Paralell Architectures

Why parallel architecture? The history of speeding up the computer performance (except for clock rate improvement thanks to technology scaling) has been driven by following major architectural innovations:

- **bit-level parallelism:** from 4-bit to 64-bit processors; doubling **the datapath widths** reduces the number of cycles to perform full 32-bit arithmetic operations; however, further increase in word width is driven mostly by 1) **improved floating-point representation** and 2) **larger address space**; now that we have large address space 2^{64}, it's unlikely we will need 128-bit processor any time soon
- **instruction-level parallelism:** in addition to **pipelining**, *multi-issue* **superscalar** processors have becom prevalent; however, increasing the amount of instruction-level parallelism (deeper pipelines) require every increasing resources (e.g. *more caches* and *more HW logic for branch prediction*; since longer pipeline and more-simultaneous-instruction issue introduces more dependenices resolving this dependencies require a lot of design effort and HW resources

  also, it's been observed that at-most 4-issue superscalar processor is the sort of limit that we can expect. (8-issue is not much better than 4-issue)

  reducing data dependencies can be done by **renaming**, which is simliar to SSA in compilers
- **thread-level parallelism:** current trend

Shared memory multiprocessors

- there is a global physical memory, there is a physical memory region which is shared by all processors (i.e. some shared portion of virtual address space of each processor is mapped to these physical pages)
- **SMP (symmetric multiprocessors)**
  - bus-based interconnect: not scalable since bandwidth is fixed (adding more processors reduces ...)
  - crossbar switch
  - multistage interconnection network (butterfly, hypercube, etc.)
- **NUMA (non-uniform memory access) MP**

Message passing multiprocessors

Data parallel processors