

# Outline of Assignment 4, Part I

## Deadlock-free communication

pmd.c

```
if even
    MPI_Send()
    MPI_Recv()
else if odd
    MPI_Recv()
    MPI_Send()
else (self)
    memory copy
```

No ifs & buts!  
cleaner

pmd\_irecv.c

```
MPI_Irecv()
MPI_Send() // OK to send to self
MPI_Wait()
```

## Where?

```
2 in atom_copy()
2 in atom_move()
```

4 code segments in total

## Computation (ns)/communication ( $\mu$ s-ms) overlap

```
MPI_Irecv()
?  
MPI_Send()  
MPI_Wait()
```

[discovery ~]\$ ping hpc-transfer.usc.edu  
time=0.074 ms

...

# Bash Programming

pmd\_irecv.sl

```
mpicc -O -o pmd_irecv pmd_irecv.c -lm
```

```
counter=0
```

Value of a variable

```
while [ $counter -lt 3 ]; do
```

```
  echo "***** Asynchronous *****"
```

Print to terminal

```
  mpirun -n $SLURM_NTASKS ./pmd_irecv
```

```
  echo "***** Synchronous *****"
```

```
  mpirun -n $SLURM_NTASKS ./pmd
```

```
  let counter+=1
```

Evaluate a mathematical expression & stores its result into a variable

```
done
```

See “Bash scripting tutorial for beginners”

<https://linuxconfig.org/bash-scripting-tutorial-for-beginners>

# Runtime Fluctuation

- Due to (1) network interference & (2) shared access to computing nodes, measured runtimes will fluctuate
- The latter could be avoided by exclusive access (`#SBATCH --exclusive`), but **please do not use this** since it will cause very low utilization of computing resources & slow down other users' work

```
***** Asynchronous *****
CPU & COMT = 5.190056e-01 1.406116e-01
***** Synchronous *****
CPU & COMT = 5.163595e-01 1.690525e-01

***** Asynchronous *****
CPU & COMT = 5.119587e-01 1.358150e-01
***** Synchronous *****
CPU & COMT = 5.163731e-01 1.707310e-01

***** Asynchronous *****
CPU & COMT = 5.117363e-01 1.349801e-01
***** Synchronous *****
CPU & COMT = 5.155800e-01 1.672257e-01
```

*pmd\_irecv.c*

*pmd.c*

# Resource Usage (1)

- Start interactive job on discovery & start a MPI program on one of the allocated computing nodes

```
[anakano@discovery cs596]$ salloc --nodes=4 --ntasks-per-node=4 -t 30
salloc: Nodes d05-[33-36] are ready for job
[anakano@d05-33 cs596]$ mpirun -n 16 ./pmd_irecv
...
```

- In another terminal, log in to another allocated node & type 'top' to see running processes

```
[anakano@discovery cs596]$ ssh d05-34
[anakano@d05-34 ~]$ top
top - 07:42:03 up 47 days, 18:34, 2 users, load average: 4.37, 3.33, 3.15
Tasks: 315 total, 8 running, 307 sleeping, 0 stopped, 0 zombie
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3262	rvandamm	20	0	1168000	1.0g	25228	R	100.0	0.5	1090:38	rna_denovo.stat
3263	rvandamm	20	0	1344840	1.2g	25228	R	99.7	0.6	1090:38	rna_denovo.stat
23608	anakano	20	0	432324	110840	8660	R	99.7	0.1	0:26.48	pmd_irecv
23609	anakano	20	0	432332	108808	8672	R	99.7	0.1	0:26.41	pmd_irecv
23610	anakano	20	0	432324	110856	8676	R	99.7	0.1	0:26.51	pmd_irecv
23607	anakano	20	0	432328	108732	8604	R	99.3	0.1	0:26.43	pmd_irecv
15225	sgopalan	20	0	11.4g	11.2g	7576	R	99.0	5.9	2072:59	R
19675	telegraf	20	0	1507240	49764	18380	S	0.3	0.0	14:08.81	telegraf
23588	anakano	20	0	164372	2508	1612	R	0.3	0.0	0:00.12	top
1	root	20	0	43572	3956	2528	S	0.0	0.0	2:02.10	systemd

4 instances (ranks) of pmd\_irecv are running per node

# Resource Usage (2)

- Type '1' (toggle to show detailed core usage): two users (including myself) are not making full use of cores; let others utilize the unused resources by avoiding exclusive access

```
%Cpu0 : 0.0 us, 0.0 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.3 si, 0.0 st
%Cpu1 :100.0 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu2 :100.0 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu3 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu4 : 76.3 us, 23.0 sy, 0.0 ni, 0.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu5 :100.0 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu6 :100.0 us, 0.0 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu7 : 99.7 us, 0.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu8 : 99.7 us, 0.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu9 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu10 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu11 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu12 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu13 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu14 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu15 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu16 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu17 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu18 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu19 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu20 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu21 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu22 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
%Cpu23 : 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
```

17 out of 24 cores unused

# Note on Assignment 4, Part II

---

- Hands-on experience in a common situation of adding new analysis functionality to an existing MPI simulation code, *via* minimally invasive surgery of the code
- Note the header, `pmd.h`, in the homework package, `csci596-as04`, was set for **Part I**:

```
int vproc[3] = {2,2,4}, nproc = 16;
```

The number of MPI ranks should match `nproc` in `pmd.h`:

```
mpirun -n 16 ./pmd (also ./pmd_irecv)
```

- Due to ‘shadow’ analysis ranks, the total number of ranks to be spawned by `mpirun` in **Part II** should instead be twice the number of spatial subsystems, `nproc`, in `pmd_split.c`:

In `pmd_split.h`:

```
int vproc[3] = {2,2,2}, nproc = 8;
```

Run:

```
#SBATCH --nodes=2
```

```
#SBATCH --ntasks-per-node=8
```

```
mpirun -n $SLURM_NTASKS ./pmd_split // $SLURM_NTASKS = 16
```

# Message Composition

---

- **Multidimensional arrays are sent as one-dimensional arrays**

```
double rv[NMAX][3];  
double dbuf[NDBUF];
```

