Xilinx Zynq FPGA, TI DSP, MCU 기반의 프로그래밍 및 회로 설계 전문가 과정

강사 – Innova Lee(이상훈)

gcccompil3r@gmail.com

학생 – hoseong Lee(이호성)

hslee00001@naver.com

목차

1. iostream

: Input + Output + Stream → 입출력 명령을 할 때 필요함

ex) hello_world

```
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ g++ hello_world.cpp
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ ./a.out
Hello World!!
HelloWorld!!
1aString
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ cat hello_world.cpp
#include <iostream>
using std::cout;
using std::endl;

int main(void)
{

        cout << "Hello World!!" << endl;
        cout << "Hello" << "World!!" << endl;
        cout << 1 << 'a' << "String" << endl;
        return 0;
}
```

1.1 std :: cout

:: 이 기호는 어디 안에 있다고 표시한 것. std 라는 곳 안에 cout 를 사용할 것이다. cout은 화면에 보여주기 위해 입력을 하는 것.

1.2 std :: endl

endl = end +line 이 명령의 줄을 띄워라

1.3 <<

쉬프트 연산자가 아니다. 넣어준다는 의미이다

2. namespace

: std 는 standard의 약자 즉, using namespace std // 표준 네임스페이스를 사용하겠다는 의미

using namespace 선언은 말그대로 소속을 나타내는 역할을 한다. (철수네 강아지, 영희네 강아지) → 철수, 영희 - 소속

ex)

```
siyun@siyun-CR62-6M:~/my_proj$ g++ namespace.cpp
siyun@siyun-CR62-6M:~/my_proj$ ./a.out
A에서 정의한 함수
B에서 정의한 함수
siyun@siyun-CR62-6M:~/my_proj$ cat namespace.cpp
#include <iostream>
using std::cout;
using std::endl;

namespace A
{
        void test(void)
        {
                cout << "A에서 정의한 함수" << endl;
        }
}

namespace B
{
        void test(void)
        {
                cout << "B에서 정의한 함수" << endl;
        }
}

int main(void)
{
        A::test();
        B::test();
        return 0;
}
```

원본 저장하기

: namespace 를 사용하면 공간을 나눠

3. cin , cout

```
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ ./a.out
1번째 정수 입력 :10
2번째 정수 입력 :40
덧셈 결과 : 50
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ cat c_in.cpp
#include <iostream>
using std::cout;
using std::endl;
using std::cin;

int main(void)
{
        int val1, val2;
        cout << "1번째 정수 입력 :";
        cin >> val1;

        cout << "2번째 정수 입력 :";
        cin >> val2;

        int result = val1 + val2;
        cout << "덧셈 결과 : " << result << endl;
        return 0;
}
```

4. class

```
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ ./a.out
현재 문의 상태 : OPEN
현재 문의 상태 : CLOSE
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ cat class.cpp
#include <iostream>
using namespace std;

const int OPEN = 1;
const int CLOSE = 2;

class Door
{
private:
        int state;
public:
        void Open(void)
        {
                state = OPEN;
        }
        void Close(void)
        {
                state = CLOSE;
        }
        void ShowState(void){
                cout << "현재 문의 상태 : ";
                cout << ((state == OPEN) ? "OPEN" : "CLOSE") << endl;
        }
};

int main(void)
{
        Door door;

        door.Open();
        door.ShowState();
        door.Close();
        door.ShowState();
        return 0;
}
```

:

5. Abstraction

```
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ ./a.out
현재 문의 상태 : OPEN
현재 문의 상태 : CLOSE
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ cat class.cpp
#include <iostream>
using namespace std;

const int OPEN = 1;
const int CLOSE = 2;

class Door
{
private:
        int state;
public:
        void Open(void)
        {
                state = OPEN;
        }
        void Close(void)
        {
                state = CLOSE;
        }
        void ShowState(void){
                cout << "현재 문의 상태 : ";
                cout << ((state == OPEN) ? "OPEN" : "CLOSE") << endl;
        }
};

int main(void)
{
        Door door;

        door.Open();
        door.ShowState();
        door.Close();
        door.ShowState();
        return 0;
}
```

6. reference

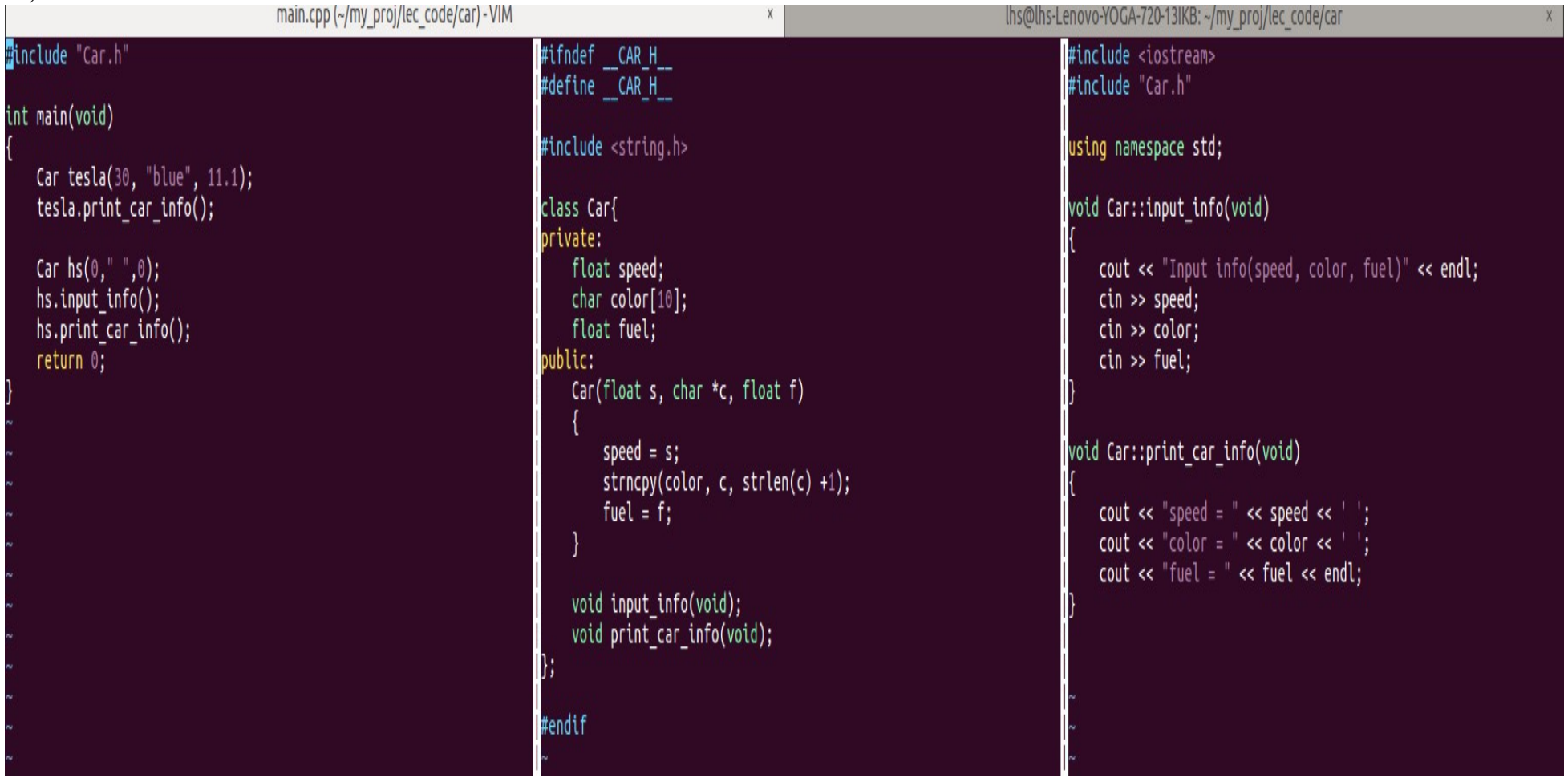
포인터랑 비슷하다. reference는 변수의 별명을 만드는 것이다. 주소를 받는게 아니고 별명을 만들어준다. Call by reference

ex) call_reference

```
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ ./a.out
val1 : 10 val2 : 20
val1 : 20 val2 : 10
lhs@lhs-Lenovo-YOGA-720-13IKB:~/my_proj/lec_code$ cat call_reference.cpp
#include <iostream>
using namespace std;

void swap(int& a, int& b)
{
        int temp = a;
        a= b;
        b= temp;

}
int main(void)
{
        int val1 = 10;
        int val2 = 20;

        cout << "val1 : " << val1 << ' ';
        cout << "val2 : " << val2 << endl;

        swap(val1,val2);
        cout << "val1 : " << val1 << ' ';
        cout << "val2 : " << val2 << endl;

        return 0;
}
```

7. 생성자

ex) construct



main.cpp (~/my_proj/lec_code/car) - VIM

```
#include "Car.h"

int main(void)
{
    Car tesla(30, "blue", 11.1);
    tesla.print_car_info();

    Car hs(0," ",0);
    hs.input_info();
    hs.print_car_info();
    return 0;
}
```

```
#ifndef __CAR_H__
#define __CAR_H__

#include <string.h>

class Car{
private:
    float speed;
    char color[10];
    float fuel;
public:
    Car(float s, char *c, float f)
    {
        speed = s;
        strncpy(color, c, strlen(c) +1);
        fuel = f;
    }

    void input_info(void);
    void print_car_info(void);
};

#endif
```

lhs@lhs-Lenovo-YOGA-720-13IKB: ~/my_proj/lec_code/car

```
#include <iostream>
#include "Car.h"

using namespace std;

void Car::input_info(void)
{
    cout << "Input info(speed, color, fuel)" << endl;
    cin >> speed;
    cin >> color;
    cin >> fuel;
}

void Car::print_car_info(void)
{
    cout << "speed = " << speed << ' ';
    cout << "color = " << color << ' ';
    cout << "fuel = " << fuel << endl;
}
```

8. 소멸자

ex) new_delete

```
#include <iostream>
#include <string.h>

using namespace std;

class Academy
{
    char *name;
    char *phone;
public:
    Academy(char *_name, char *_phone);
    ~Academy(void);
    void ShowData(void);
};

Academy::Academy(char *_name, char *_phone)
{
    name = new char[strlen(_name) + 1];
    strcpy(name, _name);

    phone = new char[strlen(_phone) +1];
    strcpy(phone, _phone);
}

Academy::~Academy(void)
{
    cout << "소멸자 호출" << endl;
    delete []name;
    delete []phone;
}

void Academy::ShowData(void)
{
    cout << "name : " << name << endl;
    cout << "phone : " << phone << endl;
}

int main(void)
{
    Academy aca("Bit", "02-111-2222");
    aca.ShowData();
    return 0;
}
```