맨 앞자리 비트값을 1로하면 음수

음수값을 알고싶을때

1은 0으로 0은 1로 바꿔주고 1더하기

0101 1001 -> 1010 0110 -> 1010 0111

2진수 표기 0b0 0b1 0b10 0b11 0b100

16진수 표기 0x1 0x2 ... 0xF 0x10 0x11 ... 0x1F 0x20

8bit = 1byte

16bit = 2byte = 1word

32bit = 3byte = 2word = 1 dword (double-word)

64bit = 4byte = 4word = 1 qword (quad-word)

EAX

EBX

ECX

EDX

mov eax -> A레지스터를 사용할건데 하위 32비트만 사용할것이다

mov rbx -> B레지스터를 사용할건데 64비트 전부 이용할것이다

mov cl -> C레지스터를 사용할건데 하위 8bit만 사용할것이다

section .data

a db 0x11

b dw 0x2222

c dd 0x33333333

d dq 0x4444444444444444

;초기화 되지 않은 데이터

; [변수이름] [크기] [개수]

; [크기] resb(1) resw(2) resd(4) resq(8)

section .bss

e resb 10

add // 더하기

; 더하기 연산

; add a, b (a = a+b)

; a는 레지스터 or 메모리

; b는 레지스터 or 메모리 or 상수

; -단! a, b모두 메모리는 x

mul // 곱하기

sub // 빼기

div // 나누기

shift 연산

shl shr

<< >>

한비트씩 왼쪽 / 오른쪽 으로 이동

최상위 비트 음수부호는 안바뀜

logical (논리 연산)

not, and, or, xor

조건 a, 조건 b

xor : a, b 둘다 true이거나 false이면 0, 아니면 1

-> 둘이 같으면 0 둘이 다르면 1

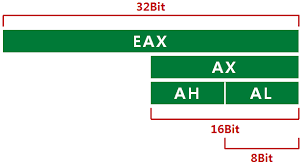
분기문

명령어 : cmp jmp je ...

레지스터 데이터 크기에 따른 이름?

Rax 64bit

Eax 32bit …



범용 레지스터 및 레지스터 종류들

A b c d 범용 레지스터

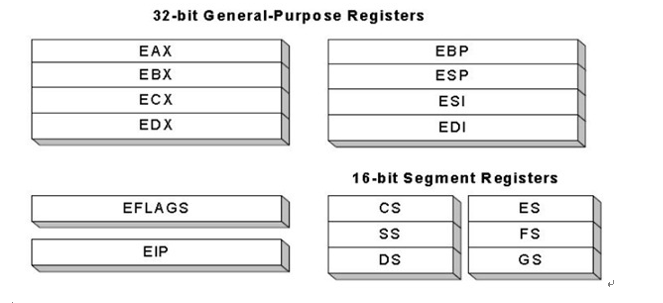
포인터 레지스터

Ip ( instruction pointer) : 다음 수행 명령어의 위치

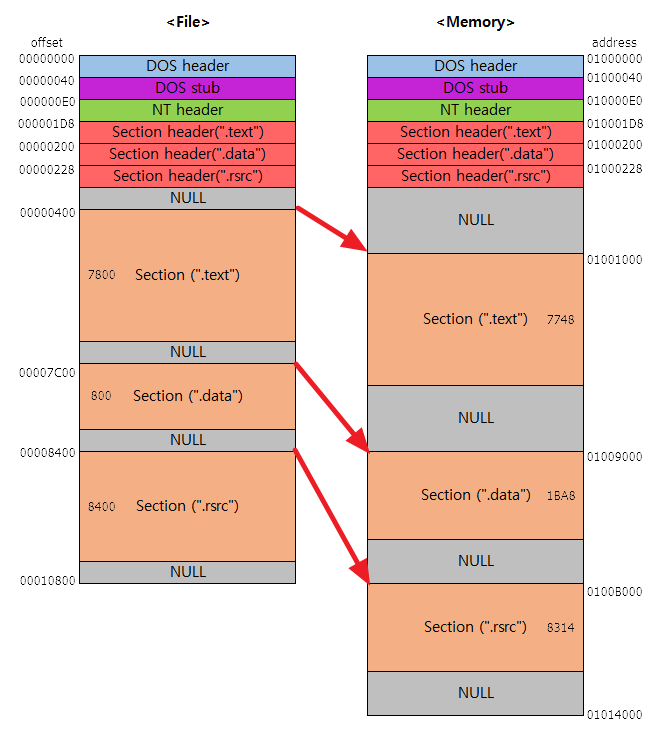
Sp (Stack Pointer) : 현재 스택 top 위치

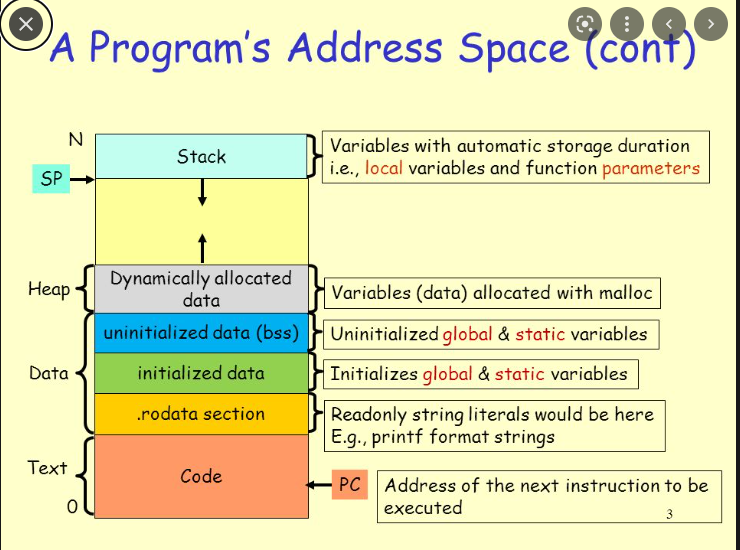
Bp (Base Pointer) : 스택 상대주소 계산용

Bp : 내 함수 스택 주소값을 임시 저장



Exe 실행 파일 구조





텍스트이(가) 표시된 사진

자동 생성된 설명

.rodata ( read only data)

Const char\* msg = “Hello World”;

.rodata에 “Hello World”가 박힌다

Const int INAIR = 0; 도 read only data에

테이블이(가) 표시된 사진

자동 생성된 설명

64bit windows OS 주소의 크기가 0x0000000000000000 = 8byte

32bit -> 4byte

주소가 가르키는 값의 byte크기는? Int ,char, short 등 type에 따라서 다른데

Int\* ptr -> 64bit환경에서는 sizeof(ptr) == 8byte;

32비트 CPU에서는 1워드(word)의 크기가 4바이트이므로, 포인터 변수의 크기 또한 4바이트가 될 것입니다

CPU에서의 워드란 CPU가 한 번에 처리할 수 있는 데이터의 크기

64bti 처리 어셈블리어를 기계어로 변환한 것을 한번에 64개의 bit를 처리

포인터 변수의 크기는 CPU의 종류와 컴파일할 때 사용된 컴파일러의 정책에 따라서 달라질 수 있습니다. 64bit CPU 0> 한 사이클에 64bit를 처리한다는 의미

클럭(헤르츠): 2.5GHz -> 초당 25억번의 사이클 실행

CPU는 초당 25억 *64 = 2,500,000,000 x* 64 = 160,000,000,000 비트를 처리한다.

메모리 구조

BSS : 초기값이 0이거나, 초기화 안된 전역변수

int num = 10;

int num1 = 10;

int num2 = 10;

int num3 = 10;

int num4 = 10;

0x000000D65D2FF5D4 0000000a 72e40800 00007ff9 70640040 00007ff9 70640040 00007ff9 130f50a7

0x000000D65D2FF5F4 0000000a 70640040 00007ff9 00000002 000000d6 7064005b 00007ff9 00000005

0x000000D65D2FF614 0000000a f36e4870 00007ff8 f36e4868 00007ff8 72e40800 00007ff9 00000000

0x000000D65D2FF634 0000000a f35d8ed3 00007ff8 00000000 00000000 00000000 00000000 00000000

0x000000D65D2FF5D4 주소에 들어있는 값 0a = 10

0x000000D65D2FF5D5 주소에 들어있는 값 00

주소값의 크기 8byte x 이 주소에 들어있는 값을 4byte씩 봤을 때고

주소단위를 4만큼씩 봤을 때

0x000000D65D2FF5D4 주소에 들어있는 값 0000000a

0x000000D65D2FF5D8 주소에 들어있는 값 72e40800

한번에 이만큼씩 처리함 0000000a 72e40800

이건 왜 32 뒤에 그다음 변수가 저장이될까 dword ptr을 썼기 때문에 같음 (추측) 32bit

0x64 = 0b0110 0100 -> 16진수 2자리에 1byte

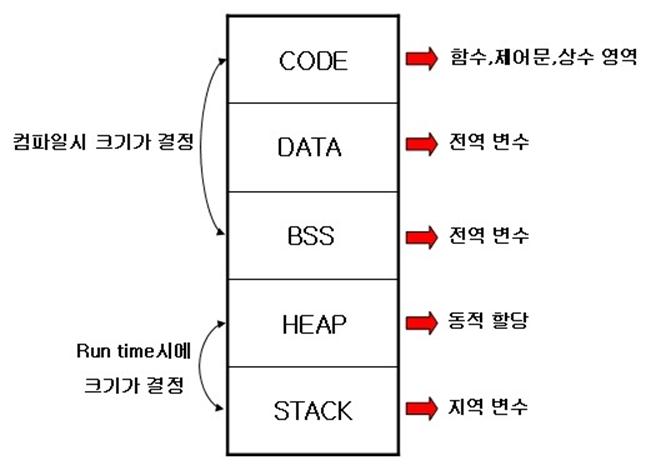
0x04 = 0b0000 0100

텍스트, 모니터, 스크린샷, 실내이(가) 표시된 사진

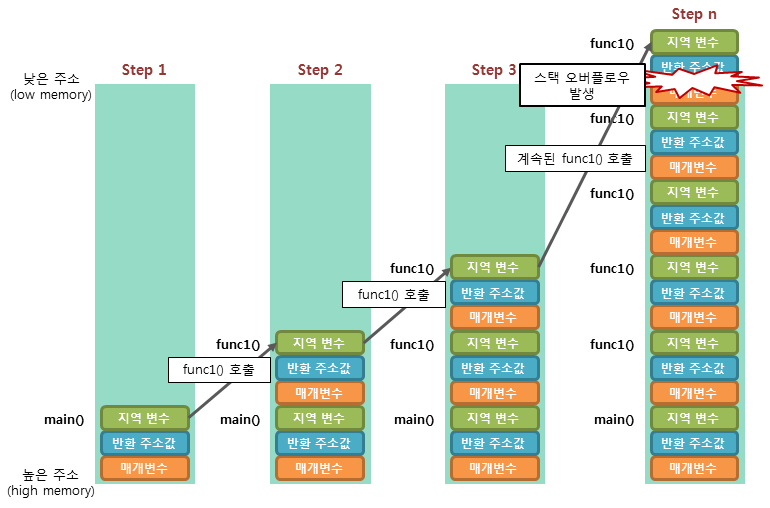
자동 생성된 설명

64bit windows os에서는 데이터를 한번에 8byte씩 처리함

데이터를 어떻게 처리하나요?



스택 메모리 구조 높은주소부터 들어감



빅엔디안 리틀엔디안

https://code-lab1.tistory.com/179

테이블이(가) 표시된 사진

자동 생성된 설명

네트워크에서 빅엔디안을 많이 쓰며

그외에는 리틀엔디안을 많이씀 숫자의 대소비교가 간편함