Parallel Computing with GPUs

OpenMP
Part 1 - OpenMP Overview

The University Of Sheffield.

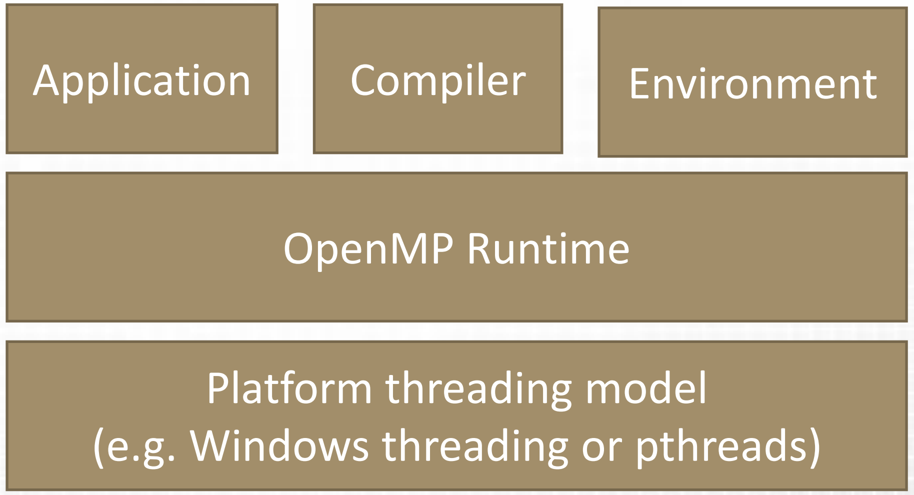
Dr Paul Richmond
http://paulrichmond.shef.ac.uk/teaching/COM4521/

This Lecture (learning objectives)

- ❑Introducing OpenMP
    - ❑Identify the language purpose and approach
- ❑OpenMP “Hello World”
    - ❑Recognise the basic structure of an OpenMP directive
    - ❑Examine output from a parallel application
    - ❑Present the fork and join model

OpenMP

- Open Multi-Processing Standard
  - An API that supports shared memory programming in C, C++ and FORTRAN
  - Cross platform support using native threading
    - Higher level than OS models and portable
  - Is not suitable for distributed computing (look at MPI)
- It is not an automatic parallel programming language
  - Parallelism is explicitly defined and controlled by the programmer
  - Requires compiler directives, a runtime, environment variables



OpenMP Compiler Directives

- Use of `#pragmas`
  - If not understood by the compiler then they are ignored
  - Does not require serial code to be changed
  - Allows behaviour to be specified which are not part of the C standard specification

```
#include <stdio.h>
#include <omp.h>

int main()
{
#pragma omp parallel
    {
        printf("Hello World\n");
    }
    return 0;
}
```

Extending OpenMP Hello World

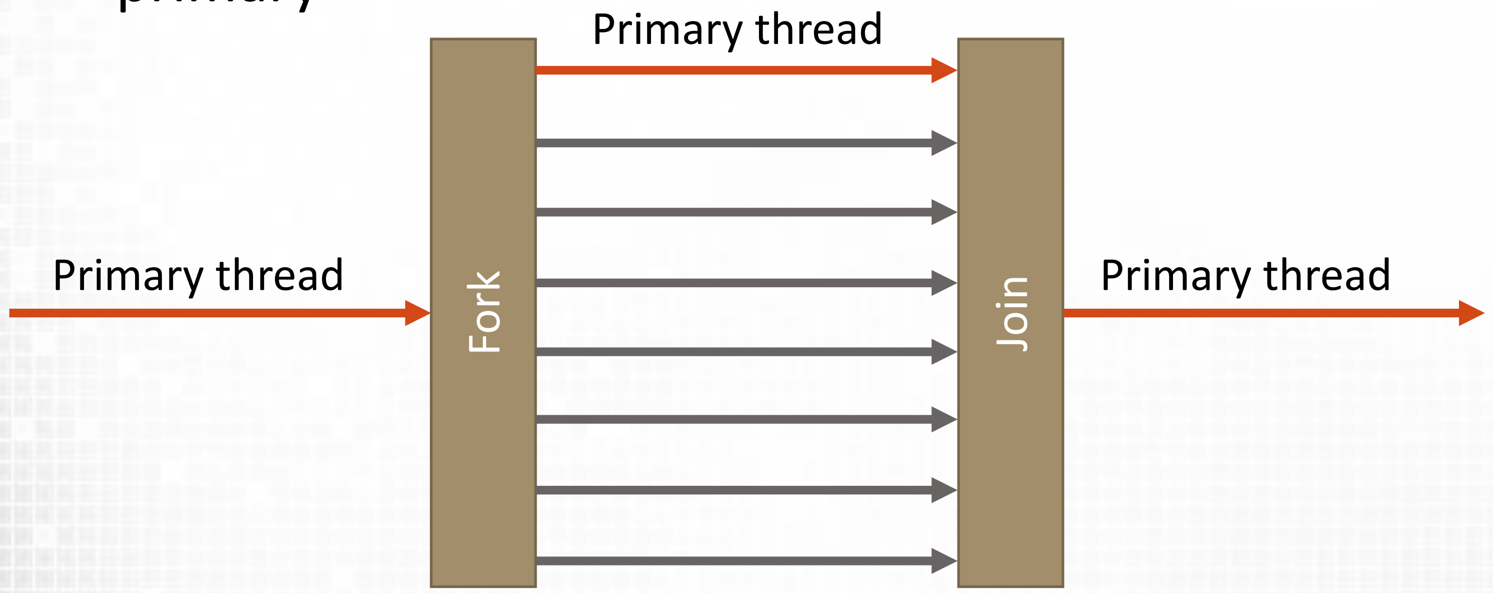
```c
#include <stdio.h>
#include <omp.h>

int main()
{
#pragma omp parallel
    {
        int thread = omp_get_thread_num();
        int max_threads = omp_get_max_threads();
        printf("Hello World (Thread %d of %d)\n", thread, max_threads);
    }
return 0;
}
```

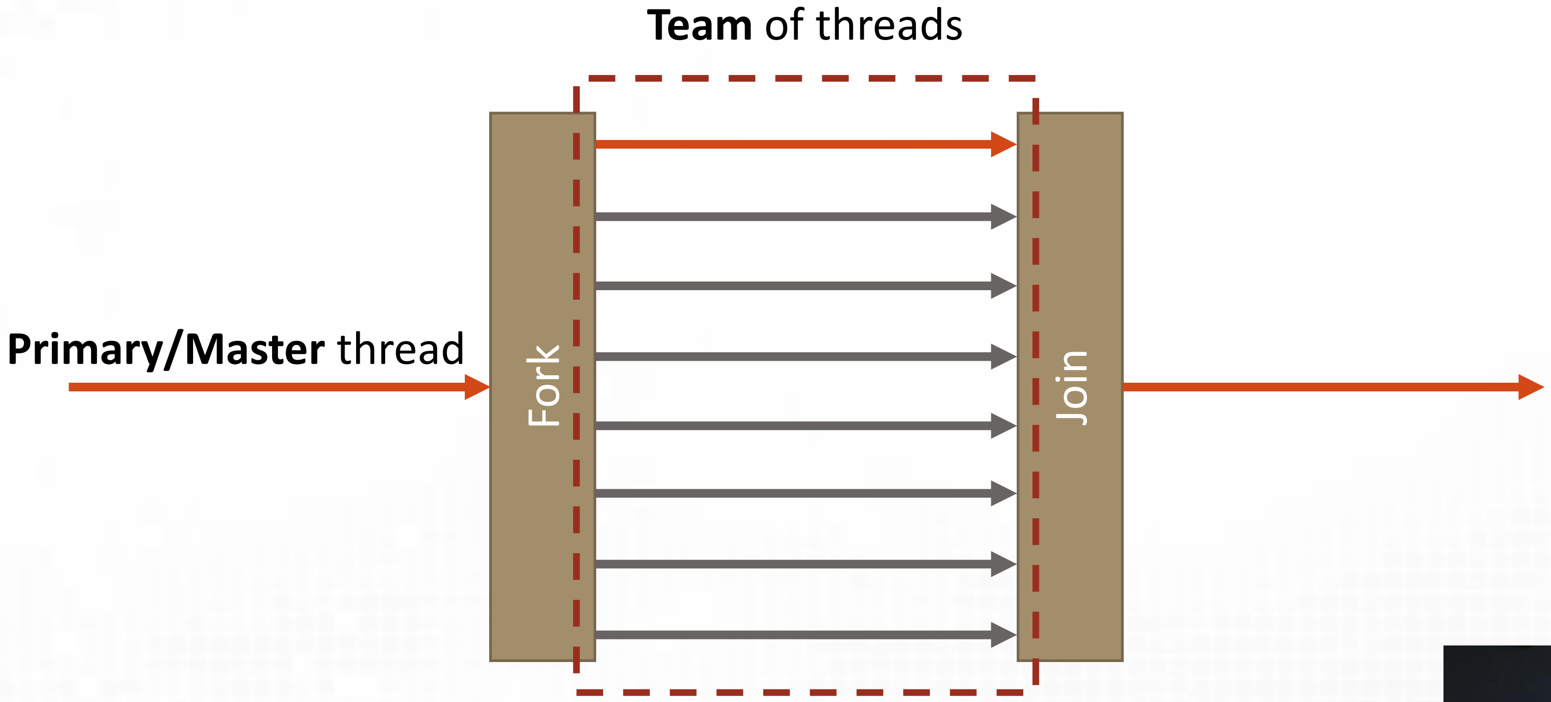
```
Hello World (Thread 5 of 8)
Hello World (Thread 6 of 8)
Hello World (Thread 2 of 8)
Hello World (Thread 7 of 8)
Hello World (Thread 1 of 8)
Hello World (Thread 0 of 8)
Hello World (Thread 3 of 8)
Hello World (Thread 4 of 8)
```

Fork and Join

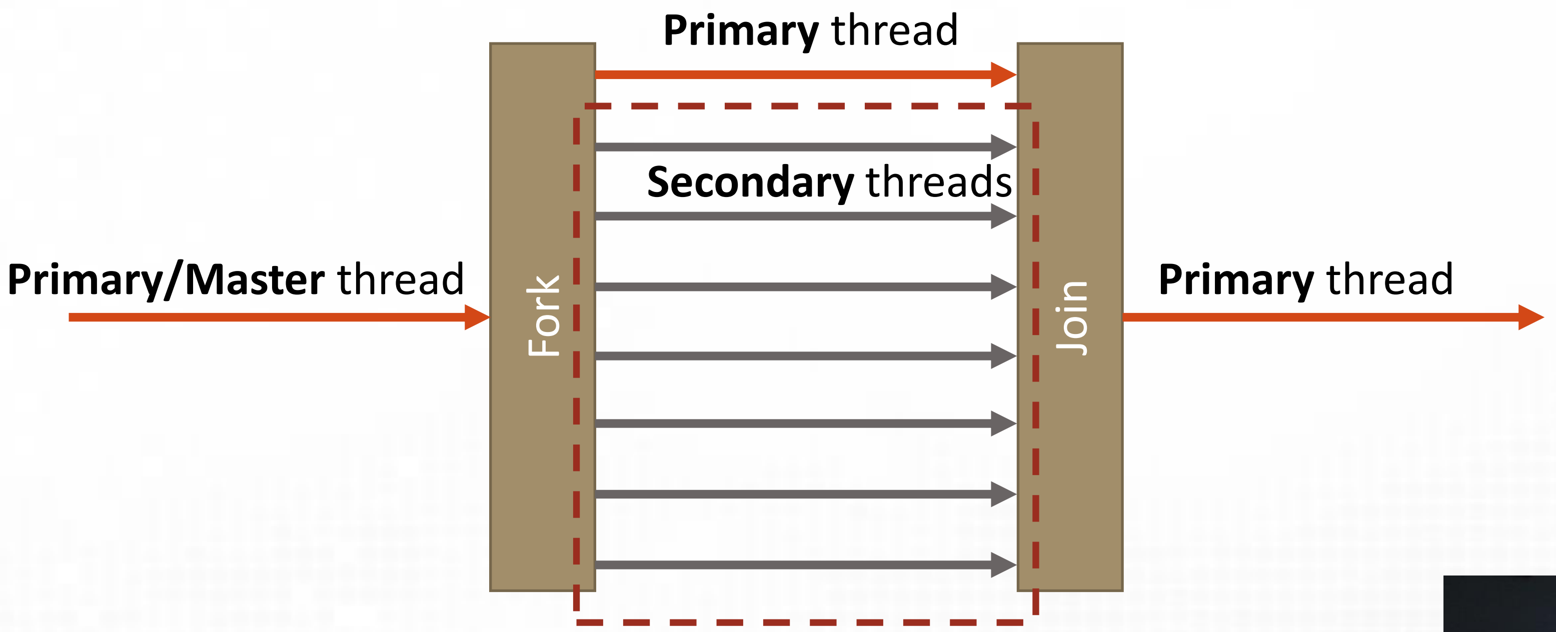
- ❑OpenMP uses a fork a join model
  - ❑Fork: Creates a number of parallel threads from a primary thread
    - ❑Primary thread is always thread 0
    - ❑No guarantee of order
  - ❑Join: Synchronises thread termination and returns program control to primary



Terminology



Terminology



Summary

- ❑Introducing OpenMP
    - ❑Identify the language purpose and approach
- ❑OpenMP “Hello World”
    - ❑Recognise the basic structure of an OpenMP directive
    - ❑Examine output from a parallel application
    - ❑Present the fork and join model

❑Next Lecture: Loops and Critical Sections