P1.c

1 //example for volatile

2 //$cc -O1 p1.c

3 #include<stdio.h>

4 #include<signal.h>

5 volatile int flag=0;

6 void isr(int n)

7 {

8 printf("in isr...\n");

9 printf("flag value updated...\n");

10 flag=1;

11 printf("isr completed...\n");

12 }

13 main()

14 {

15 printf("process started execution...\n");

16 signal(2,isr);

17 while(flag==0)

18 {

19 //some code executing until condition false

20 }

21 printf("process termination...\n");

22 }

P2.c

1 //observe that when child process terminated parent process receives SIGCHLD signal?

2 #include<stdio.h>

3 #include<signal.h>

4 void isr(int n)

5 {

6 printf("in isr...\n");

7 wait(0);

8 }

9 main()

10 {

11 if(fork()==0)

12 {

13 printf("in child:pid->%d\n",getpid());

14 sleep(10);

15 printf("child termination...\n");

16 exit(0);

17 }

18 else

19 {

20 signal(17,isr);

21 while(1);

22 }

23 }