No.

Date

Paralellism

Motivation: 2000年后，processor 遇到了功率墙 (power wall)
问题：虽可加晶体管数量，但因功率与热量限制，提升性能变得困难

Tow ways to improve performance:
① 多道程序：multiprogramming，并行运行多个程序
② 并行：让一个程序更快 topic

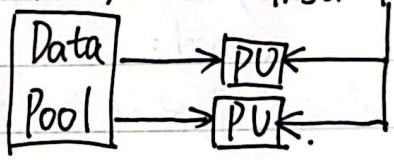
Recall Amdahl's Law: Speed up = $\frac{1}{(1-F)+F/S}$ 提升量 受影响比例

关于并行提升效率，有两种描述法：
① Strong scaling：Size of problem 不变，运行时间 $\times\frac{1}{n}$
② Weak scaling：变大，并行效率提高 (×n)，运行时间不变
↓更易实现 ↘更难，但更理想

Flynn's Taxonomy：将指令集与数据流并行性分为四种：
SISD SIMD MISD MIMD
↓
Single instruction, single data

但SIMD不同：利用单一指令流来同时处理多个数据流：

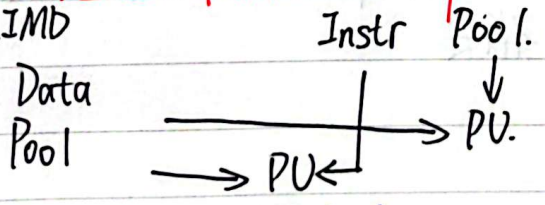
SIMD.



PU: processing unit
应用举例：Intel SIMD Instr Extensions：允许多个数据元素并行处理

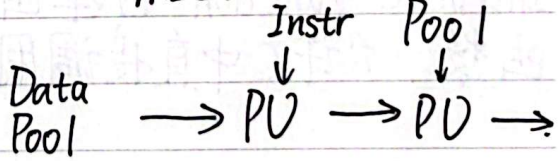
SIMD 是这一节的 main topic！

MIMD



Multicore (之后会讲)

MISD



Rare!

KOKUYO

```
for (int i---
    for (int j-----).
        double  Cij=0
        for (int k=0---)
            Cij+= A [i+ k*N]*B[j*N+k]
        C[i+j*N] = Cij.
```

What is SIMD ?

考虑矩阵乘的代码：红色部分可视为一种"操作"，其涉及大量数据作mul

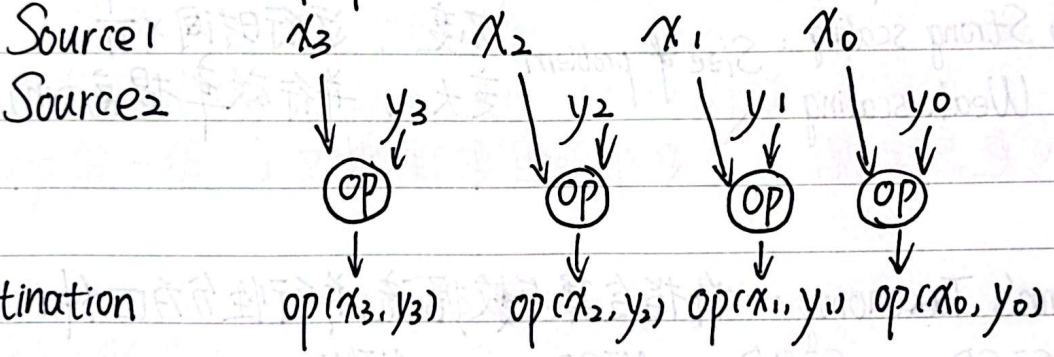
SIMD恰好就可提升此种 Data Level Processing

* One instr is fetched & decoded for entire operation
* SIMD 支持内存的并行访问. 这很重要！

那么现实中如何支持SIMD？ Inter X86 intrinsics 的 SIMD指令集有 SSE. AVX 等命令可允许扩展指令集

一般 SIMD指令会从两个源寄存器组中拿数据，送入op(operation)组件，将寄存器组会将计算结果存在一起

figure:



为了实现上述思想，甚至一些特殊数据类型被设计出来：

Packed byte：64 byte，存8个8bytes (MMX)

word： 4个16bytes

并且有寄存器专门存这种玩意儿，ns支持多个数据打包至单一reg.

Intrinsic Function：内建函数是高级语言中与汇编指令一一对应的函数，允许C中直接调用SIMD指令