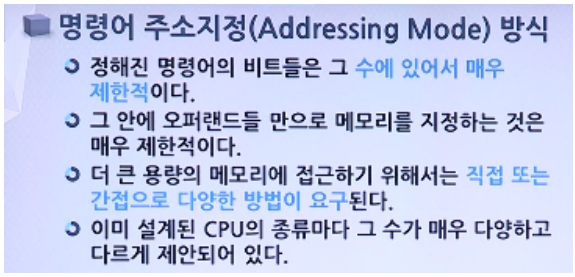
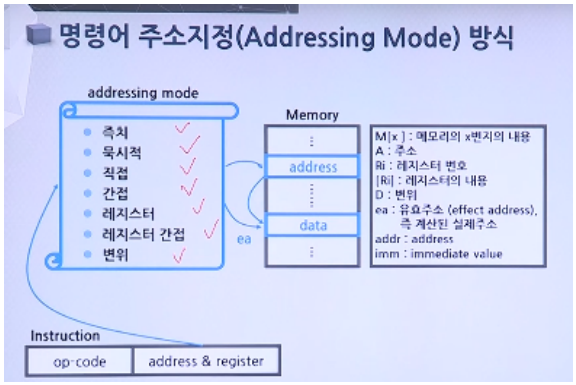
12강_명령어 주소지정 방식

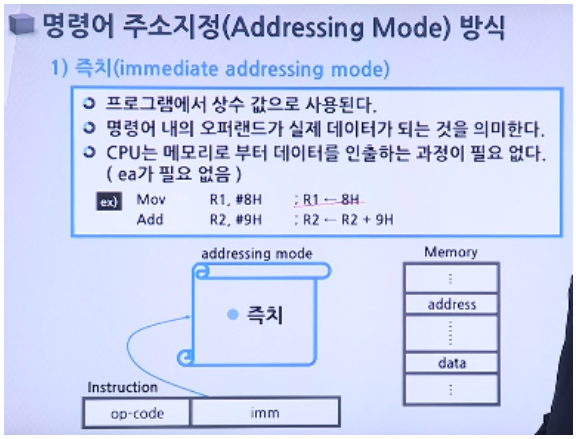
Created	@Aug 11, 2020 5:38 PM
Tags	



비트 수에 따라서 큰 메모리를 지정하게 하는 방법들을 정리해놓은 것이 Addressing Mode
아키텍쳐마다 방식이 다름



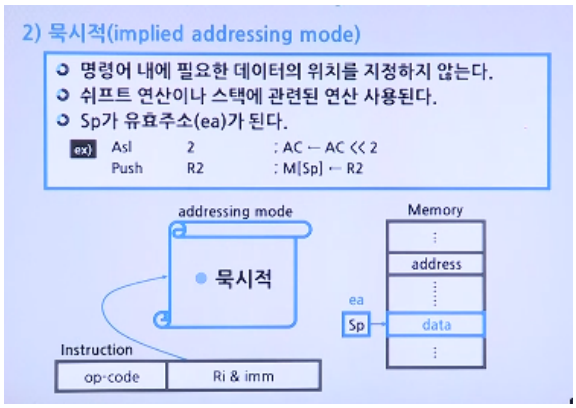
여기에 나와 있는 addressing mode들은 오퍼랜드에서 값을 가지고, 이 중에 어떤 것을 선택해서 이 ea(유효주소)를 찾는 것이다.



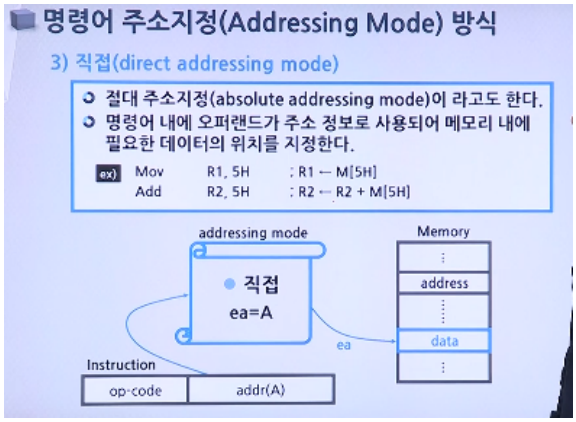
메모리에서 뭔가를 가져오는 것이 아니라, 실질적으로 이 명령어 안에, 오퍼랜드 안에 직접 값을 넣음으로써 그 값을 옮길 수 있다
실질적으로 메모리로부터 데이터를 인출하는 과정이 필요없기 때문에 = ea가 필요없다고 말할 수 있다

메모리 액세스가 안됨 → 시간이 상당히 빠른 형태

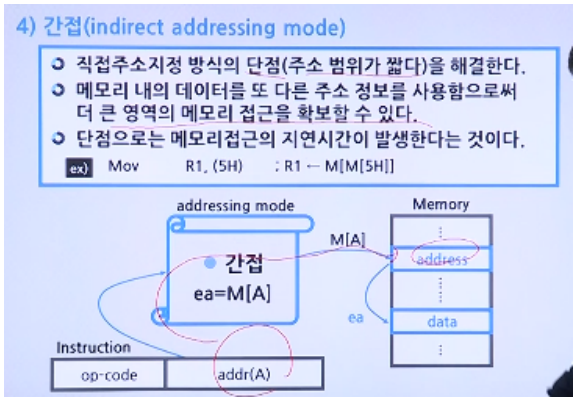
오퍼랜드가 실제 데이터가 되는 것을 의미



말 안해도 알아서 처리한다(내포되어 있다).
명령어 내에 필요한 데이터의 위치를 지정하지 않는다. 암시적으로 지정한 것처럼 사용한다
쉬프트 연산, 스택연산, 푸쉬-스택 자동증가, 감소 등등- 스택포인터가 유효주소가 되는 그런...

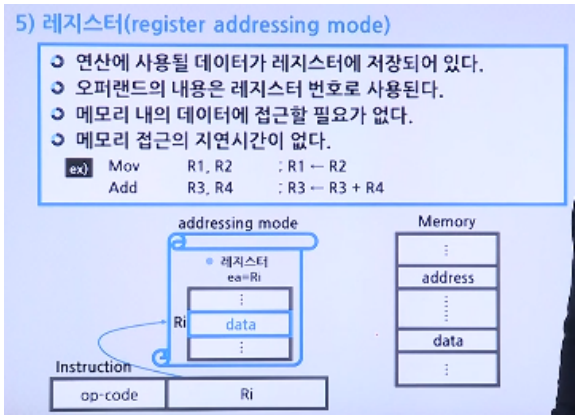


direct addressing mode를 absolute mode 라고도 한다
명령어 내에 오퍼랜드가 주소 정보로 사용되어 메모리 내에 필요한 데이터의 위치를 지정한다
오퍼랜드에 있는 어드레스 주소가 유효 주소가 되는 것.
ex) 여기가 100번지다...그럼 데이터도 100번지에 가면 있다.

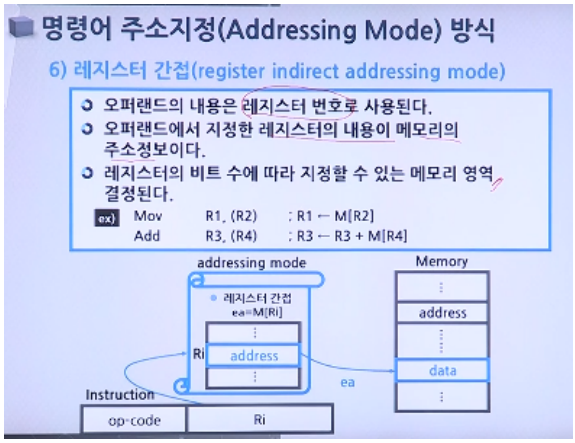


직접 주소지정의 단점(큰 메모리를 지정하기 힘들다)을 보완함.
메모리 내의 데이터를 또 다른 주소 정보를 사용함으로써 더 큰 영역의 메모리 접근을 확보할 수 있다
지정된 주소로 가면 또 주소가 있는 형식. 그 주소(이게 유효주소가 된다)를 한 번 더 가야 데이터가 있다.

Register Addressing Mode
레지스터를 사용하여 주소지정을 한다. 연산에 사용될 **데이터가 레지스터에 저장**되



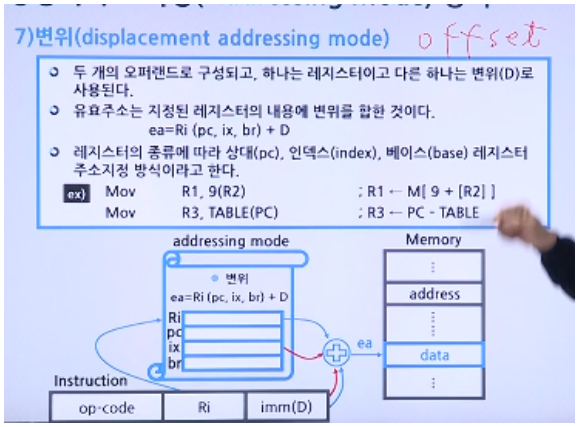
어 있고, 오퍼랜드의 내용은 레지스터 번호로 사용된다
메모리 내에 데이터에 접근할 필요가 없다, 메모리 접근의 지연시간이 없다
컴퓨터에서는 메모리에 가지 않으면 빨라서 좋다



레지스터 간접
간접인데 레지스터를 이용. 오퍼랜드 내용은 레지스터 번호로 되어 있음, 오퍼랜드에서 지정한 내용이 메모리의 주소이다.

레지스터 비트 수에 따라 지정할 수 있는 메모리 영역 결정된다
레지스터가 가지고 있는 것은 어드레스 정보.
오퍼랜드에 레지스터 번호를 갔더니 데이터가 아니라, 주소여서 레지스터 간접이 됨

레지스터 간접은 주소, 이 레지스터 크기만큼 어드레스 양을 더 확보할 수 있으니 큰 영역의 메모리를 지정할 수 있는 방식이 됨



변위(displacement addressing mode)
오프셋*offset*을 사용한다(변화량).
두 개의 오퍼랜드로 구성(하나는 레지스터, 다른 하나는 변위)
유효주소는 지정된 레지스터의 내용에 변위를 합한것이다
이곳의 레지스터는 프로그램 카운터, 인덱스, 베이스 레지스터 등등 주로 사용

TABLE(PC) 주로 변위를 얘기함
PC와 TABLE과의 차이점이 오프셋이 됨