

TI DSP, MCU 및 Xilinx Zynq FPGA

프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)

gcccompil3r@gmail.com

학생 – 문한나

mhn97@naver.com

내용정리

예제1)

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
#include <dirent.h>
#include <sys/stat.h>
#include <string.h>

void recursive_dir(char *dname); /*컴파일러 해석 위에서 아래
                                   이게 선언되어 있으면 아래도 읽어본다
                                   없으면 오류 (프로토타입 기술)
                                   이것저것 하다 보면 꼬일 수 있으니까*/

int main(int argc, char *argv[]){

    recursive_dir(".");

    return 0;

}

//dname에는 이름만 있다. 시스템 콜 통해서 포인터를 받음
void recursive_dir(char *dname){ //처음에 .이 걸림

    struct dirent *p;
    struct stat buf;
    DIR *dp;
    chdir(dname); //디렉토리 경로를 현재위치로 바꾼다
    dp = opendir("."); //디렉토리 자체 포인터 모든 것은 파일
    printf("Wt%s : Wn", dname);
    rewinddir(dp); //되감기->포인터를 맨 앞으로 되돌려 놓음
    while(p=readdir(dp)){ //이름 뿌림
        stat(p->d_name, &buf);
        if(S_ISDIR(buf.st_mode)) //실제 디렉토리인지 아닌지
            if(strcmp(p->d_name, ".") && strcmp(p->d_name, "..")) //.이랑 ..제끼자 숫자 앞
조건이 걸리면 뒤에 조건 안봄
                recursive_dir(p->d_name); //다른 디렉토리
    }

    chdir(".."); /* 상위 디렉토리로(프로그램 내부에서 이루어지는 작업)버그 나오면 고치자^^
                  만약 올라갈 때가 없으면 문제... 체크해서 마지막에서는 하지 않도록
                  이거 안쓰면 시작했던 곳에서 한단계 위에서 끝남
                  시험문제임; */
    closedir(dp); //오픈했으니 닫아주자 끝나면 리컬시브로다시(재귀니까)
```

```
mhn@mhn-900X3L: ~/my_proj/linux/22
mhn@mhn-900X3L:~$ cd my_proj/linux/22
mhn@mhn-900X3L:~/my_proj/linux/22$ gcc test1.c
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
. :
t2 :
t1 :
mhn@mhn-900X3L:~/my_proj/linux/22$ ls
a.out      ps_test2.  t2         test12.c   test3.c   test6.c   test9.c
myfifo     ps_test2.c test10.c   test1.c    test4.c   test7.c
ps_test1.c t1         test11.c  test2.c    test5.c   test8.c
mhn@mhn-900X3L:~/my_proj/linux/22$
```

예제2)

```
#include <unistd.h>
#include <stdio.h>

int main(void){

printf("before\n");
fork(); //프로세스 2개가 됨 밑에 꺼 복사
printf("after\n");

return 0;

}
```

```
mhn@mhn-900X3L: ~/my_proj/linux/22
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
before
after
after
mhn@mhn-900X3L:~/my_proj/linux/22$
```

예제3)

```
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>

int main(void){

    pid_t pid; //자료형이 int와 같음 그냥 pid파일이라서
    pid = fork(); //복사 자식의 pid값 리턴 부모는 0보다 큼

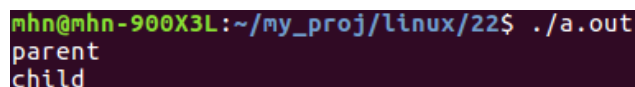
    if(pid>0) //부모
        printf("parent\n");
    else if(pid == 0) //자식
        printf("child\n");
    else{

        perror("fork()"); //에러메시지 출력
        exit(-1);

    }

    return 0;

}
```



```
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
parent
child
```

예제4)

```
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>

int main(void){

    pid_t pid;

    printf("getpid : %d\n",getpid());

    pid = fork(); //자식 프로세스 생성
    if(pid > 0) //부모
        printf("parent : pid = %d, cpid = %d\n", getpid(),pid);
    else if(pid == 0) //자식
        printf("child : pid = %d, cpid = %d\n", getpid(),pid);
    else{

        perror("fork() ");

    }

}
```

```

        exit(-1);
    }

    return 0;
//getpid()-> 자기자신의 pid값 가져옴 / pid 리턴값은 자식의 pid값
}

```

```

mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
getpid : 2821
parent : pid = 2821, cpid = 2822
child : pid = 2822, cpid = 0
mhn@mhn-900X3L:~/my_proj/linux/22$ █

```

예제5)

```

#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>

int main(void){

    pid_t pid;
    int i;
    pid = fork();
    if(pid > 0){ //대문자는 부모

        while(1){
            for(i=0;i<26;i++){
                printf("%c ",i+'A');
                fflush(stdout); //시스템 내부에 있는 출력 버퍼 좀 비워라
            }
        }
    }else if(pid == 0){ //소문자는 자식

        while(1){
            for(i=0;i<26;i++){
                printf("%c",i+'a');
                fflush(stdout);
            }
        }
    }
    else{
        perror("fork() ");
        exit(-1);
    }

    printf("Wn");

    return 0;
}

```

```
//대문자는 부모가 소문자는 자식이
//프로세스 같이 돌아가고 있으니깐 섞여서 나온다
//context_switching
```

```
mhn@mhn-900X3L: ~/my_proj/linux/22
r u sv tw ux vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mp nq or ps qt r
u sv tw ux vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mp nq or ps qt r
sv tw ux vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mp nq or ps qt ru sv
tw ux vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mp nq or ps qt ru sv t
w ux vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mp nq or ps qt ru sv tw
ux vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mp nq or ps qt ru sv tw ux
vy wz xa yb zc ad be cf dg eh fi gj hk il jm kn lo mnp oq pr qs rt su tv uw vx
wy xz ya zb ac bd ce df eg fh gi hj ik jl km ln mo nop pq qr rs st tu uv vw wx x
y yz za ab bc cd de eff gg hh ii jj kk ll mm nn oo pp qq rr ss tt uu vv ww xx yy
za zb ba cb dc ed fe gf hg ih ji kj lk ml nom pn qo rp sq tr us vt wu xv yw zx ay
bz ca db ec fghij d ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx
ey fz ga hb ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey
fz ga hb ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey f
Z ga hb ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz
ga hb ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga
hb ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga h
B ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb
ic jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic
jd ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic j
D ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd
ke lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke
lf mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke l
F mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf
mg nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf mg
nh oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf mg n
H oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf mg nh
oi pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf mg nh oi
pj qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf mg nh oi p
J qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lf mg nh oi pj
qk rl sm tn uo vp wq xr ys zt au bv cw dx ey fz ga hb ic jd ke lmnf og ph qi rj
sk tl um vn wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk
tl um vn wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk t
L um vn wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl
um vn wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um
vn wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um v
N wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn
wo xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo
xp yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo x
P yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo xp
yq zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo xp yq
zr as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo xp yq z
R as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo xp yq zr
as bt cu dv ew fx gy hz ia jb kc ld me nf og ph qi rj sk tl um vn wo xp yq zr as
bt cu dv ewfx gy hz ia jb kc ld me nf og ph qi rj sk tl uvwxm N ^Z
[1]+  Stopped                  ./a.out
```

예제6)

```
#include <unistd.h>
#include <stdio.h>
#include <errno.h>
#include <stdlib.h>

int global = 100;

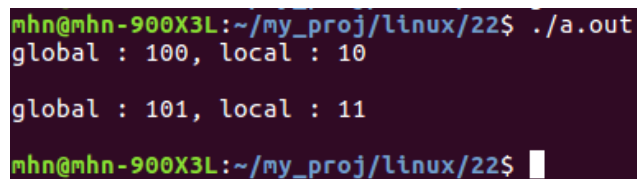
int main(void){

    int local = 10;
    pid_t pid;
    int i;
    pid = fork(); //초기상태에서 복사

    if(pid>0){//부모
        printf("global : %d, local : %d\n",global,local); //있는거 뿌림
    }
    else if(pid == 0){//자식
        global++;
        local++;
        printf("global : %d, local : %d\n",global,local); //++되서 뿌림
    }
    else{
        perror("fork() ");
        exit(-1);
    }

    printf("\n");

    return 0;
}
```



```
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
global : 100, local : 10

global : 101, local : 11
mhn@mhn-900X3L:~/my_proj/linux/22$
```

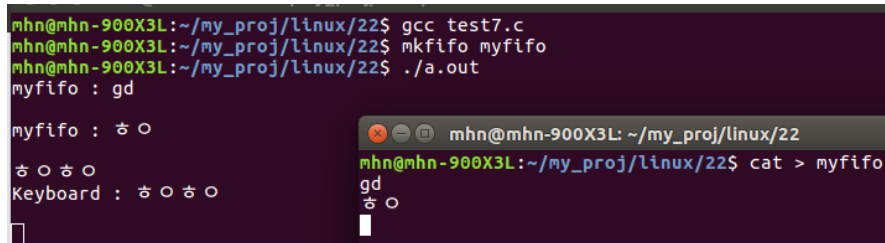
예제7)

```
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

int main(void){

    int fd,ret;
    char buf[1024];
    pid_t pid;
    fd = open("myfifo",O_RDWR);
    if((pid = fork()) > 0){
        for(;;){
            ret = read(0,buf,sizeof(buf));
            buf[ret]=0;
            printf("Keyboard : %s\n",buf);
        }
    }
    else if(pid == 0){
        for(;;){
            ret = read(fd,buf,sizeof(buf));
            buf[ret] = 0;
            printf("myfifo : %s\n",buf);
        }
    }else{
        perror("fork() ");
        exit(-1);
    }
    close(fd);
    return 0;
}
```

//논블로킹안썼는데 그렇게 나옴 이유는?
//프로세서가 두개라서
//각자 할 일이 분할됨
//키보드는 부모가 파이프는 자식이 처리함
//그러니까 문제없음



```
mhn@mhn-900X3L:~/my_proj/linux/22$ gcc test7.c
mhn@mhn-900X3L:~/my_proj/linux/22$ mkfifo myfifo
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
myfifo : gd

myfifo : ㅎㅇ
ㅎㅇㅎㅇ
Keyboard : ㅎㅇㅎㅇ

```


예제8)

```
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
```

```
int main(void){

    pid_t pid;
    if((pid = fork()) > 0)
        sleep(1000);
    else if(pid == 0);
    else{
        perror("fork() ");
        exit(-1);
    }
    return 0;
}
```

//실질적으로 이러한 프로세스를 좀비 프로세스라고 한다
//자식이 죽었는데 부모가 자고 있음 그래서 자식이 수습 해달라고 떠돌아다님 -> 좀비
//만약 오랫동안 수습 안해주면 신이 수습해줌
//좀비가 되려면 부모가 자도 되고 같이 죽어도 됨
//부모랑 자식이 죽으면 할아버지가 수습함
//여기서는 부모가 자고 일어나면 수습함(defunct가 없어짐)

예제9)

```
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
```

```
int main(void){

    pid_t pid;
    int status;
    if((pid = fork()) > 0){

        wait(&status); //자식의 상태를 봄(어떻게 죽었는지)
        printf("status : %d\n",status);

    }

    else if(pid == 0)
        exit(7);
    else {
        perror("fork() ");
    }
}
```

}

//프로세스는 시그널을 맞으면 죽음

//시그널의 종류가 많음

```
#include <stdio.h>
```

```
#include <errno.h>
```

```
#include <fcntl.h>
```

```
#include <stdlib.h>
```

```
#include <unistd.h>
```

```
#include <sys/types.h>
```

```
#include <sys/wait.h>
```

```
int main(void){
```

```
pid_t pid;
```

```
int status;
```

```
if((pid = fork()) > 0){
```

```
wait(&status);
```

```
printf("status : 0x%x\n", (status >> 8) & 0xff);
```

}

```
else if(pid == 0)
```

```
exit(7);
```

```
else {
```

```
per ror( "fork() ");
```

```
exit(-1);
```

}

```
return 0;
```

$$\}$$

//정상종료는 앞을 땡겨서 봄 8비트

정상종료



비정상종료



```
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
status : 0x7
```

예제11)

```
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main(void){

    pid_t pid;
    int status;
    if((pid = fork()) > 0){

        wait(&status);
        printf("status : 0x%x\n",WEXITSTATUS(status));

    }

    else if(pid == 0)
        exit(7);
    else {
        perror("fork() ");
        exit(-1);
    }
    return 0;
}
```

```
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
status : 0x7
```

예제12)

```
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>

int main(void){

    pid_t pid;
    int status;
    if((pid = fork()) > 0){

        wait(&status);
        printf("status : 0x%x\n", (status & 0x7f));

    }

    else if(pid == 0)
        abort();
    else {
        perror("fork() ");
        exit(-1);
    }
    return 0;
}
```

```
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out
status : 0x6
```

시그널 종류) kill -l

```
mhn@mhn-900X3L: ~
mhn@mhn-900X3L:~$ kill -l
 1) SIGHUP      2) SIGINT      3) SIGQUIT     4) SIGILL      5) SIGTRAP
 6) SIGABRT     7) SIGBUS     8) SIGFPE      9) SIGKILL     10) SIGUSR1
11) SIGSEGV    12) SIGUSR2   13) SIGPIPE    14) SIGALRM    15) SIGTERM
16) SIGSTKFLT 17) SIGCHLD  18) SIGCONT    19) SIGSTOP    20) SIGTSTP
21) SIGTTIN    22) SIGTTOU  23) SIGURG     24) SIGXCPU    25) SIGXFSZ
26) SIGVTALRM 27) SIGPROF  28) SIGWINCH   29) SIGIO      30) SIGPWR
31) SIGSYS     34) SIGRTMIN 35) SIGRTMIN+1 36) SIGRTMIN+2 37) SIGRTMIN+3
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9 56) SIGRTMAX-8 57) SIGRTMAX-7
58) SIGRTMAX-6 59) SIGRTMAX-5 60) SIGRTMAX-4 61) SIGRTMAX-3 62) SIGRTMAX-2
63) SIGRTMAX-1 64) SIGRTMAX
mhn@mhn-900X3L:~$
```

Quiz) ls-alr구현해보기

```
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <pwd.h>
#include <grp.h>
#include <time.h>
#include <dirent.h>

int main(int argc, char **argv){

    DIR *dp;
    struct stat buf;
    struct dirent *p;
    struct passwd *pw;
    struct group *gr;
    struct tm *tm;
    char ch;
    char perm[11] = "-----";
    char rwx[4] = "rwx";
    // char sst[4] = "sst";
    int i;
    // dp = opendir(".");
    stat(argv[1], &buf);
    if(S_ISDIR(buf.st_mode))
        perm[0] = 'd';
    if(S_ISREG(buf.st_mode))
        perm[0] = '-';
    if(S_ISFIFO(buf.st_mode))
        perm[0] = 'p';
    if(S_ISLNK(buf.st_mode))
        perm[0] = 'l';
    if(S_ISSOCK(buf.st_mode))
        perm[0] = 's';
    if(S_ISCHR(buf.st_mode))
        perm[0] = 'c';
    if(S_ISBLK(buf.st_mode))
        perm[0] = 'b';

    for(i=0; i<9; i++)
        if((buf.st_mode >> (8-i)) & 1)
            perm[i+1] = rwx[i%3];

    /*
    for(i=0; i<3; i++){
        if((buf.st_mode >> (11-i)) & 1)
            if(perm[(i+1) * 3] == '-')
                perm[(i+1)*3] = sst[i] ^ 0x20;
            else perm[(i+1) * 3] = sst[i];
    }*/

    printf("%s ", perm);
    printf("%d ", buf.st_nlink);
    pw = getpwuid(buf.st_uid);
    printf("%s ", pw->pw_name);
    gr = getgrgid(buf.st_gid);
    printf("%s ", gr->gr_name);
```

```
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out test1.c
-rw-rw-r-- 1 mhn mhn 1555 2018-03-25 09:25 test1.c
mhn@mhn-900X3L:~/my_proj/linux/22$ ./a.out t1
drwxrwxr-x 2 mhn mhn 4096 2018-03-25 09:26 t1
mhn@mhn-900X3L:~/my_proj/linux/22$
```

두 번째로, 이 연구의 한 가지 한계는, 이 연구가 횡단연구라는 점이다. 이 연구는 2014년 12월과 2015년 12월 두 시점에서만 자료를 수집하였다. 이 때문에 이 연구는 인과관계를 밝히는데 한계가 있다. 이 연구의 한계점을 보완하기 위하여, 이 연구는 2016년 12월과 2017년 12월 두 시점에서 자료를 수집하여, 이 연구의 한계점을 보완할 예정이다.

