Virtual Table

Virtual Table

Base class

```
class Base
{
public:
    virtual void VFunc1() {
        cout << "Base::VFunc1" << endl;
    }
    virtual void VFunc2() {
        cout << "Base::VFunc2" << endl;
    }
    void NonVFunc() {
        cout << "Base::NonVFunc" << endl;
    }
};
```

Virtual Table

Derived class

```
class Derived :public Base
{
public:
    virtual void VFunc1() override {
        cout << "Derived::VFunc1" << endl;
    }
    virtual void VFunc2() override {
        cout << "Derived::VFunc2" << endl;
    }
};
```

Virtual Table

main

가상 함수가 아닐 때(정적 바인딩)

```
Base * b = new Derived;

//b->NonVFunc();
__asm
{
    call Base::NonVFunc
}
```

Virtual Table

main

가상 함수일 때(동적 바인딩)

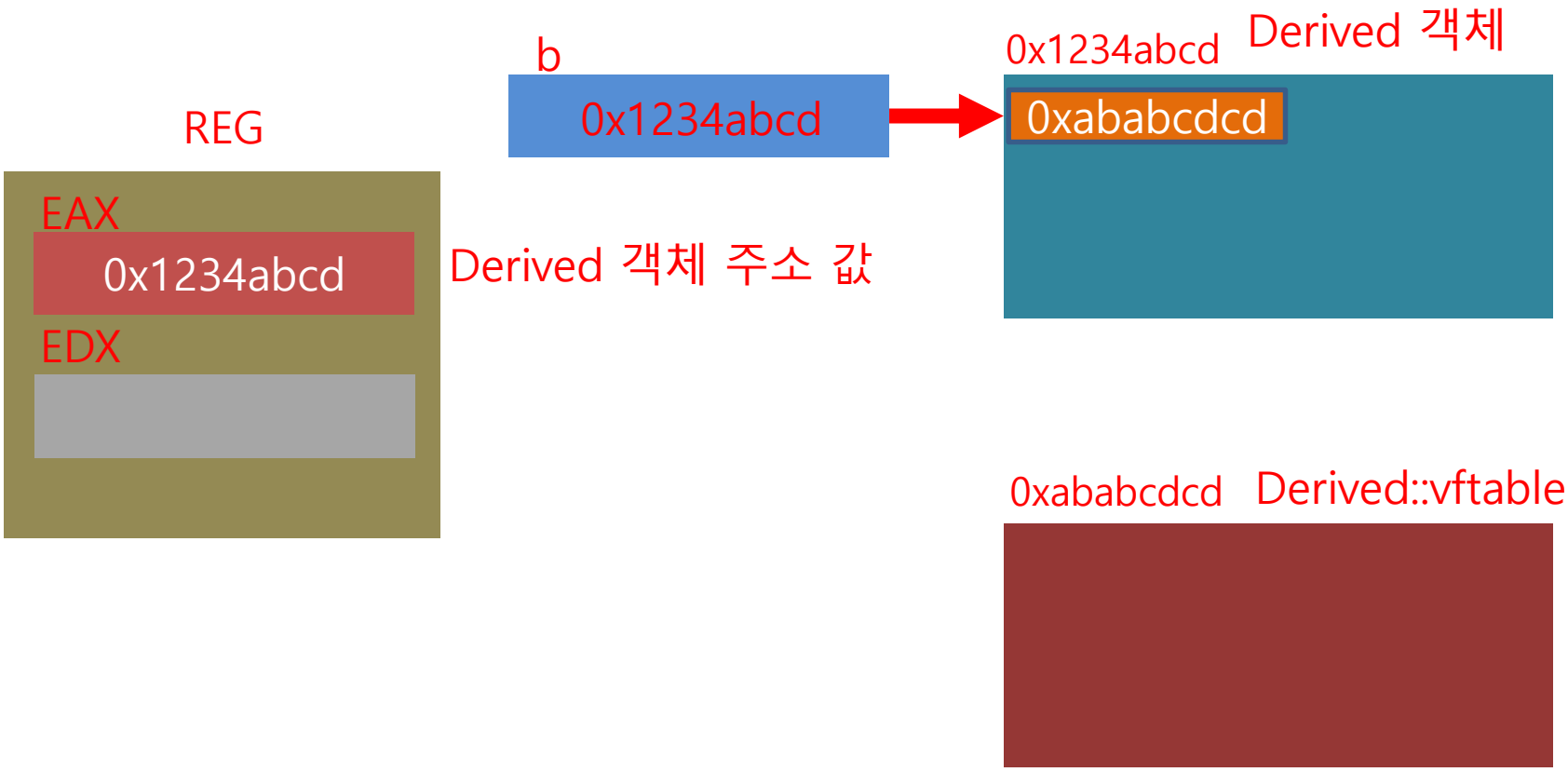
```
//b->VFunc1();
__asm
{
    mov eax, dword ptr[b]
    mov edx, dword ptr[eax]
    mov eax, dword ptr[edx]
    call eax
}
```

Virtual Table

Virtual table

```
mov eax, dword ptr[b]
mov edx, dword ptr[eax]
mov eax, dword ptr[edx]
call eax
```

&b를 의미한다

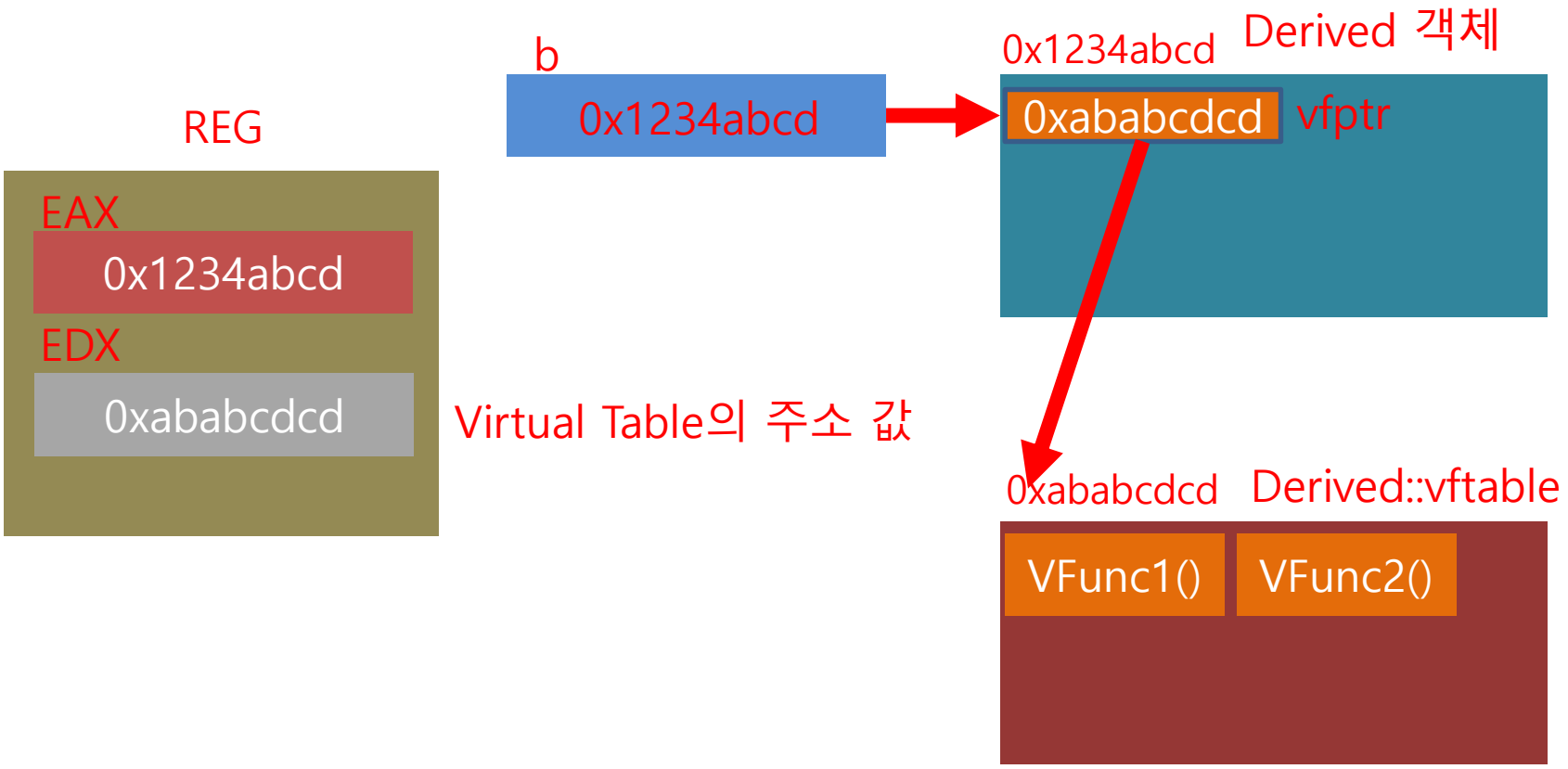


Virtual Table

Virtual table

```
mov eax, dword ptr[b]
mov edx, dword ptr[eax]
mov eax, dword ptr[edx]
call eax
```

Eax를 주소 값으로 4 바이트 가져옴

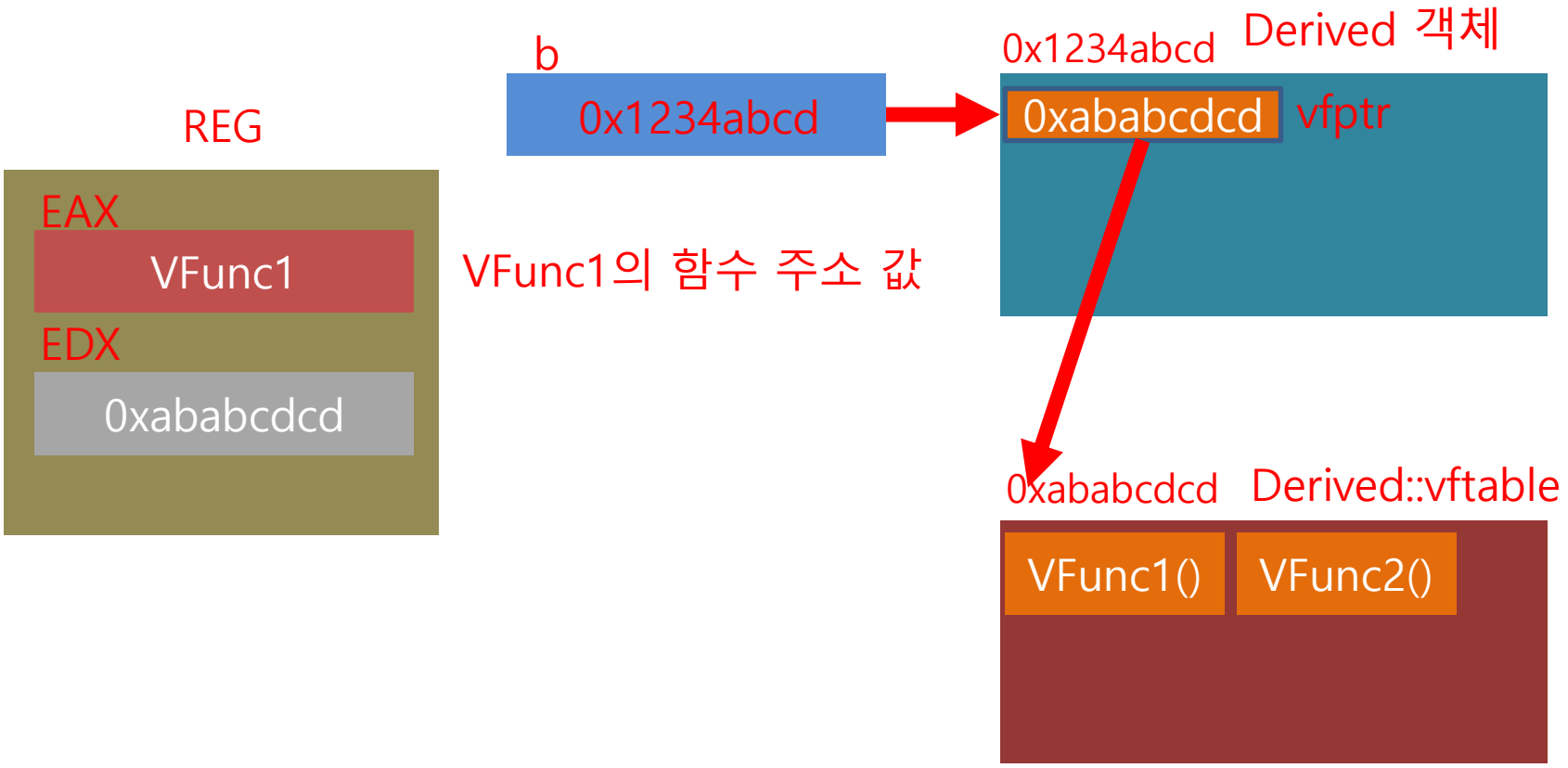


Virtual Table

Virtual table

```
mov eax, dword ptr[b]
mov edx, dword ptr[eax]
mov eax, dword ptr[edx]
call eax
```

Edx를 주소 값으로 4 바이트 가져옴



Virtual Table

Virtual table

함수 호출

```
//b->VFunc1();
__asm
{
    mov eax, dword ptr[b]
    mov edx, dword ptr[eax]
    mov eax, dword ptr[edx]
    call eax
}
```

Derived::VFunc1()