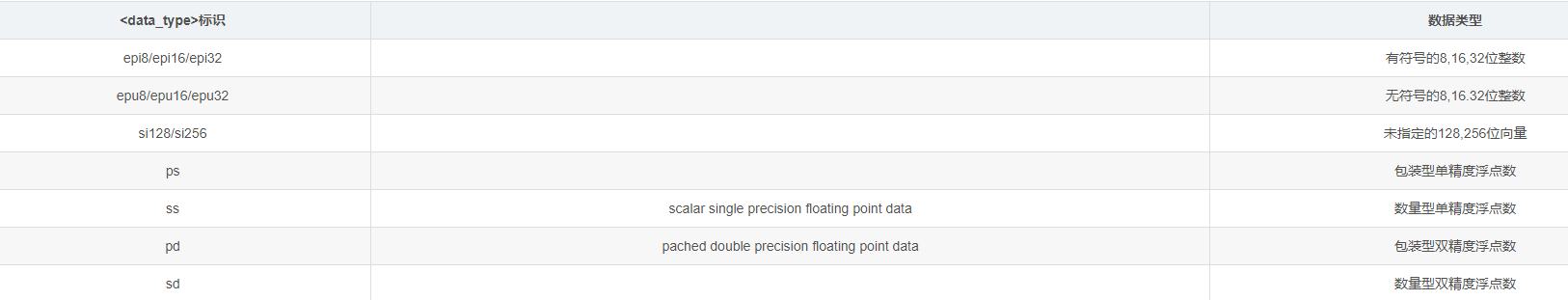
1. 背景介绍

SIMD（Single Instruction Multiple Data）指令集，指单指令多数据流技术，可用一组指令对多组数据通进行并行操作。详细参考<https://blog.csdn.net/qq_32916805/article/details/117637192>。

1. 常见数据类型与函数命名规则
2. 常用数据类型命名规则  
   • 每一种类型，从1个下划线开头，接一个m，然后是向量的位长度  
   • 如果向量类型是以d结束的，那么向量里面是double类型的数字。如果没有后缀，就代表向量只包含float类型的数字  
   • 整形的向量可以包含各种类型的整形数，例如char,short,unsigned long。也就是说，\_\_m256i可以包含32个char，16个short类型，8个int类型，4个long类型。这些整形数可以是有符号类型也可以是无符号类型  
   
3. 常用函数命名规则   
   函数命名：\_mm<bit\_width>\_<name>\_<data\_type>  
   <bit\_width> 表明了向量的位长度，即操作对象的数据类型大小，对于128位的向量，这个参数为空，对于256位的向量，这个参数为25  
   <name>描述了内联函数的算术操作。一般由两部分组成  
   第一部分是表示指令的作用，比如加法add等;  
   第二部分是可选的修饰符，表示一些特殊的作用，比如从内存对齐，逆序加载等  
   <data\_type> 表明了操作的粒度，具体情形见下表  
     
   
4. SIMD工作流程

加载：常用set或load函数，数据从内存进入CPU寄存器；

处理：数据处理；

存回：常用store函数，数据从CPU寄存器进入内存

1. SIMD使用方法
2. 需要包含头文件#include <x86intrin.h>

2、编译SIMD函数库需要添加编译选项-msse2 -mssse3 -msse4.1 -msse4.2 -mavx

其中-msse2 -mssse3 -msse4.1 -msse4.2用来支持sse指令集，寄存器为128位，-mavx用来支持avx指令集，寄存器为256位

3、 -march=native选项告诉编译器为当前的CPU架构和指令集生成优化的代码

Makefile使用时可以如下

GCC = g++

CFLAGS = -O2 -std=c++14

SSEFLAGS = -msse2 -mssse3 -msse4.1 -msse4.2 -mavx -march=native

$(GCC) $(CFLAGS) $(SSEFLAGS) -o throughput.out throughput.cpp

1. 常用函数查询网址

[Intel® Intrinsics Guide](https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html#techs=AVX,AVX2&ig_expand=133,156,4233,4295,4374,6188,4879,6188)