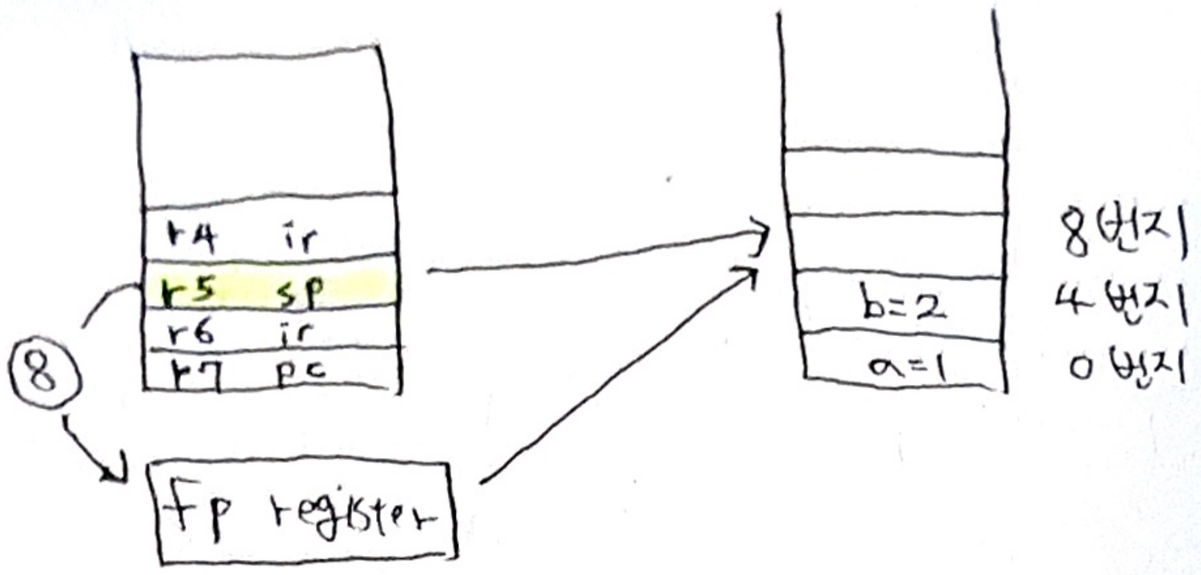
10장 | 절차적 함수 호출 지원 CPU 모델

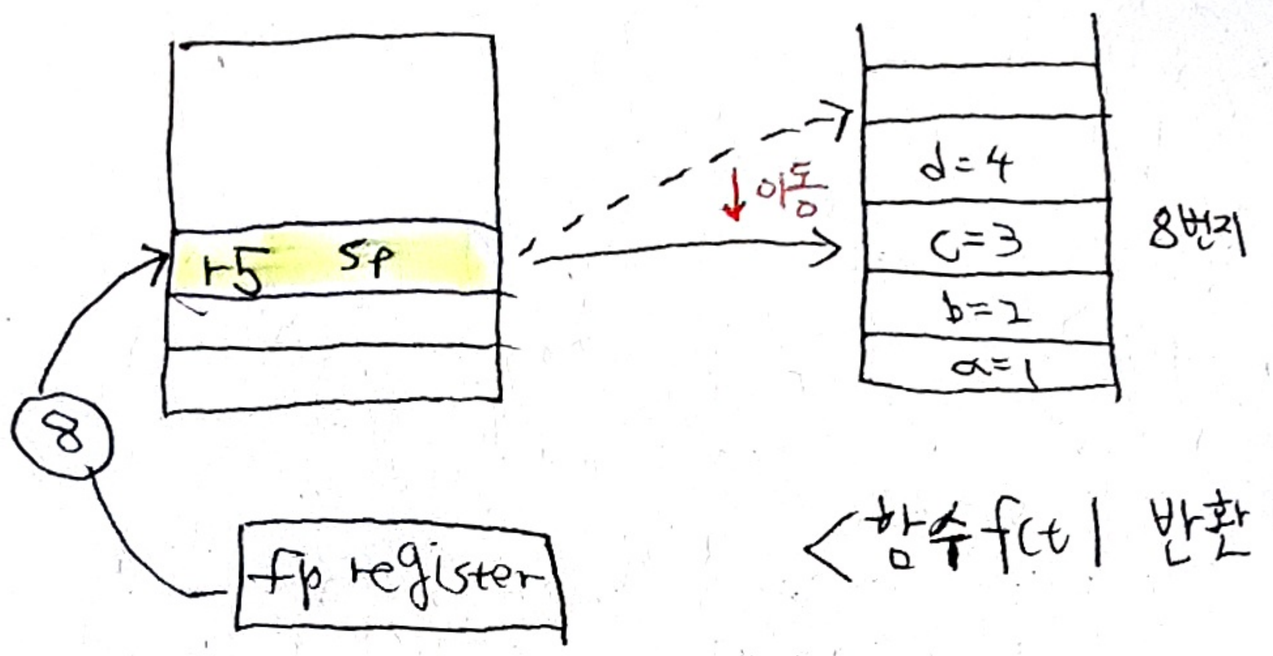
-1/15 이충현

```
int main(void){
    int a=1;
    int b=2;
  fct1();
  }
```



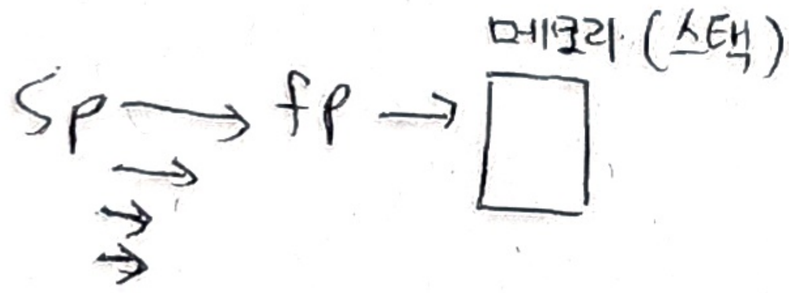
<함수 fct1 호출시>

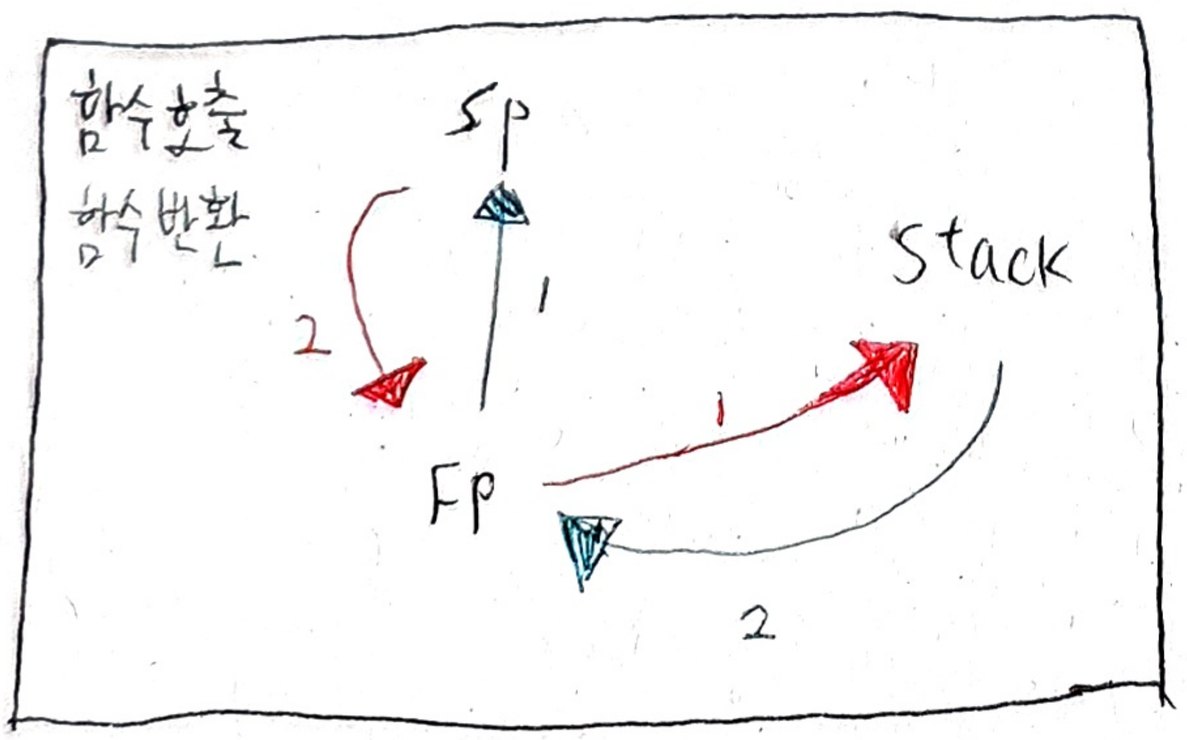
```
Void fct1(void){
    int c=3;
    int d=4;
}
```



<함수 fct1 반환 이후>

1. sp register의 백업 = fp register
2. 지운다 = 덮는다.
3. fp 레지스터 문제점 = 함수가 꼬리에 꼬리를 물면 백업되는 값이 많아짐. (ex) 재귀
   그러나 레지스터는 1개라서 값(기존)을 덮으므로서 소실우려
4. 해결책 = fp 레지스터에도 별도의 메모리 공간이 필요하다.

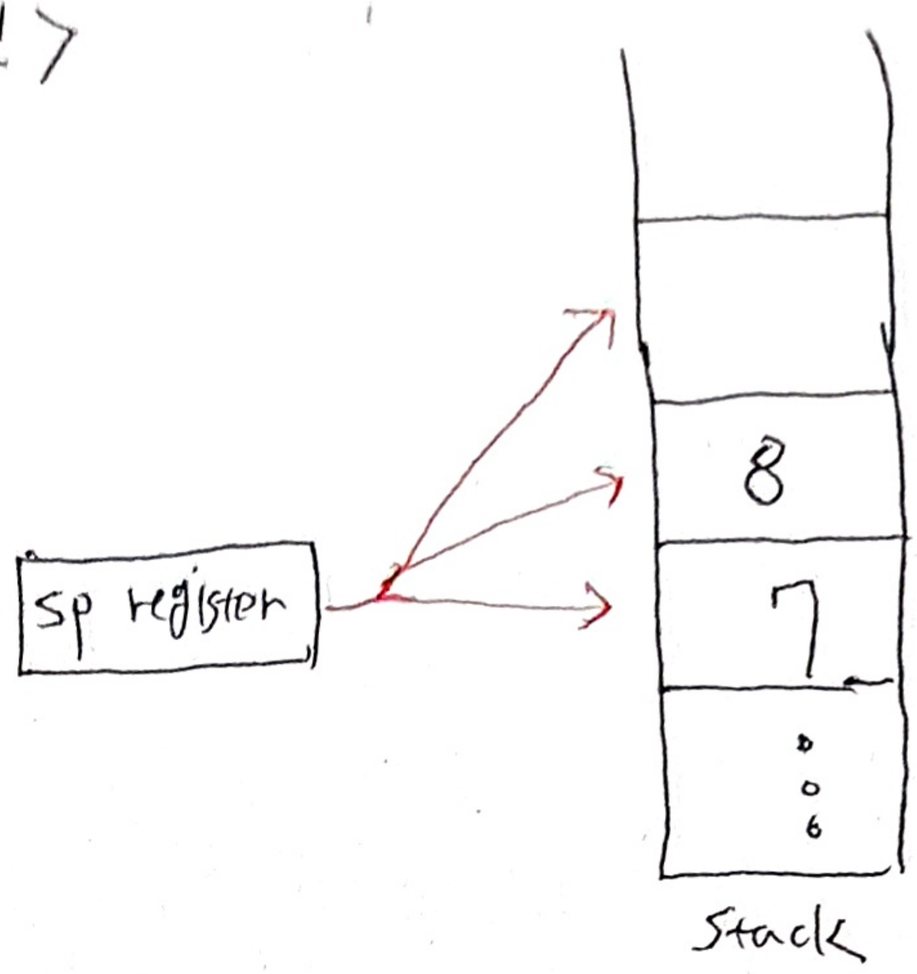




< 인자 전달과 Push & Pop 명령어 디자인 >

① 함수호출 인자의 전달방식

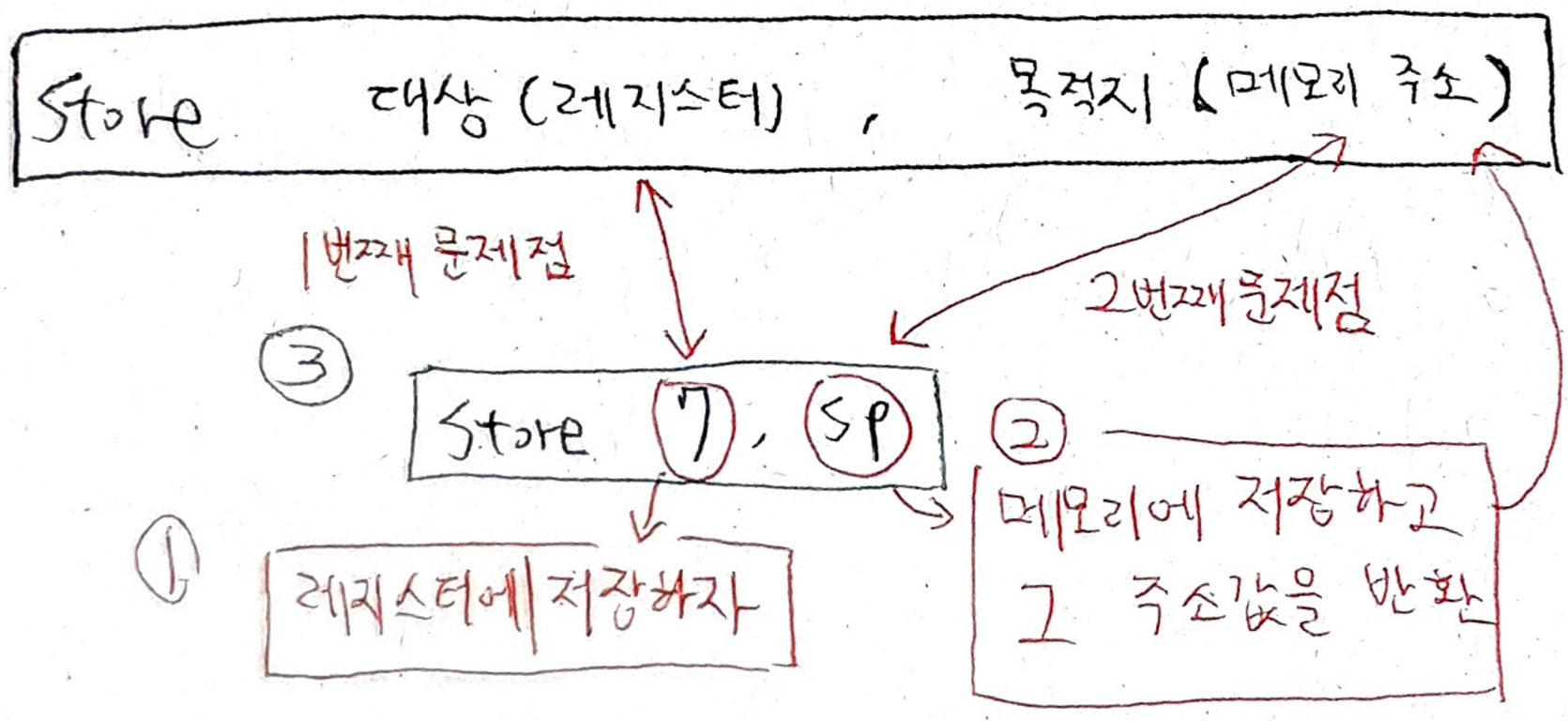
```
function(7,8)

void function( int a, int b){
    int c=9;
    }
```



sp가 가리키는 현재 위치에 전달되는 인자값 저장
→ sp 증가시켜 다음 메모리 주소를 가리키게 한다.

~~Store(ㅁ, sp)~~

② Store 대상 (레지스터) , 목적지 (메모리 주소)



① APP r1, 7, 0
② store sp, 0x40
③ store r1, [0x40] ← Indirect 모드
④ ADD sp, sp, 4 (byte)

④-1 ⇒ , -4 ⇒ pop 명령어

⇒ push 명령어