HOMEWORK

이름	문지희
날짜	2018/2/21
수업일수	1일차
담당교수	이상훈

목차

1. 항공기에서 고성능 아키텍처를 사용하지 못하는 이유

- 우주방사선 : 우주에서부터 지구로 들어오는 높은 에너지를 가진 전하를 띠는 입자와 방사능을 총틀어 말한다.
- 맥스웰 방정식 : 전기장과 자기장의 관계를 전기장의 가우스 법칙, 자기장의 가우스 법칙, 패러데이 법칙, 맥스웰이 수정한 앙페르 법칙 이 네가지의 방정식을 통하여 맥스웰 방정식을 나타내었는데 전기장과 자기장의 관계를 설명할 수 있다. 전기장과 자기장은 상호 유도관계에 있는데 전기장이 진행하면 자기장 또한 90도 방향으로 진행하게 된다.

우주에서 들어온 방사선(전하를 띄는 입자)이 지구로 들어와 입자가 이동할 때 전하를 띄고있기 때문에 전기장이 생성되고 전기장이 진행하면 자기장 또한 생성되어 전자기장이 형성된다. 전자의 움직임은 전류라고 볼 수 있는데 우주방사선은 강한 에너지를 가진 입자이기 때문에 고성능의 아키텍처에 자칫 흘러 들어가게 되면 오작동이나 고장의 원인이 될 수 있다.

2. 리눅스 명령어

Ctrl + Alt + T	터미널 열기
pwd	현재 디렉토리 표시
ls	현재 디렉토리에 있는 목록 표시
gcc [소스파일명.c]	소스파일을 컴파일 해서 a.out 이라는 실행파일 생성
gcc-o [실행파일명 소스파일명]	소스파일을 컴파일 해서 실행파일명의 실행파일 생성
gcc-o-c [실행파일명 소스파일명]	gcc-o 와 동일하게 컴파일을 하지만 디버깅이 추가됨
mkdir [이름]	디렉토리 생성
gdb [실행파일 이름]	프로그램 디버깅
cd	디렉토리명으로 디렉토리 이동
clear	화면 지우기
rm-rf [폴더명]	폴더, 파일 삭제
cp [복사할파일] [원하는이름]	복사하고자 하는 파일을 원하는 이름으로 복사
mv [이전파일명] [바꾸고자하는 파일이름]	이름을 바꾸는 명령어
:w	저장
:q	나가기
:wq	저장하고 나가기
yy	1줄 복사
y숫자y	숫자만큼 줄 복사
p	붙여넣기
dd	1줄 지우기
d숫자d	숫자만큼의 줄 키우기
:set n-	줄라인 보이기
:set ts=숫자	탭의 사이즈를 숫자만큼으로 지정
v	블록지정

* 상위, 하위 디렉토리

'..' -상위 디렉토리

'.' -현재 디렉토리

*편집모드, 명령모드

명령모드 일때는 빈페이지로 키보드 입력이 안됨

편집모드 일때는 키보드 입력 가능

[A, a, I, i] 4개의 키를 이용하여 편집모드에서 명령모드로 변환가능

esc를 누르면 편집모드에서 명령모드로 전환

* 디렉토리 이동방법

Ex) 현재 위치 : '/home/id/lecture'
test와 result라는 디렉토리 존재

1) 절대경로
'/' 최상위 루트에서 가고싶은 위치까지를 지정하는 방식
cd /home/id/lecture/test 이용하여 절대경로 방식으로 test 디렉토리로 이동

2) 상대경로
cd test 이용하여 상대경로 방식으로 test 디렉토리로 이동

* rm-rf를 사용할 때 rm-rf/를 치게되면 c드라이브 삭제될 수 있으니 주의

3. 변수와 함수

- basis of memory architecture

변수 메모리에 정보를 저장할 수 있는 공간
Stack 지역변수가 위치하는 영영
Heap 동적할당된 녀석들이 위치하는 영역
Data 전역변수 및 static으로 선언된 것들이 위치하는 영역. 초기화 되지 않은 모든 것은 0으로 저장
Text machine code가 위치하는 영역

-함수

함수란 이름 input, output으로 구성. 독립적인 기능을 가지는 구성요소

함수선언
출력의 데이터타입 함수이름
{
}

- data type

Int, short, char, float, double, long double

%c char
%d int
%f float
%lf double
%llf long double
%s 문자열

-변수선언 자료형 변수이름=초기값;

Ex)

```
int num1=3;
int num2=7;
int num3=3.3;
int num4=7.7;
```

(예제)

```
#include <stdio.h>

int main(void) {
int num1=3;
int num2=7;
float num3=3.3;
```

```
double num4=7.7;

int res1;
double res2;

printf("num1=%d, num2=%d, num3=%f, num4=%f₩n",num1,num2,num3,num4);

res1=num1*num2;
res2=num1*num3;

printf("res1=%d,res2=%lf",res1,res2);
return 0;
}
```

출력값
num1=3, num2=7, num3=3.3, num4=7.7
21,9.9

예제1
// 입력=3, 출력=6

```
#include<stdio.h>


        intmyfunc(int num)
        {
        // return num * 2;
        return num + 3;
        // return num << 1;
        }

        intmain(void)
        {
        int num = 3, res;
        res = myfunc(num);
        printf("res = %d\n",
        res);

        return0;
        }
```

```c
#include<stdio.h>

// input = 3, 6, output = 21.3
floatmyfunc(int num1, int num2)
{
return num1 * num2 + 3.3;
}

intmain(void)
{
int num1 = 3, num2 = 6;
float res;

res = myfunc(num1, num2);
printf("res = %f\n", res);

printf("this = %f\n", myfunc(num1, num2));

return0;
}
```

```c
#include<stdio.h>
// input = 3, 6, output = 21.3
floatmyfunc(int num1, int num2)
{
return num1 * num2 + 3.3;
}

floattest(int a, int b)
{
printf("I can call myfunc()\n");
returnmyfunc(a, b);
}

intmain(void)
{
int num1 = 3, num2 = 6;

printf("res = %.1f\n", test(num1, num2));

return0;
}
```

```
// 2,4,6,8,10의 input, output
#include<stdio.h>
        inttest(int num1, int num2, int num3, int num4, int num5)
        {
        return num1 + num2 + num3 + num4 + num5;
        }

        intmain(void)
        {
        int num1 = 2, num2 = 4, num3 = 6, num4 = 8, num5 =
        10, res;

        res = test(num1, num2, num3, num4, num5);
        printf("res = %d\n", res);

        return0;
        }
```