Concurrency

Why bother?

- Sequential Processors are reaching hard limits
  - Power Wall
    - Heat dissipation
  - Memory Bottleneck
    - Von Neumann Architecture
  - Physical Size Limits
    - Cannot go much smaller
  - Complexity Issues
    - Pipelining, Lookahead, Out of Order Execution, Instruction Level Parallelism
- Moore’s Law is in trouble
- New Architectures are Required
  - Multicores
  - General Purpose GPUs
  - Clusters – the new supercomputer architecture
  - Manycore (Xeon Phi)
- Each has its own issues to overcome
- Each requires a different programming approach
- ALL REQUIRE PARALLELISM

Definitions

- Concurrency is the decomposability property of a program, algorithm, or problem into order-independent or partially-ordered components or units
- Parallellism is a type of computation in which many calculations or the execution of processes are carried out simultaneously
- All parallel programs are concurrent but not all concurrent programs are parallel
- Processes and Threads
  - Both processes and threads are independent sequences of execution. The typical difference is that threads (of the same process) run in a shared memory space, while processes run in separate memory spaces.
  - Processes communicate via message passing commonly termed Inter Process Communication (IPC) (using e.g. TCP/IP)
  - As threads share memory space they can access each others memory (and therefore variables) directly commonly called the Shared Memory Approach
    - See also hardware threads, hyper threading and SMT

Issues in Concurrency

- Sequential Tools and Thinking
- Memory Bottlenecks
- No single model of Parallel Architecture yet
- Load Balancing
- Non determinism
- New Problems
  - Mutual Exclusion
- New categories of Error
  - Deadlock

What we want

- Performance Measures
  - Scalability
  - Speedup
  - Efficiency
- Portability
- Maintainability
- Determinism
- Composability
- Safety
- Ease of Development

OpenMP

- Open Multi-Processing is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming
- It works by allowing us to annotate code (new or existing) indicating where concurrency is allowed
  - Supports C, C++ and Fortran
- Latest version supports offloading

```
#include <stdio.h>
int main(void)
{
    #pragma omp parallel
    printf("Hello, world.\n");
    return 0;
}
```

-

Next Steps