프로그래밍 언어 3차 과제

20181256 수학, 컴퓨터공학과 김도현

test.c 캡쳐본.

텍스트이(가) 표시된 사진

자동 생성된 설명

test.s 캡쳐본

텍스트이(가) 표시된 사진

자동 생성된 설명

어셈블리 코드 해석. 어셈블리어를 통해 6단원에서 배운 데이터 타입이 어떻게 메모리를 할당하고 어떤 크기를 가지는지에 대해 주로 분석해보았다.

file “test.c” 에 관련된 어셈블리 코드이다.

main: 위에 써 있는 곳에 a, bb, user\_computer, dd, cc, q가 있는 것으로 보아 글로벌 변수에 관련된 내용이 써져 있음을 확인 할 수 있다.

<a,3375>

a는 char 형의 15\*15\*15 배열이다. a의 크기는 계산을 해보면 15\*15\*15=3375 임을 알고 있다. 즉 char 형은 각 element당 1 byte로 부여하기 때문에 a는 1\*3375=3375 만큼의 메모리 크기를 갖는다고 생각할 수 있다.

<bb,8>

Struct 형인 bb는 char b, float d로 구성되었다. 우리는 struct 안에 들어오는 변수들의 모든 메모리 크기를 다 가지고 있음을 알고 있다. char 는 1byte, float 는 4byte 만큼의 크기를 가져 5byte일 것 같지만 여기서는 8byte가 부여되었다. 즉 char에 대해 3byte만큼의 추가 메모리, 즉 word alignment가 생겼다고 판단할 수 있다.

<user\_computer,4>

user\_computer는 enumeration, enum 형으로 c언어에서는 implicitly 하게 integer로 변환된다. int 처럼 생각하면 된다. 즉 4byte가 부여되었다. (Apple -0, Samsung-1, Dell-2)

<dd,8>

dd는 union 형으로 struct형은 모든 변수에 관해 메모리 할당이 일어나지만 union은 내부에서 들어있는 타입들 중 가장 큰 메모리 크기를 가지는 것으로 메모리 할당이 일어난다. 여기서는 struct b 8byte, char hi 는 1byte를 예상할 수 있어 union dd의 크기는 b의 메모리인 8 byte임을 확인해 볼 수 있다.

<cc,8>

이후 char \*\*cc에 8 byte가 부여되었다. 수업시간에는 32bit 컴퓨터에 관해 주로 배웠기 때문에 포인터 크기는 일반적으로 word에 해당하는 4byte라 배웠다. 하지만 해당 test.c는 64bit 컴퓨터이기 때문에 4byte가 아닌 8byte로 예상할 수 있다.(하지만 정확히 확인하기 위해서는 어떤 회사의 어떤 cpu를 사용하고 있는지 세부사항을 확인해야 할 것이다.)

<q,1>

또한 char q는 단순한 char 형이기에 1byte를 받아온다.

main: 아래 코드, 즉 어떻게 main이 돌아가는지에 대해 분석 할 것이다.

movzbl a+2405 를 분석하자면 a에 2405에 해당하는 위치를 부여한다는 것이다. a[10][10][5] 에서 c언어는 row major 언어이기 때문에 10\*15\*15+10\*15+5=2405 가 나온다. 첫번째 index는 0~9까지 각각 15\*15개가 있고 그중 두번째 index 는 0~9까지 각각 15개, 이후 [10][10][0] 부터 [10][10][4] 까지 5개가 있다. movzbl 를 보아 zero extensiton한 결과를 %eax에 넣고 movb는 char 형에 대한 것으로 a를 char 형인 q에 넣는다고 생각해볼 수 있다. (q=a[10][10][10])

movss 는 단정도 부동 소수점 값 복사에 관련된 것으로 즉 float 와 연관 있는 어셈블리 명령어 임을 알 수 있다. Struct 안의 float를 의미하는 bb.d 코드이다. 여기서는 .LCO 즉 메모리 주소를 꺼내 %xmm()에 옮기고 이걸 bb+4에 넣어준다. 즉 struct 구조상 char b 가 먼저 있기 때문에 float d 앞에는 4byte(1byte+word alignment 3byte)가 이미 존재한다. 그래서 bb+4인 곳에 넣어주는 것이다.(bb.d=4.0)

여기서는 union 형인 안의 struct 형인 dd.b.b에 q를 넣는 작업이다. char 형인 q 를 꺼내어 %eax에 넣고 이를 다시 movb 즉 char 형으로 dd에 넣는다. 이때 dd를 분석하면 union은 값이 받아질 때 해당 받아오는 데이터형으로 인식하기 때문에 그대로 struct b역시 dd의 메모리 주소를 사용한다. 그리고 struct에서도 char b가 가장 선행되어 오기 때문에 해당 주소가 dd가 dd.b.b 주소임을 알 수 있다. (dd.b.b=q)

이후 enum에 관련된 코드인데 enum 안에는 Apple, Samsung, Dell 3개가 있다. C언어에서는 Apple을 0, Samsung을 1, Dell 을 2로 함축적으로 인식하고 있다. $0이라는 의미는 Apple을 의미한다. 이를 user\_computer로 넣어준다. Movl은 c언어에서 int 형으로 인식하고 있다는 의미이다. (user\_computer=Apple)

마지막은 포인터형에 관련된 코드인데 dd.hi의 주소는 dd가 union이기 때문에 hi와 상관없이 dd의 주소가 오게 된다. 이를 포인터 char 형인 \*cc, 즉 cc에 넣어준다. 이때 movq는 해당 컴퓨터에서 char 포인터 크기가 8byte가 할당되기 때문에 이를 사용하여 cc에 넣었다. (cc=&dd.hi)