

# **Signals**

## **LAB # 11**



**Fall 2023**

### **CSE-302L Systems Programming Lab**

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **C**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

**Engr. Abdullah Hamid**

Date:

**1<sup>st</sup> February 2024**

**Department of Computer Systems Engineering  
University of Engineering and Technology, Peshawar**

### Task 1:

Implement wait ( ) function

- a. By changing the default behavior of SIGCHLD (without using pause or sigsuspend or sigwait)
- b. Using pause ( ) function
- c. Using signal suspend option
- d. Using sigwait

### Part a:

Code:

```
t4.c
1 #include <stdlib.h>
2 #include <stdio.h>
3 #include <unistd.h>
4 #include <dirent.h>
5 #include <sys/stat.h>
6 #include <signal.h>
7
8 int x;
9 void mywait()
10 {
11     while (x == 0)
12         ;
13     printf("parent terminated\n");
14 }
15 void myhandler(int no)
16 {
17     printf("child terminated\n");
18     x = 1;
19 }
20 int main()
21 {
22
23     struct sigaction act;
24     act.sa_handler = myhandler;
25     sigaction(SIGCHLD, &act, NULL);
26
27     int y = fork();
28     if (y > 0)
29         mywait();
30 }
```

## Output:

```
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t1.o
child terminated
parent terminated
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$
```

## Part b:

## Code:

```
6 sigset_t myset;
7 int x;
8 void my_wait(){
9
10     if(x > 0){
11
12         sigdelset(&myset, SIGCHLD); //UNBLOCK SIGCHLD
13         sigprocmask(SIG_SETMASK, &myset, NULL);
14         pause();
15         printf("\nParent has been terminated..\n");
16     }
17 }
18
19 }
20
21 void my_handler(int sig_no){
22
23     printf("\nChild has been terminated..\n");
24 }
25
26 int main(int argc, char* argv[]){
27
28     struct sigaction my_action;
29
30     my_action.sa_flags = 0;
31     my_action.sa_handler = my_handler;
32     sigemptyset(&my_action.sa_mask);
33
34     sigemptyset(&myset);
35     sigfillset(&myset); //BLOCK ALL SIGNALS
36
37     sigaction(SIGCHLD, &my_action, NULL);
38
39     sigprocmask(SIG_SETMASK, &myset, NULL);
40
41     x = fork();
42
43     if(x > 0){
44         printf("\nParent Waiting..\n");
45         my_wait();
46         printf("\nParent Waited succesfully..\n");
47     }
48
49     return 0;
50 }
```

## Output:

```
ali@Ubuntu: ~/Desktop/SP Lab/Lab 11
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t1.o
Parent Terminatinnngg..
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t2.o
Parent Waiting..
Child has been terminated..
Parent has been terminated..
Parent Waited succesfully..
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$
```

## Part c:

### Code:

```
1 #include<stdlib.h>
2 #include<unistd.h>
3 #include<signal.h>
4
5
6
7 sigset_t myset;
8 int x;
9
10 void my_wait(){
11     if(x > 0)
12         sigsuspend(&myset);
13 }
14
15
16 void my_handler(int sig_no){
17     printf("\nChild has been terminated..\n");
18     //raise(sig_no);
19 }
20
21
22 int main(int argc, char* argv){
23     struct sigaction my_action;
24
25     my_action.sa_flags = 0;
26     my_action.sa_handler = my_handler;
27     sigemptyset(&my_action.sa_mask);
28
29     sigemptyset(&myset);
30     sigfillset(&myset); //BLOCK ALL SIGNALS
31     sigdelset(&myset, SIGCHLD);
32
33     sigaction(SIGCHLD, &my_action, NULL);
34
35     x = fork();
36
37     if(x > 0){
38         printf("\nParent Waiting..\n");
39         my_wait();
40     }
41
42     return 0;
43 }
44 }
```

## Output:

```
oifali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t3.o
Parent Waiting..
CHild has been terminated..
oifali@Ubuntu:~/Desktop/SP Lab/Lab 11$
```

## Part d:

## Code:

```
2 #include<stdlib.h>
3 #include<unistd.h>
4 #include<signal.h>
5 #include<string.h>
6
7 sigset_t myset;
8 int x;
9
10 void my_wait(){
11     int y;
12     if(x > 0)
13         sigwait(&myset, &y);
14 }
15
16 void my_handler(int sig_no){
17
18     printf("\nCHild has been terminated..\n");
19     printf("%s\n", strdup(sys_siglist[sig_no]));
20 }
21
22 int main(int argc, char* argv[]){
23
24     struct sigaction my_action;
25
26     my_action.sa_flags = 0;
27     my_action.sa_handler = my_handler;
28     sigemptyset(&my_action.sa_mask);
29
30     sigemptyset(&myset);
31     sigaddset(&myset, SIGCHLD);
32     sigaction(SIGCHLD, &my_action, NULL);
33
34     x = fork();
35
36     if(x > 0){
37         printf("\nParent Waiting..\n");
38         my_wait();
39         printf("\nParent Waited succesfully..\n");
40     }
41
42     else{
43         sleep(2);
44     }
45     return 0;
46 }
```

### Output:

```
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t4.o  
  
Parent Waiting..  
  
Parent Waited succesfully..  
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$  
    sigemptyset(&my_action.sa_mask);
```