SIMD프로그래밍 소개

SIMD?

- Single Instruction Multiple Data
- 명령어 하나로 여러 개의 데이터를 처리
- MMX (16 bits 정수 4개)
- SSE (32 bits float 4개, 64 bits double 1개)
  - 버전이 올라가면 정수도 처리 가능
- AVX (32 bits float 8개, 64 bits double 2개)
  - 정수 처리 가능
- AVX512(32 bits float 16개, 64 bits double 4개)

SIMD - registers

- SSE – XMM 레지스터 사용 XMM0 – XMM7
  - 64Bits 모드에선 XMM0 – XMM15
- AVX – YMM레지스터 사용 YMM0 – YMM7
  - 64Bits 모드에선 YMM0 – YMM15
- AVX512 – ZMM레지스터 사용 ZMM0 - ZMM31

SIMD – 컴파일러 & OS 지원

- 컴파일러가 자동으로 SIMD명령을 사용하기도 함.
- 현세대 컴파일러는 32 bits float의 경우 기본적으로 FPU대신 SSE레지스터를 사용(단 Multiple이 아닌 Single처리)
- Windows OS 어느 버전부터(아마도 Windows 7?)는 내부적으로 SIMD명령을 사용함.
- MSVCRT의 경우 내부적으로 SIMD사용
- 수학함수등 목적이 확실한 경우 컴파일러의 최적화를 기대하지 말고 직접 SIMD코를 작성해야 한다.
- ASM or Compiler Intrinsic(C/C++)으로 작성

SIMD코드 작성 – Assembly language

- Debug/release 모드 상관없이 항상 균일한 속도 보장
- C/C++함수 안에 인라인시키면 최적화에 지장을 준다. 따라서 함수 하나 단위로 완전히 ASM코드로 짤 것.
- VC++의 경우 x86모드는 인라인 어셈블리로 가능, x64모드인 경우 .asm파일로 작성해서 ml64를 사용.

SIMD코드 작성 – Compiler Intrinsic

- asm명령에 대응하는 Compiler built-in 함수.
- asm명령과 거~~~의 1:1 대응
  - 1:1대응이 아닌 경우도 있다.
- 쉽고 편하다. SSE만 알면 AVX, AVX512도 추가 지식 없이 적용 가능.
- Release 모드에선 asm코드와 거의 속도차이 없음.
- Debug 모드에선 엄청 느림. 따라서 충분히 검증된 후에는 SIMD코드를 별도 DLL로 분리할 필요가 있다.

Copy (load, save)

movaps
_mm_load_ps()
_mm_store_ps()

movups
_mm_loadu_ps()

movhlps
_mm_movehl_ps()

movlhps
_mm_movelh_ps()

Copy (load, save)

movlps
_mm_loadl_pi()
_mm_storel_pi()

movss
_mm_load_ss()
_mm_store_ss()

산술연산 – 덧셈, 뺄셈, 곱셈, 나눗셈

addps
_mm_add_ps()

subps
_mm_sub_ps()

mulps
_mm_mul_ps()

divps
_mm_div_ps()

산술연산 – 역수, 평방근, 평방근의 역수

rcpps
_mm_rcp_ps()

sqrtps
_mm_sqrt_ps()

rsqrtss
_mm_rsqrt_ps()

비교 치환

cmpps
_mm_cmpeq_ps()
_mm_cmpge_ps()
_mm_cmpgt_ps()
_mm_cmple_ps()
_mm_cmplt_ps()

비교 분기

comiss
_mm_comieq_ss()
_mm_comige_ss()
_mm_comigt_ss()
_mm_comile_ss()
_mm_comilt_ss()

shuffle

shufps
_mm_shuffle_ps()

수평 합

haddps
_mm_haddps()

변환 (정수 <-> 실수)

32bits float 4샘플 -> 32비트 int 4샘플
cvttps2dq
_mm_cvttps_epi32()

32 bits int 4샘플 -> 32bits float 4샘플
cvtdq2ps
_mm_cvtepi32_ps()

32 bits float 1샘플 -> 32 bits int 1샘플
cvttss2si
_mm_cvttss_si32()

32 bits int 1샘플 -> 32 bits float 1샘플
cvtsi2ss
_mm_cvtsi32_ss()

캐스팅 (바이너리 그대로)

_mm_castsi128_ps()

_mm_castps_si128()

movaps로 구현

참고자료

- Intel® Intrinsics Guide

(https://software.intel.com/sites/landingpage/IntrinsicsGuide/)

Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 2 (2A, 2B, 2C & 2D): Instruction Set Reference, A-Z

(https://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-instruction-set-reference-manual-325383.pdf)