gcc의 builtin 함수 정리

void *__builtin_return_address(unsigned int LEVEL)
함수의 리턴 주소,즉 함수를 호출한 지점의 주소를 반환한다. LEVEL로 함수를 몇 번 거슬러 올라갈지 결정할 수 있다. x86 에서는 %ebp+4 의 위치에 대한 포인터가 리턴된다.

void *__builtin_frame_address(unsigned int LEVEL)
함수의 프레임 포인터를 반환한다. LEVEL은 위의 함수와 동일하고 x86에서는 %ebp의 위치에 대한 포인터가 리턴된다.

int __builtin_types_compatible_p(TYPE1,TYPE2)
TYPE1과 TYPE2의 호환성을 검사한다.

TYPE __builtin_choose_expr(CONST_EXP,EXP1,EXP2)
CONST_EXP ? EXP1 : EXP2 와 동일한 기능을 한다. 다만 컴파일 시에 결정된다.

int __builtin_constant_p(EXP)
매개변수 EXP가 정수인지 판단

long __builtin_expect(long EXP,long C)
EXP의 값이 C의 값이 될 경우가 많을 것이라는 판단의 근거로, 분기를 최적화 할때 사용

void __builtin_prefetch(const void *ADDR,int RW, int LOCALITY)
ADDR에 있는 데이터를 캐시에 prefetch하자고 할 경우 사용

__attribute__

```
ex)  int foo(int n) __attribute__((애트리뷰트));
     int foo(int n)
     {
     ...
     }

     or

     __attribute__ ((애트리뷰트)) int foo(int n)
     {
     ...
     }
```

종류
constructor
main 함수가 호출되기 전이나 공유 오브젝트에서 함수 실행이 가능하게...

destructor
exit 하기 전이나 공유 오브젝트가 언로드되기 직전에 함수 실행이 가능하게...

cleanup
auto 변수가 주어진 영역에서 벗어나 없어질 때 호출되는 함수를 지정

section
특정 섹션에 코드를 배치

used
어디에서도 호출하지 않는 함수이더라도 반드시 코드를 생성(어셈블러 코드에서 호출하는 경우에 사용)

weak
weak심볼인 코드를 생성

alias
심볼의 별칭을 지정한다. 보통 weak와 함께 사용

stdcall
x86 호출 규약. 스택을 호출된 함수에서 pop

cdecl
x86 호출 규약. 스택을 호출한 함수에서 pop

fastcall
x86 호출 규약. 처음 두 매개변수를 %ecx, % edx를 이용해 호출

regparm
레지스터에 전달할 매개변수의 개수를 제어

vector_size
변수의 벡터 크기를 지정

dllimport
MS 윈도우의 dllimport

dllexport
MS 윈도우의 dllexport

const
매개변수 만으로 반환 값이 결정

malloc
NULL 외의 반환 값이 다른 포인터와 공유되지 않는 경우에 사용

noreturn
exit(2)와 같이 반환되지 않는 함수에 사용

noinline
인라인 전개되기를 원하지 않는 경우에 사용

always_inline
최적화 되지 않더라도 항상 inline 전개

nonnull
NULL이 될 수 없는 포인터 매개변수를 가리킨다.

unused
사용하지 않을 경우에도 경고 출력 X

deprecated
사용될 경우에 경고를 출력

데이터에 관한 attribute

aligned
변수 영역의 정렬을 제어

packed
구조체 내부에 정렬에 의한 채우기를 최소화

common
변수를 common영역에 배치

nocommon
변수를 common 영역에 배치하지 않음

shared
DLL을 사용한 프로세스 전체로 공유된 매개변수에 사용