이번 발표에서는 수업시간에 다루었던 RISC구조와 CISC구조의 차이와 특징 그리고 실제 RISC의 구조의 CPU에 대한 간단한 소개를 하도록 하겠습니다.

RISC구조와 CISC구조를 한 마디로 정의하면 기성세대와 신세대의 대결로 볼 수 가 있습니다.

저는 RISC와 CISC 두 가지중에 현재 핫한 RISC에 조금 더 초점을 두고 진행하도록 하겠습니다. RISC방식은 기존 CISC방식의 구조의 복잡성과 명령어의 길이를 간소화하고자 개발되었으며 현재 스마트폰, 태블릿 pc등 여러전자기기에서 사용되고있습니다.

그럼 요즘 데세인 RISC구조에 대해 알아보도록 하겠습니다. 그에 앞서서 먼저 간단한 대표적인 RISC구조 프로세서 2가지에 대해 설명드릴까 합니다. 바로 현재 가장 유명한 두 회사 애플과 삼성의 cpu AP 프로세서입니다.

삼성은 엑시노스라는 AP 프로세서를 직접 설계, 생산을 담당하고 있습니다. 현재 사용되고 있는 엑시노스는 엑시노스9810모델까지 생산되었으며 10nm공정으로 설계.생산된 제품입니다.

여기서 말하는 nm공정은 회로폭을 의미하며 회로폭이 작을수록 빠르고 효율성도 뛰어납니다. 삼성은 앞으로 신의영역인 2nm공정까지 도전중에 있습니다.

앞으로의 삼성의 반도체개발이 기대됩니다.

다음은 애플의 A시리즈 입니다.

애플의 A시리즈 CPU는 요즘대세인 옥타코어가 아닌 싱글코어 2개에 효율코어 4개를 이용하는 독특한 구조를 지니고 있는데 이 독특한 구조는 다른 CPU제조업체 성능의 2배에 가까운 성능을 보입니다. 높은 가격정책으로 비난을 받고 있지만 대단한 것만은 사실인 것 같습니다.

지금부터는 RISC 동작방식과 특징을 얘기하도록 하겠습니다.

동작방식

* 3-주소 명령어 형식이란?

동작코드 목적지 오퍼랜드 소스 오퍼랜드1 소스 오퍼랜드2 로 이루어져있다.

전체적으로 명령어 길이가 길지만 고급 언어로 작성된 프로그램을 기계어로 변환할 때 필요한 명령어들의 수는 적어진다.

그리고 RISC는 고정 길이 명령어들을 사용한다. 조합 순차논리회로 사용으로 수정시 복잡하다는 단점이 있지만 간결하다는 장점이 단점을 상쇄시킵니다.

특징

* 사이클 당 단일 명령어 : 기계 사이클(machine cycle) 당 하나의 기계어. 마이크로프로그램 제어 장치 액세스 필요 무.
* 많은 수의 범용 레지스터 : 레지스터 간의 연산 등으로 기억장치 참조 횟수를 줄임. 중첩 레지스터 윈도우 등의 레지스터 참조 최적화 기법 사용. 기억장치 액세스 동작은 간단한 LOAD와 STORE 명령어만으로 이루어짐.

간단한 주소지정 방식 : 명령어 세트와 제어 장치가 단순화 됨.  
  
간단한 명령어 형식 : 하나 또는 적은 수의 형식이 사용. 특히 OP 코드의 길이가 고정됨.

여기까지 RISC에 대한 설명을 마치고 기성세대 방식인 CISC에 대해 설명드리겠습니다.

* CISC(Complex instruction set computer): 마이크로프로그래밍을 통해 사용자가 작성하는 고급언어에 각각 하나씩 기계어를 대응시킨 회로로 구성된, 중앙처리장치의 한 종류이다.
* 출처: 두산백과
* 1971년 인텔은 4비트 마이크로프로세서 4004을 내놓은 것이 시초인데 그 이후로 인텔은 지금까지 CISC방식을 사용하고 있다.
* CISC계열의 마이크로프로세서들은 대부분 pc기반이다.
* EX)인텔 i시리즈, 펜티엄, AMD 라이젠 시리즈이다.

CISC의 개발동기

보다 강력하고 복잡한 고급 프로그래밍 언어, 즉 고급 언어(High-Level Language, HLL)에 대한 보다 나은 지원은 CPU 설계자의 공통된 목표.이를 위해 컴파일러의 단순화와 성능 향상을 목표로 두었고 이로 인해 CPU의 명령어 세트는 명령어 수가 더 많아지고 복잡한 경향이 생김.

동작방식

* CISC의 명령어들은 2-주소 명령어 형식을 사용한다.
* 2-주소 명령어 형식이란?

동작코드 목적지 오퍼랜드/소스 오퍼랜드 소스 오퍼랜드 로 이루어져있다.

3-주소 명령어 형식에 비하여 명령어 자체의 길이는 짧아지지만,

고급언어 프로그램을 기계명령어로 표현할때 명령어의 수가 증가한다.

현재 CISC 계열의 마이크로프로세서들이 주로 사용한다.

그리고 CISC는 가변 길이 명령어들을 사용한다.

특징

* 가변 길이의 많은 명령어 : HLL이 요구하는 다양한 명령어에 대한 요구를 직접적으로 처리 가능.
* 다양한 주소 지정 방식을 동시에 수용 : 즉치, 직접, 간접, 레지스터, 변위 주소지정 방식 등
* 적은 개수의 특화된 레지스터(데이터 레지스터, 주소 레지스터 등) : 오퍼랜드 지정자가 참조할 레지스터의 형태가
* OP 코드 내에 묵시적으로 지정되므로
* 비트 절약의 효과.
* 프로그래밍의 융통성이 제한됨.
* 파이프라이닝, 수퍼스칼라 방식 적용이 상대적으로 어려움 : 다양한 명렁어, 명령어 형식, 주소지정 방식으로 인한 복잡도 증가로 인함
* control unit 구현 방식 : micro programm 방식.

(많은 수의 명령어, 즉 많은 수의 op 코드로 인한 복잡도 증가로 인해 hardwired 방식으로는 구현이 어려움)  
cf) HLL=HIGH LEVEL LANGUAGE

RISC구조가 나날이 갈수록 그 성능이 기하급수적으로 좋아지고 있음에 따라 RISC구조의 프로세서들이 CISC기반의 프로세서들을 잠식하는 날이 머지않아 올 것 같다. 최근에 뉴스들이 그 단면을 잘 보여주고 있다. 인텔이 스카이레이크에 14nm공정으로 생산한 cpu이후로 계속 nm공정을 줄이지 못하고 성능또한 뚜렷한 향상이 줄어들고 있다. 그 에반해 삼성 이나 퀄컴, 애플의 cpu등은 날이면 날마다 그 기술과 효율성이 좋아지고 있다. 조만간 인텔의 시대는 한 풀 꺽일것같다.