TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

2018-03-19 (18일차)

강사: Innova Lee(이상훈)
gcccompil3r@gmail.com
학생: 정유경
ucong@naver.com

시스템 호출이란?

운영체제는 사용자모드와 커널모드로 나뉘어 동작한다
일반 응용 프로그램이 시스템의 자원을 사용하여 작업을 하려고 한다면 시스템 콜을 사용하여 작업을 한다.

시스템 호출의 세 가지 기능
1. 사용자 모드에 있는 응용 프로그램이 커널의 기능을 사용할 수 있도록 한다.
2. 시스템 호출을 하면 사용자 모드에서 커널 모드로 바뀐다.
3. 커널에서 시스템 호출을 처리하면 커널 모드에서 사용자 모드로 돌아가 작업을 계속한다.

시스템 자원에 따른 시스템 호출의 유형
1. 프로세스: fork, exec, exit, wait, getpid, getppid
2. 파일 : open, close, read, write, dup, lseek
3. 시그널: signal, alarm, kill, sleep
4. 메모리: malloc, calloc, free
5. 프로세스 간 통신: pipe, socket

1. file_io2.c

int filedes : 파일 디스크립터 열린 파일을 나타내는 번호.

if((filedes = open(pathname, O_CREAT | O_RDWR | O_EXCL, 0644))

Open( ) : 파일을 열고 열린파일의 파일 디스크립터를 반환한다.

첫번째, path가 나타내는 파일을 연다.

파일 열기에 성공하면 filedes를 반환하고, 실패하면 -1을 반환한다.

두번째, 대상파일의 입출력방식을 지정한다.

O_RDONLY, O_WRONLY, O_RDWR

다음은 선택적으로 사용 가능하다

O_EXCL: 해당파일이 이미 있으면 오류

O_CREAT: 해당파일이 없는 경우 파일 생성

O_TRUNC: 파일이 이미 있는 경우 내용을 지우고 파일 크기를 0으로 한다

세번째, 파일의 사용권한을 지정한다

O_CREAT에만 사용된다.

close(filedes);

close( ): 파일디스크립터가 나타내는 파일을 닫는다.

성공하면 0, 실패하면 -1반환

2. file_io3.c

filedes2 = creat("data2.txt", 0644);

Creat( )

path가 나타내는 파일을 생성하고 0644 권한을 부여한다.

파일생성에 성공하면 생성된 파일의 파일 디스크립터를 반환하고 실패하면 -1을 반환

Open( ) 시스템 호출과 동일하다

3. file_io4.c

```
ssize_t nread;
char buf[1024];
fdin = open("temp1.txt", O_RDONLY);
fdout = open("temp2.txt", O_WRONLY|O_CREAT|O_TRUNC,
0644);

while((nread = read(fdin, buf, 1024))>0)
```

Read( ): 파일로부터 데이터 읽기

Fdin에서 1024바이트 읽고 buf에 저장

```
if(write(fdout,buf,nread)<nread)
```

Write( ): 파일에 데이터 쓰기

Fdout에 buf에 있는 nread의 데이터 쓰기

데이터의 바이트 수 반환

4. file_io5.c

filedes = open("data1.txt", O_RDONLY);

newpos = lseek(filedes, (off_t)0, SEEK_END);

Lseek( ): 파일 내에서 위치이동

SEEK_END : 파일의 끝

printf("file size: %d₩n", newpos);

파일사이즈(newpos)를 출력한다.

5. mycp.c

int main(int argc, char**argv)

- Argc : 운영체제가 이 프로그램을 실행했을 때 전달되는 인수의 개수
- Argv : 이중 포인터로서 문자열 배열을 가리키는 포인터 char* argv[4] ={"aaa", "bbb", "ccc", "ddd"};

while((nread = read(fdin, buf, 1024))>0)

if(write(out, buf, nread) < nread)

exit(-1); // 프로세스 상태체크....????

→ 비정상종료

실행: ./a.out test.txt a.txt

6. mycat.c (system call이 속도가 훨씬 빠르다!)

1. System Call

fd=open(argv[1], O_RDONLY);

파일에 대한 디스크립터 얻는다

while(ret = read(fd, buf, sizeof(buf)))

파일디스크립터에서 buf사이즈만큼 읽는다

{

write(1, buf , ret);

0: 표준 입력(키보드)

1: 표준 출력(모니터)

2: 표준 에러(error)...?????

}

close(fd);

실행방법: mycat filename

2. .....?????

FILE *fp = fopen("mycat.c","r");

파일에 대한 디스크립터 얻는다

char buf[1024];

while(ret = fread(buf, 1, sizeof(buf), fp))

Fp를 1바이트씩 1024번 읽어서 buf에 저장

{

usleep(1000000);

1000000 Micro second (1초) Sleep

fwrite(buf, 1,ret, stdout);

stdout: 표준출력

}

fclose(fp);

7-1. myscanf.h

(실행방법: gcc test.c my_scanf.c)

1 #ifndef __MY_SCANF_H__ // 헤더파일의 기본 1,2,3

2 #define __MY_SCANF_H__ // 없어도 되나...? 없으면 중복을 허용한 것..?// 지정한 것 없으면 0으로 정의

전처리문, 헤더파일이 겹치는 것을 막기 위한 일종의 매크로

#ifndef 헤더명_H__

#endif

이전에 include되어 있으면 #endif로 점프하여 한번만 선언되도록 함

두 번 define하면 syntax error!

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

int myscanf(char*, int);
```

3 #endif

7-2. myscanf.c

```
#include <stdio.h> // <시스템 헤더>즉, 라이브러리
#include "myscanf.h" // "사용자 정의 헤더"

int main(int argc, char **argv)
{
	int fd, ret;
	char buf[1024];

	ret = read(0, buf, sizeof(buf));
	write(1, buf,ret);
	printf("ret = %d₩n",ret);
	return 0;
}
```

7-3. test.c

```
#include "myscanf.h"
int myscanf(char *buf, int size)
{
        int ret=read(0,buf,size);
        return ret;
}
```

파일분할이유...?????

test.c는 왜 있는걸까...????

8-1. wc [옵션] "파일명"

wc [옵션] "파일명"

wc 명령어는 파일 내의 라인, 단어 문자의 수를 출력

(-c : 전체 문자, -l : 전체 라인, -w : 전체 단어의 수)

```
[root@localhost ]# wc -c sendmail.cf
58290 sendmail.cf
```

```
root@localhost ]# wc test.txt
 1827  9497 58290 sendmail.cf
```

파일이 1827줄로 구성되었고, 9497개의 단어가 있으며, 58290개의 철자(문자)로 이루어졌다

8-2. mywc.c

char ch; //임시적으로 어떤 문자 왔는지 체크

if(argc !=2) // 사용법 출력(예외처리)

perror("open( ) "); // 에러발생시, "open( ) 오류메세지" 출력

while(read(fd, &ch,1))
cnt++; // 문자 수증가 (모든 문자, 공백, 숫자, 특수문자 포함)
// ch에 배열넣었으니까 주소! 전달해야 함...???
// read한번 할 때마다 fd에서 1byte씩 증가..???
// read할 것 없으면 while루프 끝난다 (공백 ' ' 과는 다르다)
파일끝에는 뭐가 있을까....???

if(ch=='₩n') // 개행하면 엔터 입력시 라인수 증가

if(ch !='₩n' && ch != '₩t' && ch !=' ')
개행, 탭, 공백도 아니고(숫자or문자)

else
단어 끝나고 공백 ' ' 일때 들어간다.
Flag를 0으로 만들어 준다