# TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 최대성
c3d4s19@naver.com

2018.03.28. – 024일차 수업

---

* 함수 안에서 main()으로 점프하는 함수

goto의 경우 함수 안에서 main으로 이동이 불가하다

<setjmp.h> 헤더 필요

jmp_buf [env]

-> 점프하기 위해서 구조체부터 선언해야함

setjmp([env])

-> 처음 리턴값 0, 이후 리턴값은 longjmp()의 두번째 인자에서 설정, longjmp()실행 전에 setjmp()가 먼저 실행되어야 한다.

longjmp([env], [리턴값])

-> setjmp([env])위치로 점프(goto)함 ,setjmp([env])가 [리턴값]을 반환하도록 만듬

<setjmp.h> 사용 예제

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

jmp_buf env;

void test(){
        longjmp(env, 1);
        printf("test!₩n"); //이거 안나옴
}

int main(){
        int ret;
        if((ret = setjmp(env)) == 0){
                printf("this₩n");
                test();
        }
        else if(ret > 0)
                printf("error₩n");
        return 0;
}
```

---

signal() 함수

-> <signal.h> 헤더 필요

signal([시그널 종류], [실행할 함수 포인터])

-> 바로 이전에 실행했던 함수 포인터 반환

이전에 실행한 signal()함수가 없으면 NULL 반환

(리눅스 명령어 kill –l로 [시그널 종류] 볼 수 있음)

[실행할 함수 포인터] -> void (*func)(int)형

func([시그널 번호])

-> 이런식으로 시그널 없이 함수실행 가능

ex) my_sig(SIGALRM);

signal() 함수 사용 예제

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

void my_sig(int signo){
        printf("my_sig called₩n");
}

void my_sig2(int signo){
        printf("my_sig2 called₩n");
}

int main(){
        void (*old_p)(int);
        void (*old_p2)(int);
        //signal()은 이전에 실행했던 함수 반환
        old_p = signal(SIGINT, my_sig);
        pause(); //signal올때까지 대기
        old_p2 = signal(SIGINT, my_sig2);
        pause();
        old_p2 = signal(SIGINT, old_p2);
        pause();
        old_p2 = signal(SIGINT, old_p2);
        pause();
        for(;;)
                pause();
        return 0;
}
```

signal() 함수 이용한 업다운 게임

```
#include <stdio.h>
#include <signal.h>
#include <fcntl.h>
#include <stdlib.h>
#include <time.h>
#include <setjmp.h>
#include <unistd.h>
void my_sig(int signo){
        printf("Game Over!₩n");
        exit(0);
}

int main(){
        srand(time(NULL));
        jmp_buf env;
        char buf[1024];
        int ran, i, start = 1, num = 100;
        signal(SIGALRM, my_sig);
        ran = rand() % num + start;
        for(int t = 0; t < 5; t++){
                alarm(3);
                printf("숫자 입력(%d ~ %d):₩n",
start, num + start - 1);
                read(0, buf, sizeof(buf));
                i = atoi(buf);
                alarm(0); //알람제거
                if(ran > i){
                        printf("Up₩n");
                }
                else if(ran < i){
                        printf("Down₩n");
                }
                else{
                        printf("Win!₩n");
                        return 0;
                }
        }
        my_sig(SIGALRM);
        //리눅스 kill -l로 시그널 번호 확인가능

        return 0;
}
```

alarm([알람시간]);

-> [알람시간] 후에 시그널 SIGALRM 발생

alarm(0); 인 경우는 기존 알람 취소함