레지스터 정리

- 레지스터 | Register
  - 컴퓨터의 프로세서 내에서 자료를 보관하는 아주 빠른 기억 장소
  - 메모리 계층의 최상위에 위치하여, 가장 빠른 속도로 접근 가능함
- Basic Program Execution Register : 기본적인 프로그램 실행 레지스터
  - General Purpose Registers | 범용 레지스터 (32 bit)
    - EAX (Extended Accumulator Register) : 산술, 논리 연산 수행 및 함수의 리턴 값에 사용. Win32 API 함수들은 return 값을 EAX에 저장함
    - EBX (Extended Base Register) : 메모리 주소를 저장하기 위해 사용됨
    - ECX (Extended Counter Register) : 반복문 명령어에서 반복 카운터로 사용됨
    - EDX (Extended Data Register) : EAX 레지스터와 같이, 부호 확장 명령 등에 쓰임. 큰 수의 곱셈 또는 나눗셈 등의 연산에서 EAX와 함께 쓰임
      - Win32 API에서는 함수 내부에서 ECX, EDX를 사용하므로 그 전에 쓰던 값은 다른 레지스터에 저장해두어야 함.
    - ESI (Extended Source Index) : 데이터를 조작 혹은 복사시에 데이터의 주소를 저장함.
    - EDI (Extended Destination Index) : 데이터를 조작 혹은 복사시에 목적지의 주소를 저장함.
      - 따라서 ESI와 EDI는 [ESI] → [EDI]로 복사할 때 많이 사용됨
    - EBP (Extended Base Pointer) : 하나의 스택 프레임의 시작 지점 주소를 저장함.
      - 스택 프레임 : ESP가 아닌 EBP를 사용하여 스택 내의 로컬 변수, 파라미터, 복귀 주소에 접근하는 기법
        - ESP는 프로그램 내에서 수시로 변경되기에, 스택에 저장된 변수, 파라미터에 접근하고자 할 때 ESP를 사용하면 프로그램 작성이 어려워짐. 따라서 함수 시작 때의 ESP를 EBP에 저장해 준다면, EBP를 기준으로 안전하게 변수 등에 접근 가능

```
PUSH EBP            ;함수의 시작 (함수의 시작 전, EBP의 초기 값을 스택에 저장)
MOV EBP ESP         ;현재 ESP 값을 EBP에 저장

--- 함수의 본체 ---

MOV ESP EBP         ;ESP 값을 함수가 시작했을 때 초기값으로 변경
POP EBP             ;리턴 전, 함수의 시작에서 저장해 놓았던 EBP값 가져옴
RETN                ;함수 종료
```

    - ESP (Extended Stack Pointer) : 하나의 스택 프레임의 끝 지점 주소를 저장함.

- Segment Regitsters (16 bit)
    - 가상 메모리 : 각 프로그램에 실제 메모리 주소가 아닌 가상의 메모리 주소를 주는 방식
    - Segmentation : 메모리를 서로 크기가 다른 논리적인 블록 단위인 '세그먼트' 로 분할하고 할당하여 물리 주소를 논리 주소로 변환하는 것. 이때, 세그먼트 메모리는 SDT(Segment Descriptor Table)에 기술되어 있는데, Segment Register은 이 SDT의 Index를 가짐
    - Paging : 가상 메모리를 모두 같은 크기의 블록으로 편성하여 운용하는 기법
    - CS (Code Segment) : 프로그램의 코드 세그먼트의 시작 주소를 포함. 이 세그먼트 주소에 명령어 포인터 (IP) 레지스터의 오프셋 값을 더하면 실행하기 위해 메모리로부터 가져와야 할 명령어의 주소가 됨. → 명령어 코드의 베이스 주소
    - SS (Stack Segment) : 메모리 상에 주소와 데이터의 임시 저장 목적인 스택의 구현을 가능케 함. 이 주소에 스택 포인터 (SP) 레지스터의 오프셋 값을 더하면 참조되고 있는 스택의 현재 워드를 가리킴
    - DS (Data Segment) : 프로그램의 데이터 세그먼트 시작 주소를 포함함. 명령어는 이 주소를 사용해 데이터의 위치를 알아냄.
    - ES (Extra Data Segment) : 메모리 주소 지정을 다루는 스트림 연산에서 사용됨. 이 때, ES 레지스터는 DI (Destination Index) 레지스터와 연관됨.
    - FS, GS : 추가적인 데이터 세그먼트

- EFLAGS Register (32 bit) : 프로그램을 실행하거나 프로세스 제한을 제어하는 레지스터

- 목적에 따라 상태 플래그, 제어 플래그, 시스템 플래그로 나뉨.
- 이 중 상태 플래그는 조건 분기 명령어에서 플래그의 값을 확인해 동작 수행 여부를 결정하기에 중요함
- CF (Carry Flag) : 산술 연산 수행 시, 자리 올림 혹은 내림이 발생하면 set(1)이 됨.
- OF (Overflow Flag) : 산술 연산 결과의 오버플로우 여부를 표시함. 연산의 결과값이 MSB(부호 표시 비트)를 제외한 최대 허용 정수값보다 크거나 최소 정수보다 작으면 set(1)이 됨.
  - Signed 연산 오버플로우 → OF = 1
  - Unsigned 연산 오버플로우 → CF = 1
- SF (Sign Flag) : MSB와 같은 값을 가지는 부호를 표시하는 역할.
- ZF (Zero Flag) : 연산 결과가 0 혹은 비교연산 '='에서 set(1)이 됨.

- EIP Register (32 bit) : 다음 명령어 실행을 저장하는 포인터 레지스터
  - CPU는 이 레지스터에 저장된 메모리 주소의 명령어를 하나 처리하고 난 뒤, 자동으로 그 명령어 길이만큼 EIP 레지스터의 값을 증가시켜 명령어를 처리해 나감
  - 직접 변경할 수 없는 레지스터이므로, JMP 등의 특정 명령어를 사용하거나 인터럽트 혹은 예외를 발생시켜야만 변경할 수 있음

참고

프로세서 레지스터

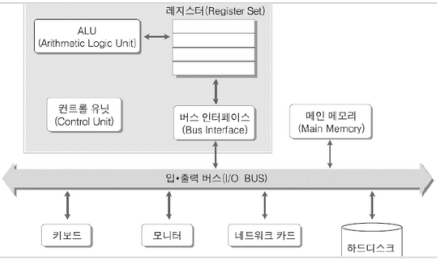
프로세서 레지스터(processor register, 기록기) 또는 단순히 레지스터는 컴퓨터의 프로세서 내에서 자료를 보관하는 아주 빠른 기억 장소이다. 일반적으로 현재 계산을 수행중인 값을 저장하는

𝕎 https://ko.wikipedia.org/wiki/%ED%94%84%EB%A1%9C%EC%84%B8%EC%84%9C_%EB%A0%88%EC%A7%80%EC%8A%A4%ED%84%B0

Basic Program Execution Registers

Reversing (1) Register 레지스터 (Register) 는 컴퓨터의 프로세서 내에서 자료를 보관하는 아주 빠른 기억 장소임 . 일반적으로 현재 계산을 수행 중인 값을 저장하는 데 사용됨 . 대부분의 현대

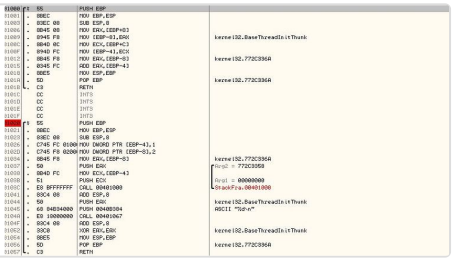
https://m.blog.naver.com/PostView.nhn?blogId=koromoon&logNo=120204956555&proxyReferer=https%3A%2F%2Fwww.google.com%2F



Red_Seek :: 스택 프레임

안녕하세요. Seek입니다. 오늘은 디버깅할 때 큰 도움이 되는 내용을 공부하고자 합니다. 바로 스택 프레임입니다. 스택 프레임을 이해하고 나면 스택에 저장된 함수 변수와 함수 로컬 변수 등이 쉽

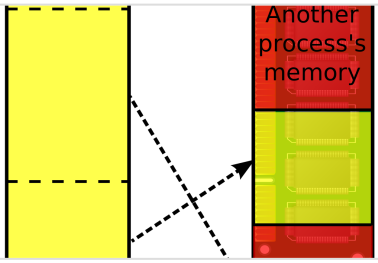
https://redscreen.tistory.com/10



가상 메모리

가상 메모리(가상기억기) 또는 가상 기억 장치는 RAM 을 관리하는 방법의 하나로, 각 프로그램에 실제 메모리 주소가 아닌 가상의 메모리 주소를 주는 방식을 말한다. 이러한 방식은 멀티태스킹 운

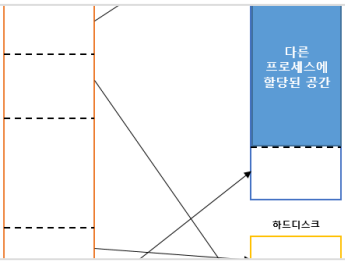
https://ko.wikipedia.org/wiki/%EA%B0%80%EC%83%81_%EB%A9%94%EB%AA%A8%EB%A6%AC



세그먼테이션(Segmentation) 과 페이징(Paging)

Linux 세그먼테이션(Segmentation) 과 페이징(Paging) 가상메모리를 관리하는 기법인 세그먼테이션과 페이징에 대해 알아 보자. 먼저 는 메모리에 로드된 즉, 실행중인 프로세스가 가상의 공간

https://m.blog.naver.com/s2kiess/220149980093



페이징

페이징 기법(paging)은 컴퓨터가 메인 메모리에서 사용하기 위해 2차 기억 장치로부터 데이터를 저장하고 검색하는 메모리 관리 기법이다. 즉, 가상기억장치를 모두 같은 크기의 블록으로 편성하여

https://ko.wikipedia.org/wiki/%ED%8E%98%EC%9D%B4%EC%A7%95