PES UNIVERSITY ONLINE

Microprocessor & Computer Architecture (µpCA)

UE19CS252

Dr. D. C. Kiran

Department of
Computer Science and Engineering

Microprocessor & Computer Architecture (μpCA)

Unit 5: Advanced Architecture

Dr. D. C. Kiran

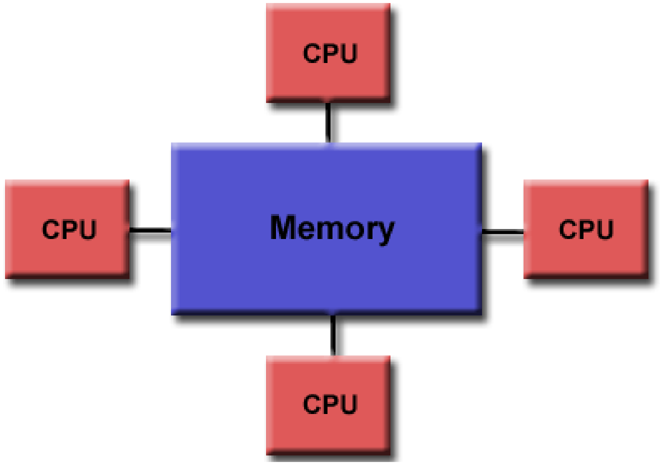
Department of Computer Science and Engineering

Parallel Computer Memory Architectures

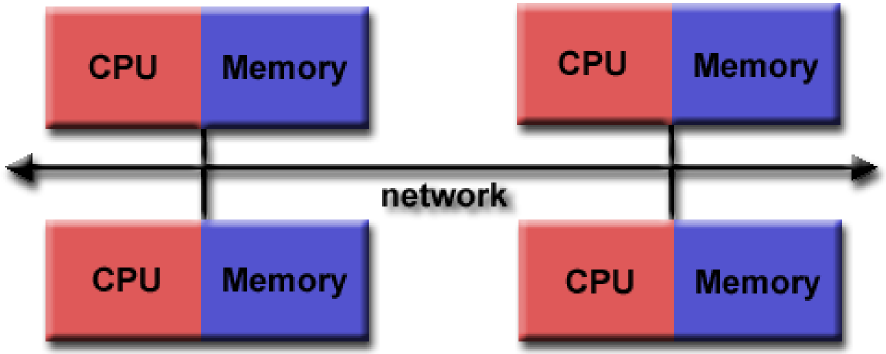
- Shared Memory
- Distributed Memory
- Hybrid Distributed-Shared Memory

Microprocessor & Computer Architecture (μpCA)

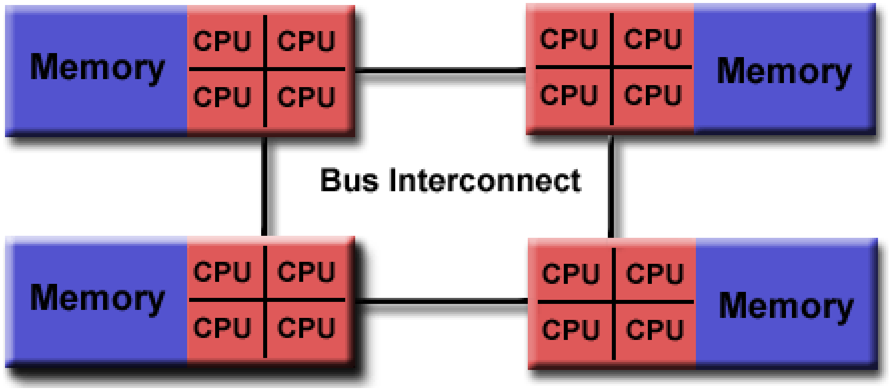
Parallel Computer Memory Architectures



Shared Memory Architecture



Distributed Memory Architecture



Hybrid Architecture

Parallel Programming Languages

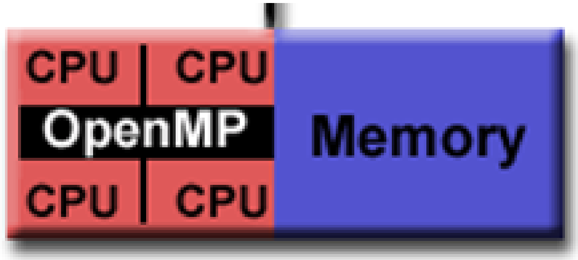
OpenMP: (Open Multi Processing):
API that Support multiprocessing in C, C++, Fortran. Now with Python also.

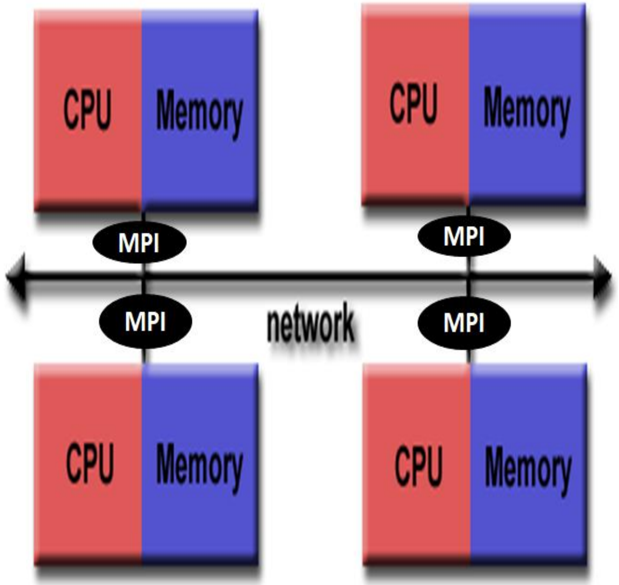
MPI: (Message Passing Interface):
C, C++, Fortran, Java, Python, Ocaml, R……etc

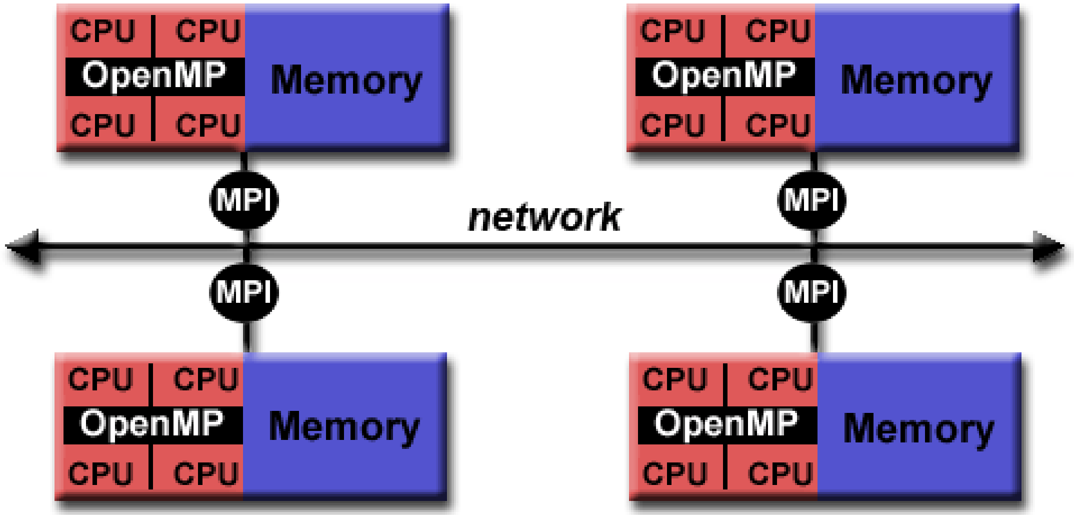
CILK: Customized C Language

CUDA (Computer Unified Devise Architecture): for Nvidia GPU

Pthreads:







Shared Memory Programming

OpenMP Program to print **Hello-PESU** on **4 CPU** with **Shared Memory**

```
#pragma omp parallel default(shared) private(iam, np)
 {
   np = omp_get_num_threads();
   iam = omp_get_thread_num();
   printf("Hello-PESU: from thread %d out of %d ", iam, np);
 }
```

```
Hello-PESU from thread 0 out of 4
Hello-PESU  from thread 2 out of 4
Hello-PESU  from thread 1 out of 4
Hello-PESU  from thread 3 out of 4
Hello-PESU from thread 0 out of 4
Hello-PESU  from thread 2 out of 4
Hello-PESU  from thread 1 out of 4
Hello-PESU  from thread 3 out of 4
```

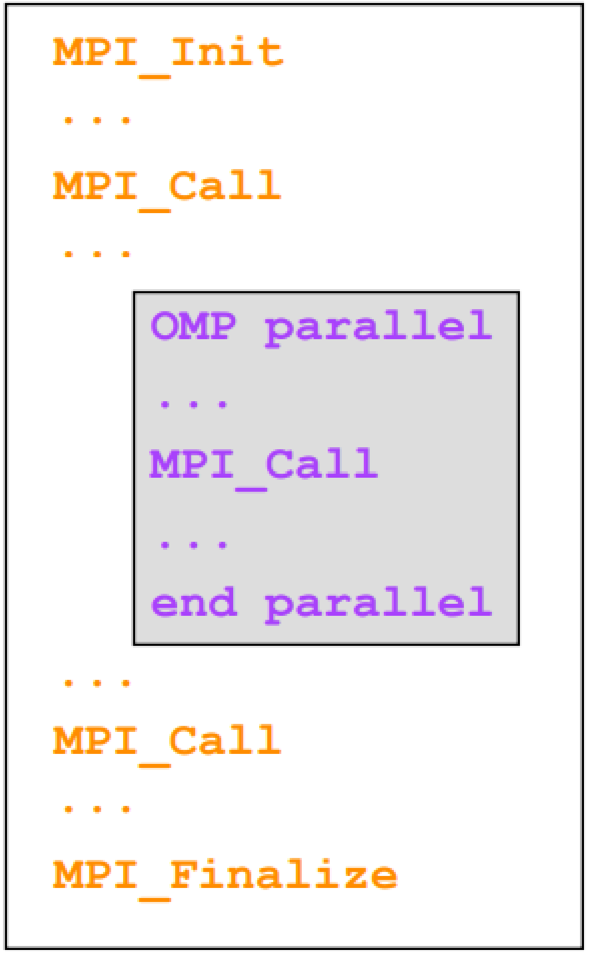
Distributed Memory Programming

MPI Program to print **Hello-PESU** on **2 CPU** with **Distributed Memory**

```
MPI_Init(&argc, &argv);
 MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
printf("Hello-PESU from rank %d out of %d processors\n",rank, numprocs);
```

```
Hello-PESU from rank 0 out of 2
Hello-PESU  from rank 1 out of 2
```

Hybrid Programming



```
MPI_Init
...
MPI_Call
...
    OMP parallel
    ...
    MPI_Call
    ...
    end parallel
...
MPI_Call
...
MPI_Finalize
```

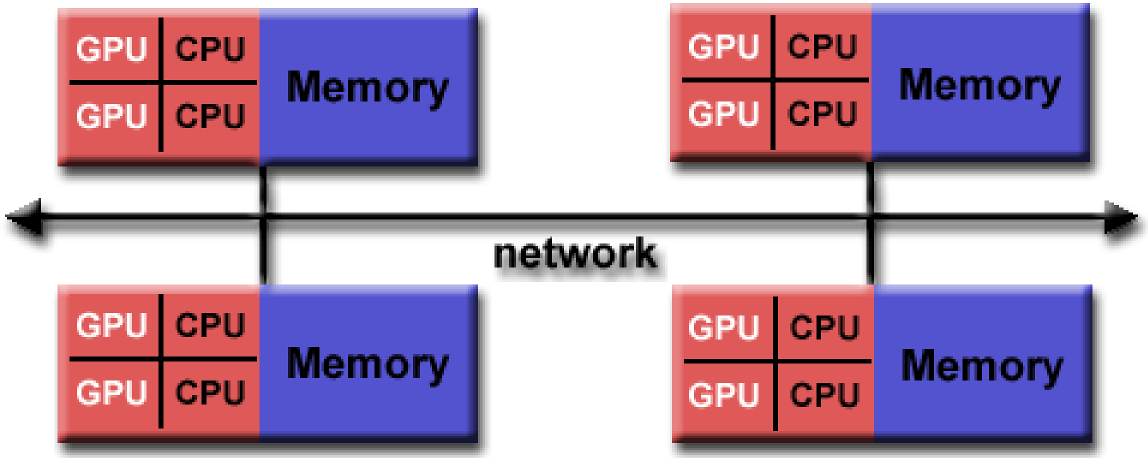
Hybrid Programming

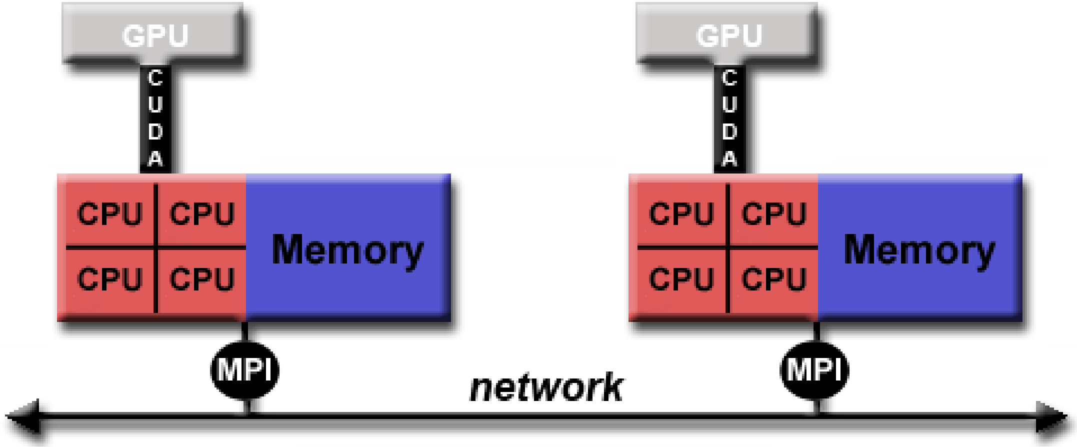
Hybrid (MPI + OpenMP) Program to print **Hello-PESU** on **with 4 core and 2 CPU**

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

MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);

#pragma omp parallel default(shared) private(iam, np)
 {
   np = omp_get_num_threads();
   iam = omp_get_thread_num();
   printf("Hello-PESU: from %d Thread out of %d Threads  & %d Rank of %d Processors", iam, np,rank,numprocs);
 }
```

```
Hello-PESU from  0 Thread out of 4 Threads & 0 Rank of 2 Processors
Hello-PESU  from  2 Thread out of 4 Threads & 1 Rank of 2 Processors
Hello-PESU  from  1 Thread out of 4 Threads & 0 Rank of 2 Processors
Hello-PESU  from 3 Thread out of 4 Threads & 1 Rank of 2 Processors
Hello-PESU from 0 Thread out of 4 Threads & 0 Rank of 2 Processors
Hello-PESU  from 2 Thread out of 4 Threads & 1 Rank of 2 Processors
Hello-PESU  from 1 Thread out of 4 Threads & 0 Rank of 2 Processors
Hello-PESU  from 3 Thread out of 4  Threads & 1 Rank of 2 Processors
```

With Graphical Processing Unit





PES UNIVERSITY ONLINE

THANK YOU

Dr. D. C. Kiran

Department of Computer Science and Engineering

dckiran@pes.edu

9829935135