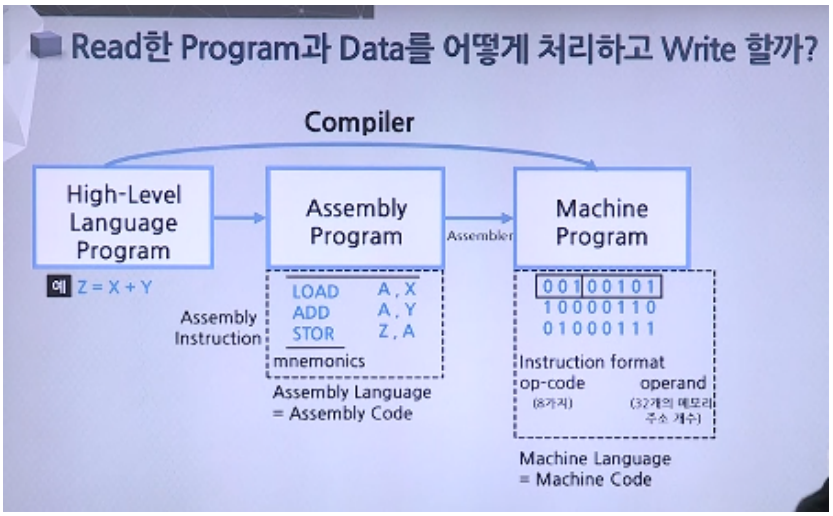
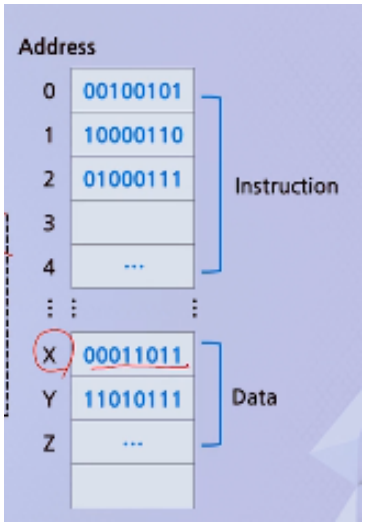
3강_Program과 Data의 처리, 반도체 개요

Log into your K-MOOC Account | K-MOOC

http://www.kmooc.kr/courses/course-v1:SMUCk+CK.SMUC03k+2017_T6/courseware/e0c4ed29d8f3418185b69b57a6b78fe9/fc2e00b2353146fb95e9310426c8deb6/?child=first

Read한 Program과 Data를 어떻게 처리하고 Write할까?





Assembly Instruction

LOAD A,X : Assembly Instruction에서는 항상 목적지가 앞에 나옵니다.

1. ‘X에 있는 값을 A로 LOAD 해라.’ LOAD : 메모리에 있는 어떤 데이터를 CPU에 있는 레지스터로 갖고 오라

레지스터 A = Accumulator : 누산기. ‘누산기로 X라는 메모리에 저장된 값을 A로 가지고 와라, LOAD 해라.’ 라는 의미

2. ADD 명령어가 있네요. ADD 명령어. 목적지가 앞에 있다고 그랬죠. Accumulator A에 어떤 값이 저장 되있어요. 그 저장된 값은 X에서 가져온 값이었어요.

X에서 가져온 값을 A에 저장됨(누산기), Y라는 곳에 저장되어 있는 값을 A로 가져오는데 그냥 가져 오는게 아니라 기존에 있던 A값에 더하라는 의미

예를 들어서 X=10, Y=5면 최종적으로 Accumulator에 저장된 값은 15

3. STOR

A에 있는 것을 Z로 STOR(A는 Accumulator, Z는 메모리~ 메모리에 저장된 값을, 메모리의 위치를 의미하죠. 그 위치에다가 이제 저장)

4. 결론 : X와 Y를 더해서 Z에 저장하는 형태

High-Level Language로 쉽게 짤 수 있는 것을 Assembly Language를 이용하면 이렇게 복잡하게 짜야한다

▼ 다음설명

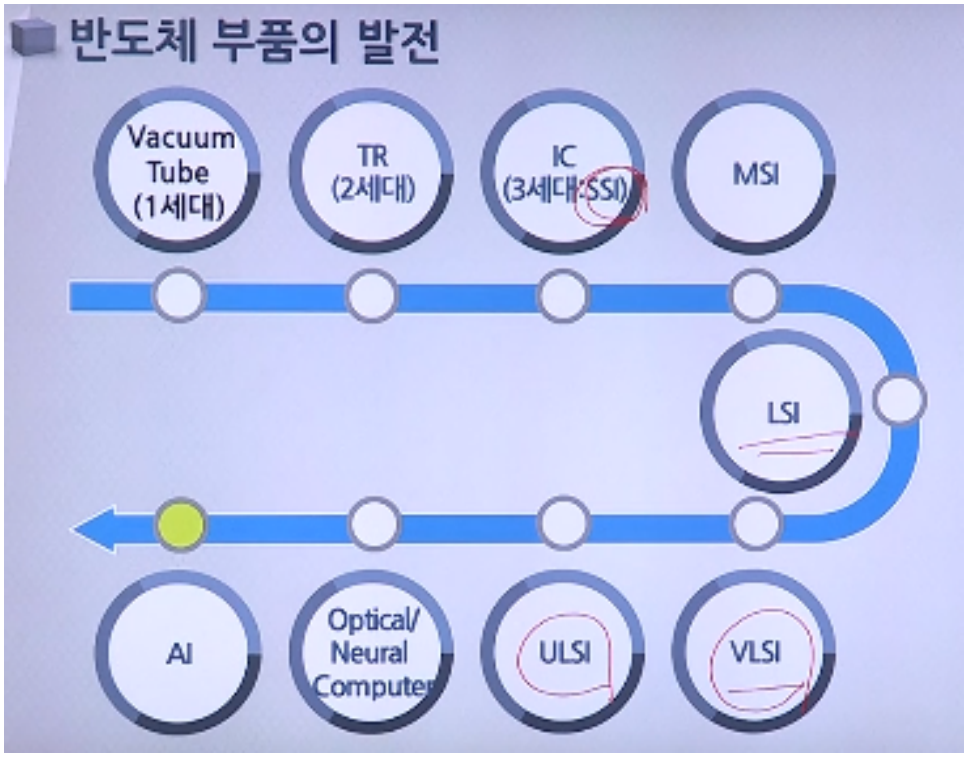
1. 자, 이런 것을 가져다가 우리가 1010 데이터로 바꿔보겠습니다. 어떻게 바뀠나 봅시다. 먼저 이렇게 한 라인씩 대응이 되는데요.
2. 먼저 LOAD라는 명령은 여기있는 001입니다. 자, 지금 3비트니까요. 총 몇 개의 비트 조합을 가질 수가 있을까요? 조합을 몇 가지를 가질 수 있냐면 8가지를 가질 수 있습니다. 그러니까 명령어의 개수를 8개를 만들어 낼 수 있습니다.
3. 자, 그 다음에 이 뒤에 보시면 X에 해당하는 게 00101이라고 있네요. 저건 2진수니까요. 10진수로 바꾸면 5입니다. 5가 되고요, 이건 6. 이건 7이죠. 이렇게 자르면. 그래서 5, 6, 7이 됩니다. LOAD 명령어는 001에 해당하고요.
4. 그 다음에 이 5는 뭐냐면 X를 의미합니다. X라는 값이 어디에 저장돼있냐면 5번지에 저장돼있는 의미입니다. 그럼 메모리에 있는 5번지를 찾아 가야 되겠죠? X에 해당하는 게, 이렇게 값이 저장되어 있겠죠.
5. 저건 2진수를 10진수로 바꾸면 금방 계산은 안 되지만, 어쨌든 이런 2진수 값을 이용할 수가 있게 되는 거고요. 자, 그 다음, 이걸 한 번 더 볼께요. 더 보시면 그럼 도대체 A는 없냐? A도 여기에 대응되는 무언가가 있어야 되는데 Binary Code에는 A가 없거든요.
6. 그건 또 이유가 있습니다. 그 이유는 LOAD가 됐든, ADD가 됐든, STOR가 됐든 모든 작업이 이루어지는 장소는 레지스터는, 그냥 Accumulator입니다. 도마라고 그랬

죠? 굳이 지정할 필요가 없어요. 무조건 Accumulator니까.

7. 당연히 이건 무슨 이야기냐? 이걸 딱보면 아 001이란 명령어가 LOAD니까 데이터를 메모리에서 가지고 오는데, 어디에 있는 거냐면 5번지에 있는 데이터를 가지고 오라는 이야기고, 5번지에 가면 어떤 데이터가 있는 것을 가지고 와서 어디에다 저장하냐는 이야기냐?
8. 여기 써 있지 않아도 당연히 Accumulator에 저장하라는 애기입니다. 기본이 Accumulator이기 때문에.
9. 두 번째 볼께요. 두 번째도 10000110인데요. 100이 의미하는 바가 뭐냐면 바로 ADD라는 명령어입니다. 그렇게 약속을 해 놓은 겁니다. 그 다음에 00110이니까 6번지에 있는 데이터를 찾아 가게 됩니다. Y값이 거기에 저장되어있습니다.
10. 찾아가보면 여기에 11010111이 있고요. 그 값을 가져오는 거죠. 값을 가져와서 그럼 어떻게 해야 되느냐? ADD를 해야되니까, 지정된 게 없죠. 어디에다 더하라는 이야기가 없잖아요 사실. 이 Binary Data에는 근데 이게 지정이 돼있습니다. 이미.
11. 묵시적으로.. 바로 Accumulator죠. Accumulator에 있는 기존 값을 더하라는 이야기입니다. 세 번째 꺼 볼까요? 자, 010은 STOR하라는 이야기입니다. STOR하라는 이야기인데 00111 이게 7번지죠. 7번지에다가 계산된 값을 저장하라는 이야기인데 도대체 무슨 값을 저장하라는 이야기냐,
12. 여기서는 이야기는 안했지만 묵시적으로 Accumulator의 값. Accumulator에서 이 두 개를 더한 값이 저장이 되어있을 텐데, 그것을 7번지에다가 저장을 하라는 이야기입니다. 그러는 식으로 코드가 구성이 되죠.
13. 그러니까 여기서 이해해야 될 것은 이것이 High-Level Language, 그다음 Assembly Language로 짤 수는 있지만 이렇게 짜진 않는다. 근데 이 Assembly Language를 왜 알아야 하냐면 여러분들 아까 말씀드렸던 Micro Instruction있잖아요. Machine Instruction과 Micro Instruction.
14. 그러니까 프로세서를 설계하는 사람들, 그 사람들은 Micro Instruction 이런 코드로 전부 다 설계를 합니다. 그러니까 이것에 대한 내용도 알고 있어야 되요. 대체로 저거 가지곤 실제 프로그램을 짜는 일은 거의 없습니다.

Machine Program

컴퓨터에 저장된 값은 이런 값들이 실제로 메모리에 저장되고, CPU에 있는 레지스터로 이동되고 이런 작업 이루어짐. 이런 형태로 이제 Read한 프로그램과 데이터 처리를 하게 됨



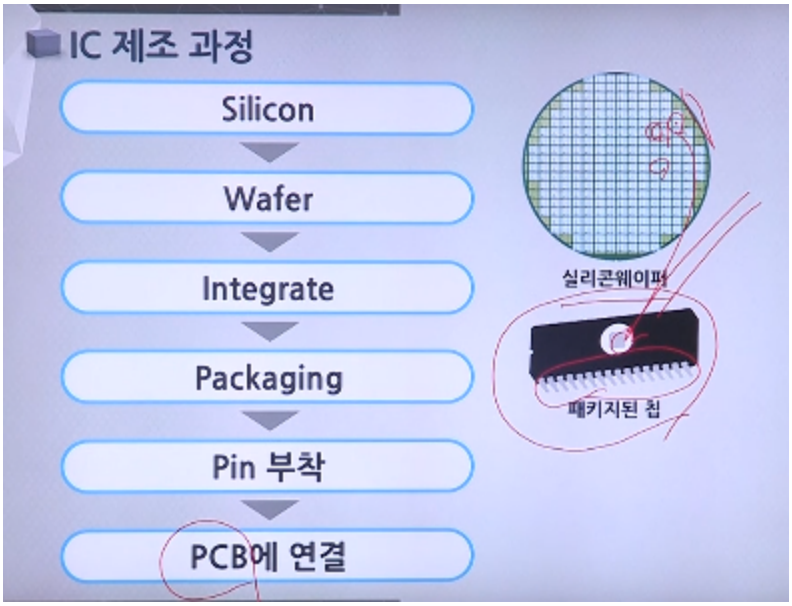
▼ 반도체 부품의 발전

1. 자, 이제 처리된 방식에 이야기는 했고요. 지금부터는 이제 마음 편하게 컴퓨터를 구성하는 여러가지 부품들이 반도체로 이루어져있기 때문에 반도체 부품이 어떠한 식으로 발전해왔고 컴퓨터는 어떠한 식으로 발전해왔는지 그냥 머리 식히는 시간을 갖도록 하겠습니다.
2. 반도체 부품의 최초는 'Vacuum Tube'라고 해서 진공관이죠. 진공관은 다이오드의 역할을 하는 건데요. 다이오드는 뭘 생각 할 수 있냐면 스위치 생각하시면 됩니다. 스위치. 스위치를 끄면 전류가 안통하고 스위치를 키면 전류가 통하죠.
3. 그다음에 'Transistor'라는 것이 있습니다. Transistor는 Vacuum Tube가 발전된 형태, 지금 현재 컴퓨터로 구성하는 모든 반도체의 기본 단위는 Transistor죠. Transistor를 만든 사람은 노벨상을 당연히 탔겠죠. 굉장히 중요한 발명이고요. 상당히 하여튼 Transistor를 발명한 사람들이 굉장히 유명합니다. 다이오드도 그렇고.
4. 그 다음에 이제 Transistor를 기판 안에 집적을 시켜서 수 백, 수 천, 수 백 만개의 Transistor를 하나의 칩 안에 이렇게 집적을 시키게 됐는데, 그 집적한 첫 번째의 그

런 것들을 우리가 IC라고 부릅니다.

5. 근데 IC의 집적도의 정도에 따라서 SSI는 Small Scale Integration. 그러니까 집적이 좀 덜 돼있다는 말이죠. 그 다음에 MSI는 Medium Scale Integration, 약간 집적이 더 많이 돼있는 거죠. 단위 면적에 Transistor가 더 많이 들어가는거죠. 그 다음에 LSI, Large Scale인데요.
6. 지금 삼성에서 반도체를 만드는 사업부의 이름이 LSI사업부라고 그러거든요. 지금 현재는 어떤 시대냐, LSI시대는 아니고요. 제가 볼 때는 VLSI에 가깝습니다. Very Large Scale, 여기에 가깝습니다. 지금 하여튼 삼성이 여전히 LSI사업부라고 이름은 붙이고 있죠. 그 다음 ULSI. 이건 아직 세대는 아닌 것 같고요.
7. 그 다음에 앞으로 나올 것들 Neural Computer라는 게 있죠, 신경망을 이용한 컴퓨터. 또는 Optical Computer가 나올 가능성이 있고요. 아주 먼 미래에는 알파고와 같은 AI, 인공지능의 형태의 어떤 반도체가 나올 수가 있을 것 같습니다.

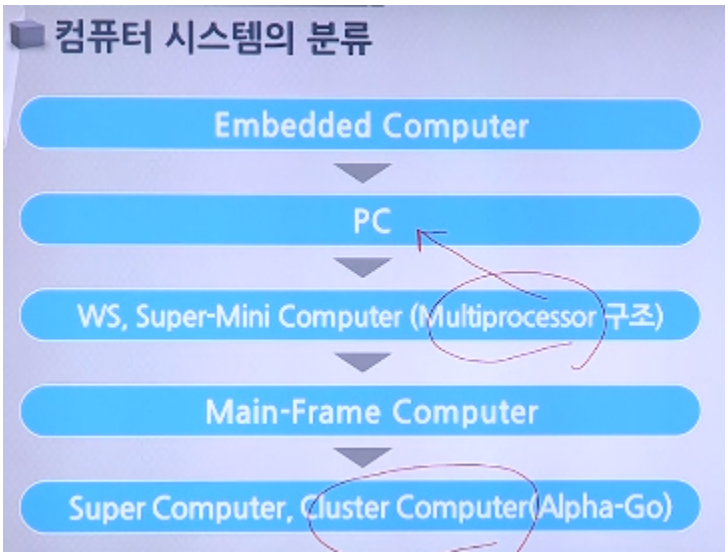
IC 제조 과정



실리콘웨이퍼, 정사각형 하나하나가 칩임, 짜투리들 버림

핀 부착 PCB : 녹색 판떼기

컴퓨터 시스템의 분류



Embedded Computer : 내장 컴퓨터. 가장 작음

PC : Personal Computer

WS : Work Station. 좀 더 크고 성능이 좋은 것. 쓰임새 줄어들어서 실제로 쓰진 않음

Main-Frame Computer

Super Computer : 물리적 크기 말고, Cluster Computer (네트워크를 모아서 Super Computer의 영향과 비슷한 형태로 만들어주는것)