory '/home/zhongkaining/Desktop/linux-4.10.14'
  CC [M]  /mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.o
/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.c: In function 'my_wait':
/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.c:148:5: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
     int a;
     ^
/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.c: In function 'my_fork':
/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.c:173:5: warning: ISO C90 forbids mixed declarations and code [-Wdeclaration-after-statement]
     pid_t pid;
     ^
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.mod.o
  LD [M]  /mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2/program2.ko
make[1]: Leaving directory '/home/zhongkaining/Desktop/linux-4.10.14'
root@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2# insmod program2.ko
root@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2# rmmod program2.ko
root@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2# dmesg | tail -n 10
[31156.589962] [program2] : Module init
[31156.589966] [program2] : Module_init create kthread start
[31156.594395] [program2] : Module_init kthread starts
[31156.595212] [program2] : The child process has pid= 31644
[31156.595213] [program2] : The parent process has pid= 31642
[31156.598542] [program2] : child process
[31156.732337] [program2] : get SIGBUS signal
[31156.732340] [program2] : child process has bus error
[31156.732341] [program2] : The return signal is 7
[31159.744433] [program2] : Module exit
root@ubuntu:/mnt/hgfs/CSC3150/Project/CSC3150_Assignment_1/source/program2#
```