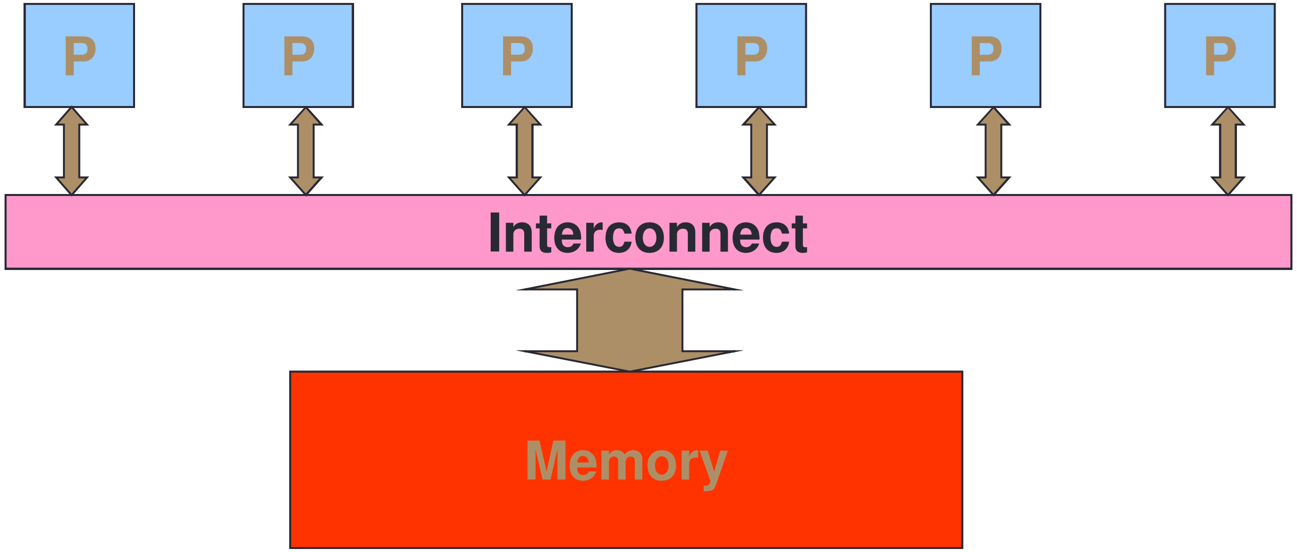
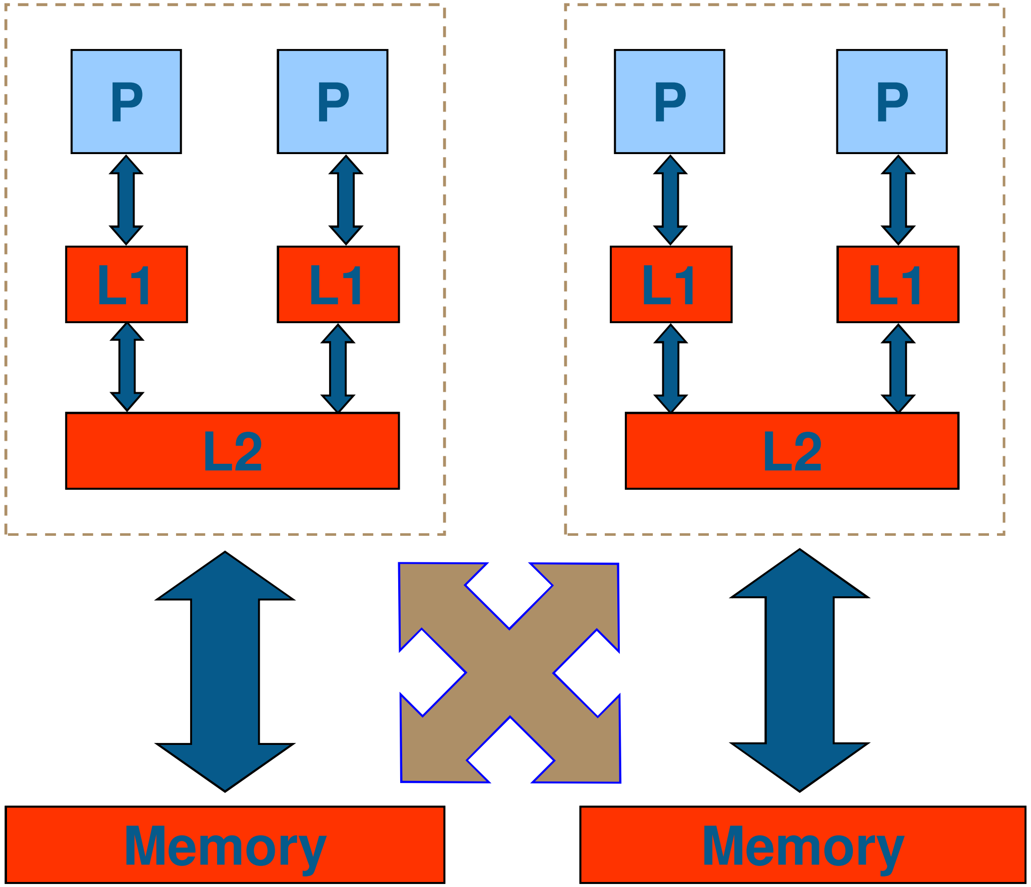
Introduction to OpenMP

Recap

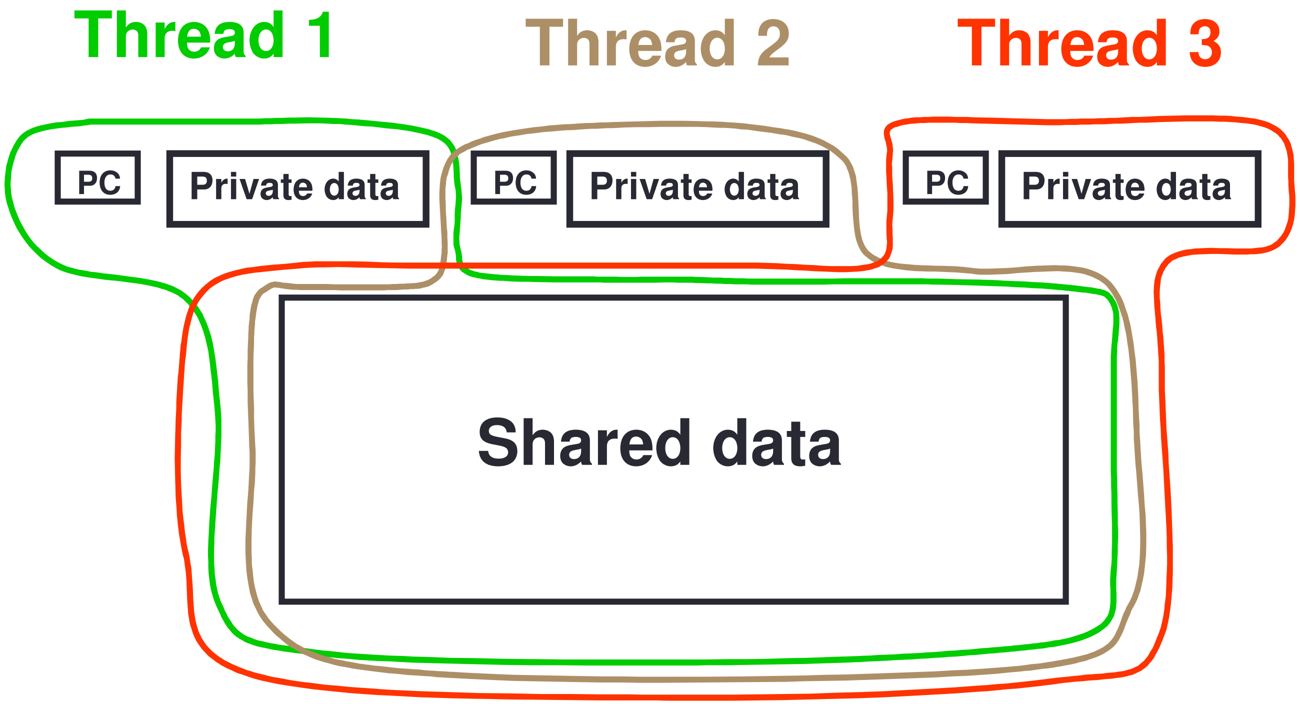
Conceptual model



Real hardware example



Threads (cont.)



Parallel region

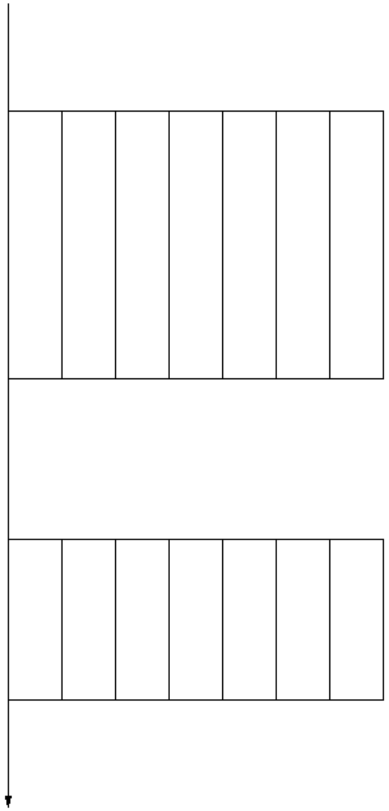
Sequential part

Parallel region

Sequential part

Parallel region

Sequential part



```
      PROGRAM FRED
          .
          .
!$OMP PARALLEL
          .
          .
          .
          .
          .
          .
          .
          .
          .
!$OMP END PARALLEL
          .
          .
          .
          .
          .
!$OMP PARALLEL
          .
          .
          .
          .
          .
!$OMP END PARALLEL
          .
          .
          .
```

Shared and private data

- Inside a parallel region, variables can either be *shared* or *private.*
- All threads see the same copy of shared variables.
- All threads can read or write shared variables.
- Each thread has its own copy of private variables: these are invisible to other threads.
- A private variable can only be read or written by its own thread.

Reductions

- A *reduction* produces a single value from associative operations such as addition, multiplication, max, min, and, or.
- Would like each thread to reduce into a private copy, then reduce all these to give final result.
- Use REDUCTION clause:

Fortran: **`REDUCTION(`***op***`:`** *list***`)`**

C/C++: **`reduction(`***op***`:`** *list***`)`**

- Can have reduction arrays in Fortran, but not in C/C++

Parallel do/for loops (cont)

Syntax:

Fortran:

```
!$OMP DO [clauses]
      do loop
[ !$OMP END DO ]
```

C/C++:

```
#pragma omp for [clauses]
      for loop
```

Parallel do loops (example)

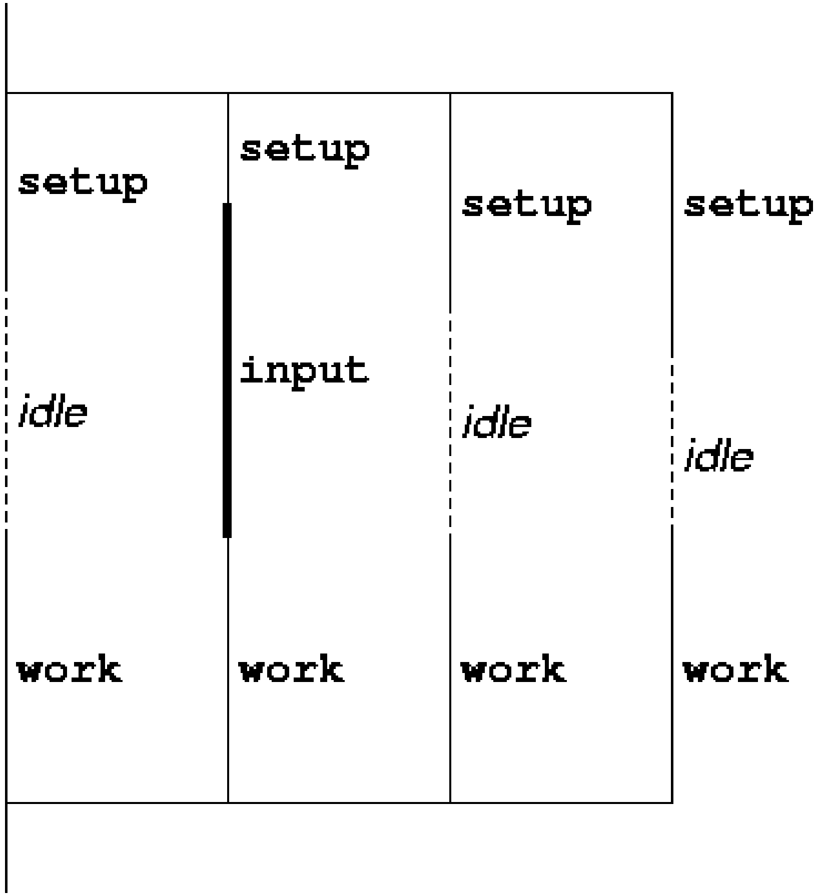
Example:

```
!$OMP PARALLEL
!$OMP DO
      do i=1,n
         b(i) = (a(i)-a(i-1))*0.5
      end do
!$OMP END DO
!$OMP END PARALLEL
```

SINGLE directive (cont)

Example:

```
#pragma omp parallel
{
    setup(x);
#pragma omp single
   {
       input(y);
   }
    work(x,y);
}
```



MASTER directive (cont)

Syntax:

Fortran:

```
!$OMP MASTER
      block
!$OMP END MASTER
```

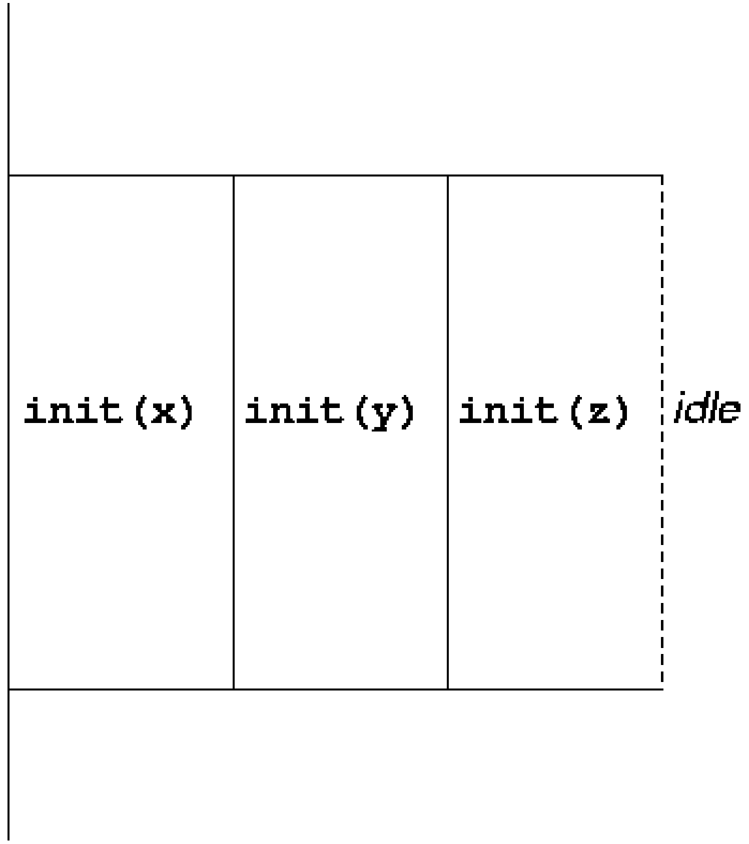
C/C++:

```
#pragma omp master
      structured block
```

Parallel sections (cont)

Example:

```
!$OMP PARALLEL
!$OMP SECTIONS
!$OMP SECTION
      call init(x)
!$OMP SECTION
      call init(y)
!$OMP SECTION
      call init(z)
!$OMP END SECTIONS
!$OMP END PARALLEL
```


init(x)
init(y)
init(z)
idle