OpenMP

What is it? Details and Setup!

Image Sources: https://computing.llnl.gov/tutorials/parallel_comp/

What is it?

- Open Specifications for Multi-processing
- Application Programming Interface (API)
- Works on top of a compiled programming language.

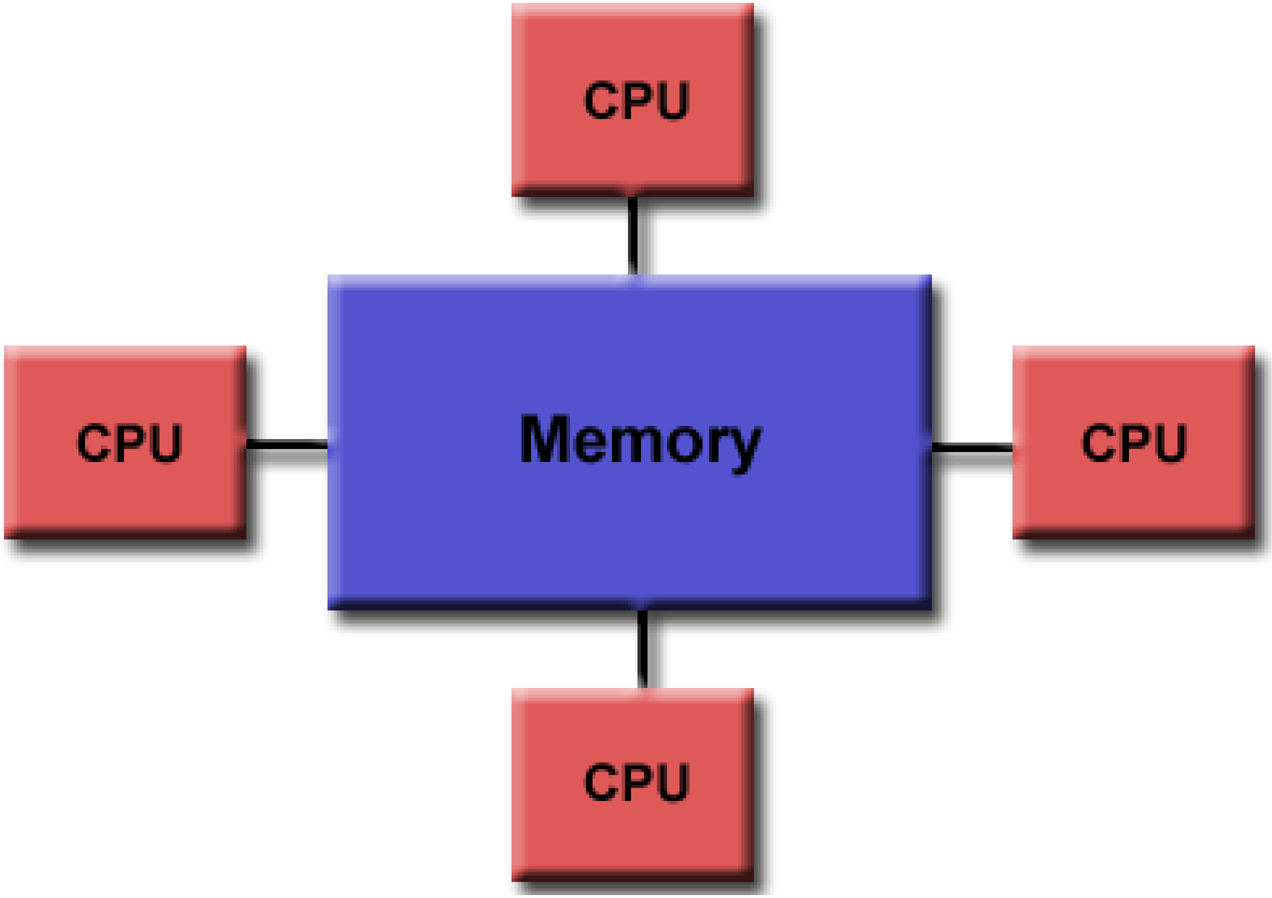
Applied to ...

- Compiled Programming Languages (C, C++, Fortran)
  - Because the code is optimized and all “memory” and functional requirements are known before the execution.
  - N/A to interpreted programming languages as this is not known before hand.

How does it Work?

- Uses pre-processor directives, functions and subroutines which COMMAND the compiler to do certain operation more stingently.
- Takes the machine codes and optimizes them even further.
- **So, taking a well working serial code and making slight tweaking using OpenMP does the job!**
- User should specify the regions where “parallelisation” is possible.
- Works on a shared memory system

Shared Memory System



Syntax

- Pre processor directives:

  ```
  !$ fortran code
  ```

  eg:

  ```
  !$ print *, (“Hello world from”, omp_get_thread_num())
  ```

Syntax

- Blocks or constructs

```
!$omp <construct name> <clause> <args>
......
!$omp end <construct name>
```

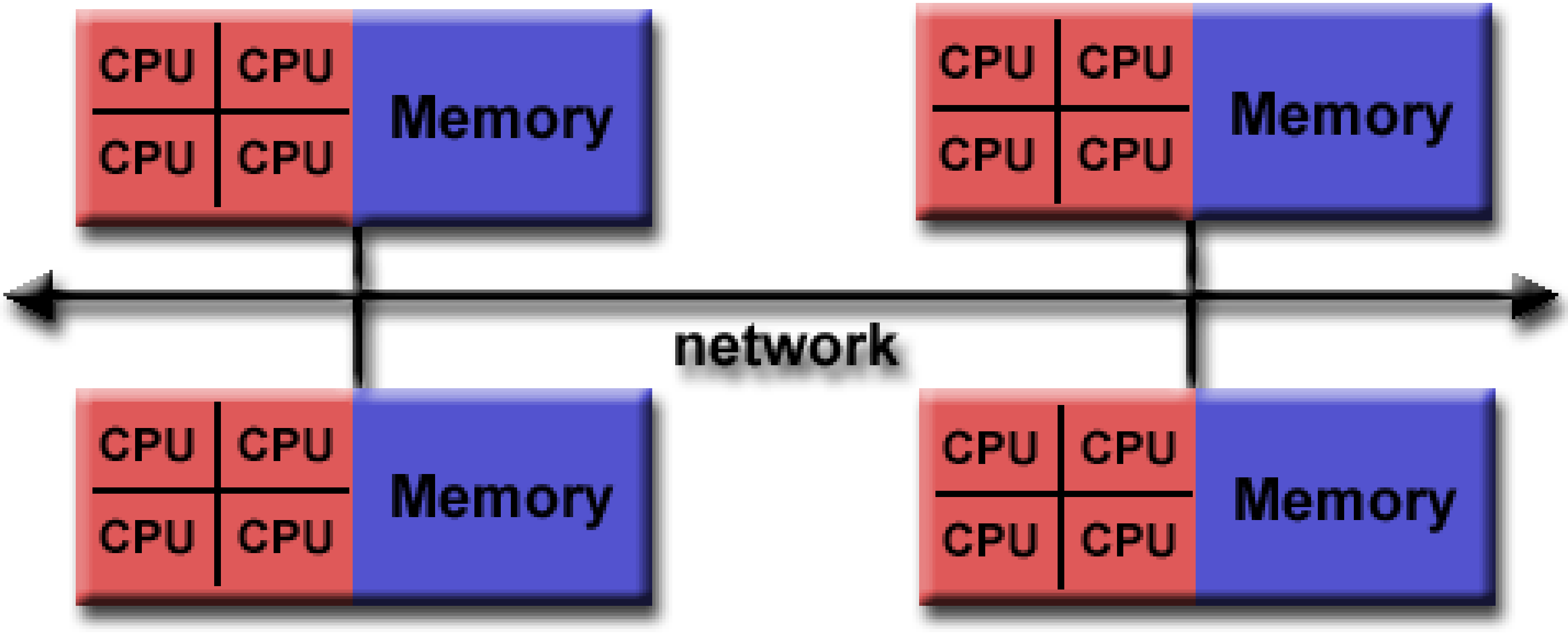
What do you need to Start with?

- GNU Fortran (preferrably a latest version)
  - It has pre-defined OpenMP commands, subroutines etc., built in it.

About MPI ...

- Message Passing Interface
  - Works on Distributed Memory Systems
  - The program is defined such that it runs on all processors
  - Selective elements are done by different processors
  - The results are “passed” between processors using “message” calls through “interfaces” - hence the name.
  - **Parallelized program cannot be run serially!!!**

Distributed Memory System


CPU
CPU
CPU
CPU
Memory
CPU
CPU
CPU
CPU
Memory
network
CPU
CPU
CPU
CPU
Memory
CPU
CPU
CPU
CPU
Memory