TI DSP,MCU 및 Xilinux Zynq FPGA

프로그래밍 전문가 과정

이름	문지희
학생 이메일	mjh8127@naver.com
날짜	2018/3/27
수업일수	24 일차
담당강사	Innova Lee(이상훈)
강사 이메일	gcccompil3r@gmail.com

목차

1. wait( )

1. signal

~소스코드

```
#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)
{
        void (*old_p)(int);//함수포인터
        void (*old_p2)(int); //함수포인터
        old_p=signal(SIGINT,my_sig);//my_sig 동작
        pause();
        old_p2=signal(SIGINT,my_sig2);//my_sig2동작
        pause();
```

```
//      old_p2=signal(SIGINT,old_p);//old_p 에 뭔가 없다. 그래서 ctrl+c 3번 누르면
꺼짐
          old_p2=signal(SIGINT,old_p2);
          pause();
          for(;;)
                    pause();
          return 0;
}
```

```
~old_p2=signal(SIGINT,old_p2); 로 바꾸기 전
^Cmy_sig called
^Cmy_sig2 called
^C

~old_p2=signal(SIGINT,old_p2); 로 바꿨을 때
^Cmy_sig called
^Cmy_sig2 called
^Cmy_sig called
^Cmy_sig called
^Cmy_sig called
^Cmy_sig called
^Cmy_sig called
```

signal 이 뭔가를 리턴해서 무한반복된다.

signal 은 바로 한단계 전에 등록한걸 리턴한다. 처음 signal 은 아무것도 없어서 null 이 들어갔었는데 나중엔 old_p 가 들어가서 계속 my_sig 가 호출됨.

2. kill -9

~소스코드

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

int main(void)
{
        signal(SIGINT,SIG_IGN);
        pause();
        return 0;
}
```

~결과

```
xeno@xeno-NH:~/proj/0327$ ps -ef | grep a.out
xeno        3695   3550   0 10:10 pts/2    00:00:00 ./a.out
xeno        3898   3864   0 10:24 pts/12   00:00:00 grep --color=auto a.out
xeno@xeno-NH:~/proj/0327$ kill -9 3695
xeno@xeno-NH:~/proj/0327$ ps -ef | grep a.out
xeno        3900   3864   0 10:24 pts/12   00:00:00 grep --color=auto a.out
```

3. goto

~소스코드

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

int main(void)
{
        int ret;
        char buf[1024]="asdf";
        func();
        if(ret=read(0,buf,sizeof(buf))>0)//키보드 입력을 받아 buf 에 넣음
                                         //0보다 크면
                                         //buf 의 사이즈 값을 ret 에 입력받음
                goto err;//err 로 이동

        printf("실행 x");
        return 0;



        err:
        perror("read()");//()안의 시스템 콜에 대한 동작을 설명
        exit(-1);
}
```

~결과

```
xeno@xeno-NH:~/proj/0327$ ./a.out

read(): Success
```

4.goto 는 그 함수 내에서만 사용가능

~소스코드

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

void func(void)
{
goto err;
}

int main(void)
{
        int ret;
        char buf[1024]="asdf";
        func();
```

-goto 가 그 함수 내에 있을 때

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

int func(void)
{
        goto err;

        return 0;
        err:
        printf("aa");
        exit(-1);

}
```

```
        return 0;

        err:
        printf("실행 x");
        exit(-1);
}
```

```
int main(void)
{
        int ret;
        char buf[1024]="asdf";
        func();

        return 0;

}
```

~결과

```
xeno@xeno-NH:~/proj/0327$ gcc 3.c
3.c: In function ‘func’:
3.c:9:1: error: label ‘err’ used but not defined
 goto err;
 ^
```

~결과

```
aa
```

goto 는 스택을 해지할 수 있는 방법이 없음 그래서 main 의 err 로 갈 수 없다.

5. setjmp

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

jmp_buf env;//전역변수

void test(void)
{
	longjmp(env,1);//setjum 의 리턴값이 1이 될것이다.
}

int main(void)
{
	int ret;
	if((ret=setjmp(env))==0)//goto 의 레이블
				//무조건 처음엔 리턴값이 0이나오게 됨
				//그래서 if 문 만족
				//err: 을 setjmp 가 대신
	{
//			printf("test₩n");
			test();
	}
	else if(ret>0)
			printf("error₩n");
	return 0;
```

```
}
```

```
~ 바꾸기 전
error

~바꾼 후
test
error
```

6. setjmp - 강사님 코드

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

jmp_buf env1;
jmp_buf env2;

void test1(void)
{
        longjmp(env1, 1); //goto 랑 동일한 역할을 함
                          //setjmp 는 goto 의 레이블
                          // env1인 setjmp 에 ret 에 1을 반환
}

void test2(void)
{
```

```
        longjmp(env1, 2); // env1인 setjmp 에 ret 에 2을 반환
}

void test3(void)
{
        longjmp(env2, 1); // env2인 setjmp 에 ret 에 1을 반환
}

int main(void)
{
        int ret;
        if((ret = setjmp(env1)) == 0)
        {
                printf("this₩n");
                test1(); //ret 에 1값이 들어가고 다시 if 문을 훑음
        }
        else if(ret == 1)
        {
                printf("1₩n");
                test2(); //ret 에 2값이 들어가고 다시 if 문을 훑음
        }
        else if(ret == 2)
                printf("2₩n"); //2를 출력하고 다음 if 문으로 들어간다
        else
        {
                printf("goto letsgo label₩n");
```

```
                goto letsgo; //letsgo 로 간다
        }



        if((ret = setjmp(env2)) == 0) // 처음엔 무조건 0이 되어 실행됨
        {
                printf("second label₩n");
                test3(); //test3을 실행하면 setjmp(env2)인 2번째 if 문을 다시 돌음
        }
        else
                longjmp(env1, 3); setjmp(env1)인 첫번째 if 문을 다시 돌게된다.
letsgo:
                goto err; //err 로 간다

        return 0;

err:
        printf("Error!!!₩n"); //문자출력
        exit(-1);
}
```

7. longjmp 순서

~소스코드

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

jmp_buf env;
void test(void)
{
	int flag=-1;
	if(flag<0)
		longjmp(env,1);//longjmp 를 하면 밑의 내용은 실행되지 않는다. 1을
반환
	printf("call test₩n");

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

~결과
error

7.alarm

~소스코드

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

void my_sig(int signo)
{
        printf("you must inser coin₩n");
        exit(0);
}

int main(void)
{

        char buf [1024];
        int ret;
        signal(SIGALRM,my_sig); //SIGALRM 이 입력되지 않았을 경우 my_sig 실행
        alarm(3); //3초가 동안 키보드 입력값을 받는다.
```

```
        read(0,buf,sizeof(buf));
        alarm(0);
        return 0;
}
```

~결과

```
xeno@xeno-NH:~/proj/0327$ ./a.out
you must inser coin
```

8. 업다운 게임

랜덤으로 숫자 1~100생성하고 숫자를 입력받고 그 숫자보다 작은지 큰지알려줌, 1초 내에 답 안넣으면 진다. 기회제한있음

~소스코드

```
#include<stdio.h>
#include<signal.h>
#include<fcntl.h>
#include<stdlib.h>
#include<time.h>
#include<setjmp.h>

jmp_buf env;

void my_sig(int signo)
{
        printf("you lose");
        exit(0);
}
```

```
int mkrand(void)
{
	int num;
	srand(time(NULL));
	num = rand()%100;

	return num;
}

void check_num(int num)
{
	int ret,r;
	char buf[1024];
	ret=read(0,buf,sizeof(buf));
	r=atoi(buf);

	if(r==num)
	{
		printf("you win₩n");
		longjmp(env,2);
	}
	else if(r>num)
		printf("down₩n");
	else if(r<num)
		printf("up₩n");
```

```
}

int main(void)
{

        char buf [1024];
        int ret,i,num;
        signal(SIGALRM,my_sig);
        num=mkrand();
        printf("1부터 100사이의 정수를 입력하시오.₩n");
        if(ret=setjmp(env)==0)
        {
                for(i=0;i<5;i++)
                {
                        printf("남은 기회 : %d ₩n",4-i);
                        alarm(1);
                        check_num(num);
                        alarm(0);
                }
        }
        return 0;
}
```

~결과

```
1부터 100사이의 정수를 입력하시오.
남은 기회 : 4
```

50
up
남은 기회 : 3
60
up
남은 기회 : 2
70
up
남은 기회 : 1
80
down
남은 기회 : 0
78
you win

~선생님이 주신 코드 해석

```
 #include <time.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <signal.h>
#include <stdlib.h>
#include <stdbool.h>

void sig_handler(int signo)
{
        printf("You lose! Input should be within 1 second!₩n");
```

```
        exit(0);
}

void make_game(int *data)
{
        *data = rand() % 100 + 1; //1~100난수생성
}

bool check_correct(int data, int cmp)
{
        if(data == cmp)//같으면
                return true; //참
        else//아니면
                return false;//거짓
}

void start_game(int data)
{
        char buf[32] = {0};
        bool fin;
        int i, cmp;

        for(i = 0; i < 10; i++)
        {
                signal(SIGALRM, sig_handler);
                printf("숫자를  맞춰봐!₩n");
```

```
                alarm(1); //1초
                read(0, buf, sizeof(buf));
                alarm(0); //0으로 초기화한다(알람을 끈다)
                        //1초내에 입력해야함
                cmp = atoi(buf); //문자열을 숫자로 바꿈

                fin = check_correct(data, cmp);

                if(fin) //값이 같으면
                {
                        printf("You Win!!!₩n");
                        exit(0);
                }
                else //다르면
                {
                        if(data > cmp)
                                printf("%d 보다 크다₩n", cmp);
                        else
                                printf("%d 보다 작다₩n", cmp);
                }
        }
        printf("You Lose!!! You Babo₩n");
}

int main(void)
{
```

```
	int data;

	srand(time(NULL)); //난수생성하기 위해 설정
	make_game(&data);

	start_game(data);

	return 0;
}
```