시스템 프로그래밍을 위한 C언어

union을 이용하여 메모리에 할당된 multiple bytes 데이터 패턴을 다양한 타입으로 바라보기

현대자동차 입문교육
박대진 교수

Union을 이용하여 특정 메모리에 할당된 데이터 패턴을 다양한 타입으로 표현하기(해석)

```c
struct bits_8 {
    unsigned char b0 : 1;
    unsigned char b1 : 1;
    unsigned char b2 : 1;
    unsigned char b3 : 1;
    unsigned char b4 : 1;
    unsigned char b5 : 1;
    unsigned char b6 : 1;
    unsigned char b7 : 1;
};
```

```c
union flag_16bits {
    unsigned char bytes[2];
    struct bits_8 b[2];
    unsigned short bits16;
};
```

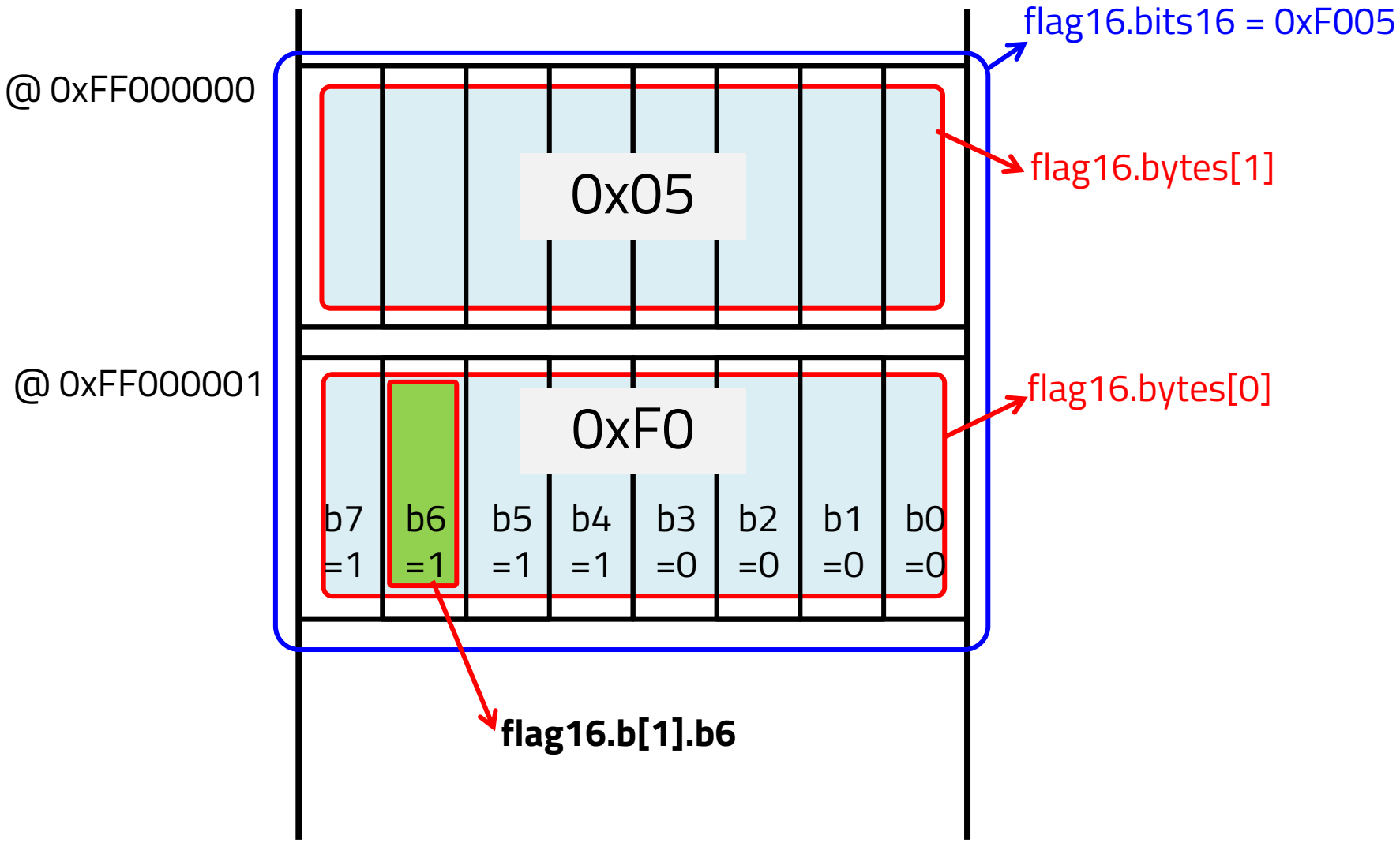
```c
union flag_16bits flag16;
printf("flag16 is with %d bytes\n", sizeof(flag16));

flag16.bits16 = 0xF005;
printf("flag16 is 0x%02X%02X\n", flag16.bytes[1], flag16.bytes[0]);

flag16.b[1].b7 = 0;
printf("flag16 is 0x%02X%02X\n", flag16.bytes[1], flag16.bytes[0]);
flag16.b[0].b3 = 1;
printf("flag16 is 0x%02X%02X\n", flag16.bytes[1], flag16.bytes[0]);
```

```
flag16 is with 2 bytes
flag16 is 0xF005
flag16 is 0x7005
flag16 is 0x700D
```

Memory Layout

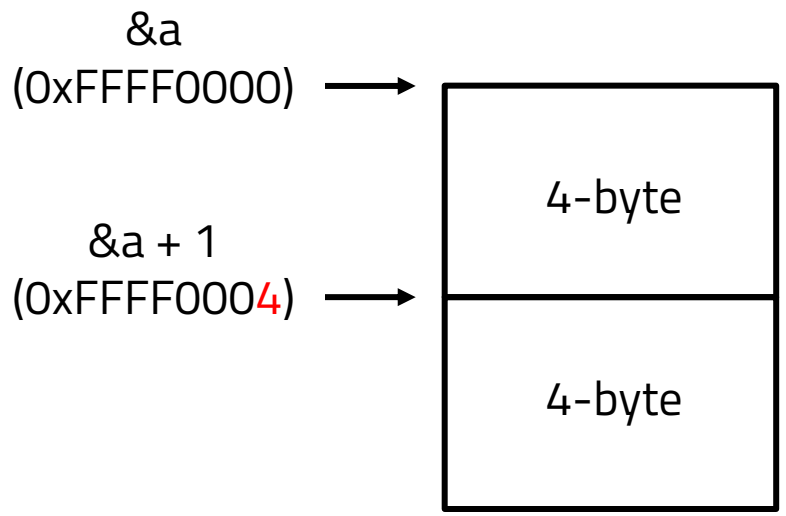


변수를 메모리에 할당된 공간(영역)으로 생각하라

- 변수 이름은 메모리에 할당된 공간
- 변수 이름 앞에 &를 붙이면 할당된 공간 전체를 가르키는 주소임
- 따라서 변수 이름앞에 &를 붙이고 +1을 하면, 가르키는 대상 변수가 차지하는 바이트수 만큼 주소가 증가해야 함.

int a

`int *a`
→ 4-byte 영역을 가리킴



항상 변수는 메모리 할당된 영역으로 보라.

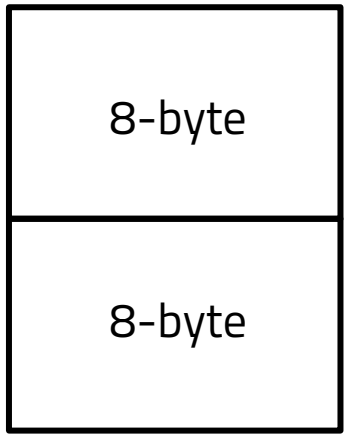
```
struct B {
  unsigned int m1;
  unsigned int m2;
};
```

struct B b

`struct B *b`
→ struct B 크기만큼의 영역을 가리킴

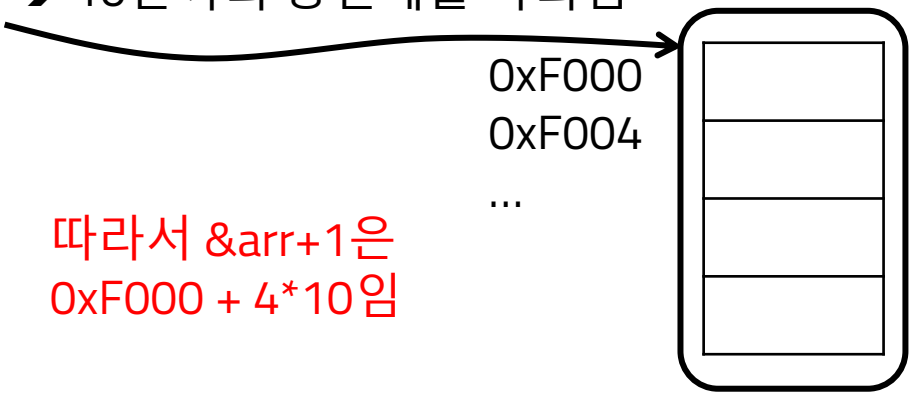
&b
(0xFFFF0000) →

&b + 1
(0xFFFF0008) →



만약 struct B 가 8-byte 크기인 경우...

int arr[10]; ➔ 10칸 방전체를 나타냄
&arr ➔ 10칸짜리 방전체를 가리킴



따라서 &arr+1은
0xF000 + 4*10임

특수하게 배열의 이름은 배열의 첫번째 방을 가리킴 (전체를 가르리키는게 아님)