MPI

Contents

[Allgemein 1](#_Toc67945869)

[MPI Process Ranks 1](#_Toc67945870)

[Message Exchange 2](#_Toc67945871)

[MPI\_Send 2](#_Toc67945872)

[MPI Send Operation Modes 3](#_Toc67945873)

[MPI\_Recv 3](#_Toc67945874)

[Non-blocking Operations 4](#_Toc67945875)

[Global Collective Operations 5](#_Toc67945876)

[MPI\_Bcast 5](#_Toc67945877)

[MPI\_Scatter 6](#_Toc67945878)

[MPI\_Gather 6](#_Toc67945879)

[MPI\_Allgather 7](#_Toc67945880)

[MPI\_Alltoall 8](#_Toc67945881)

[MPI\_Reduce 8](#_Toc67945882)

[Vektor-Varianten 9](#_Toc67945883)

[Synchronization 10](#_Toc67945884)

[MPI\_Barrier 10](#_Toc67945885)

[MPI\_Test und MPI\_Wait 10](#_Toc67945886)

[Aufgaben 10](#_Toc67945887)

[AllToAll 10](#_Toc67945888)

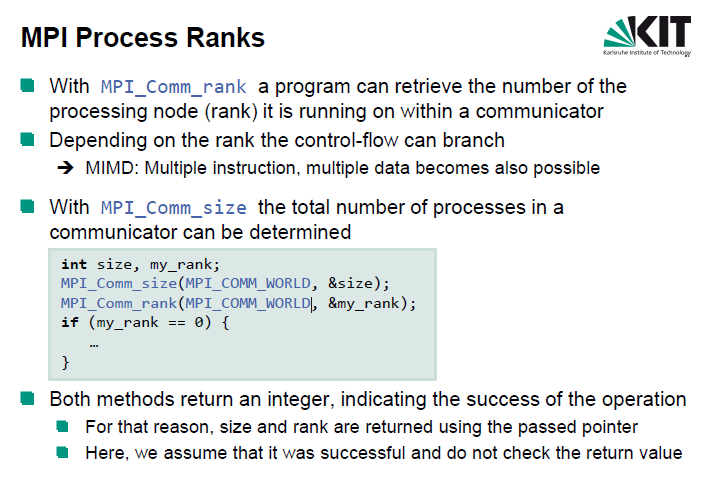
[Implementieren 10](#_Toc67945889)

[Tabelle füllen 11](#_Toc67945890)

# Allgemein

## MPI Process Ranks

root ist immer 0



# Message Exchange

## MPI\_Send

**buf :** initial address of send buffer (choice)

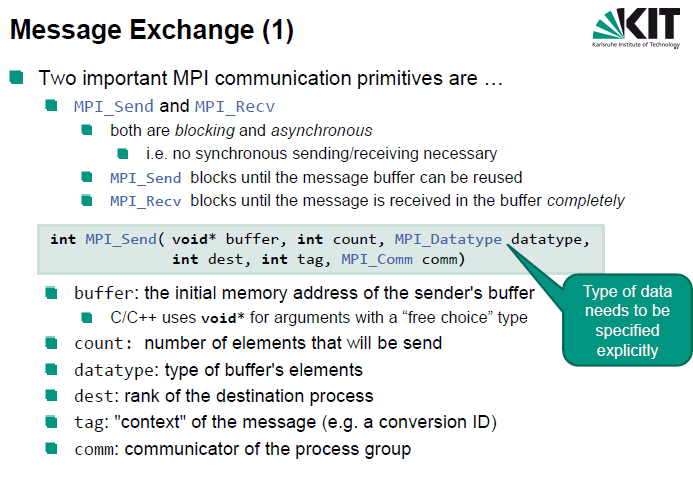
**count :** number of elements in send buffer (nonnegative integer)

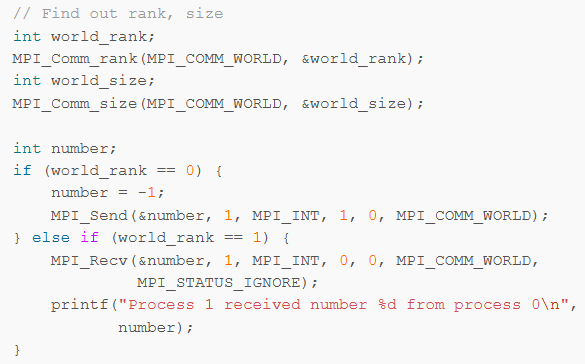
**datatype :** datatype of each send buffer element (handle)

**dest :** rank of destination (integer)

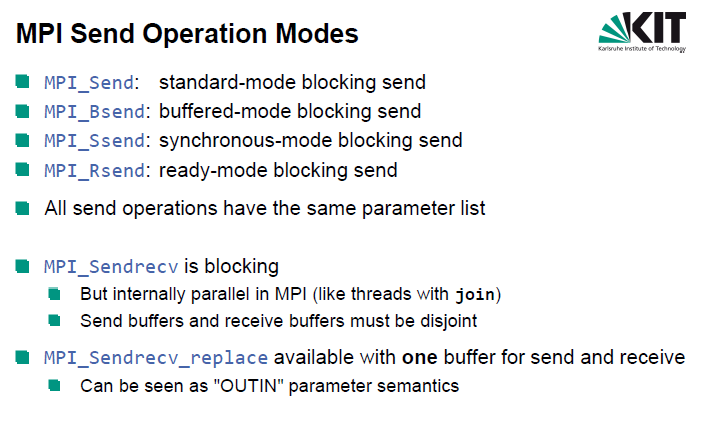
**tag :** message tag (integer)

Tag braucht man in der Prüfung nie (einfach 0 schreiben)

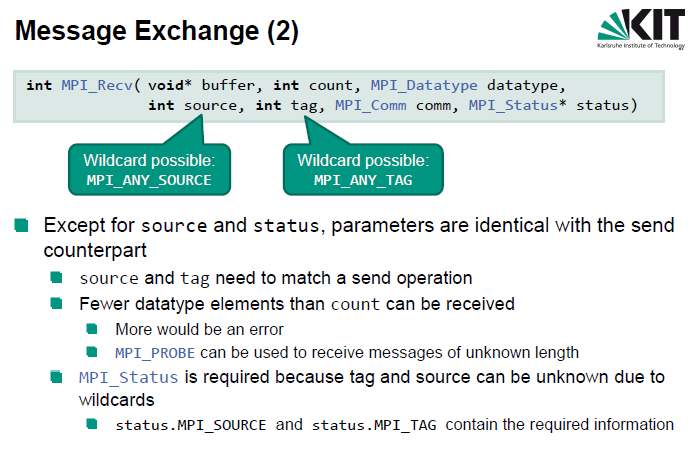




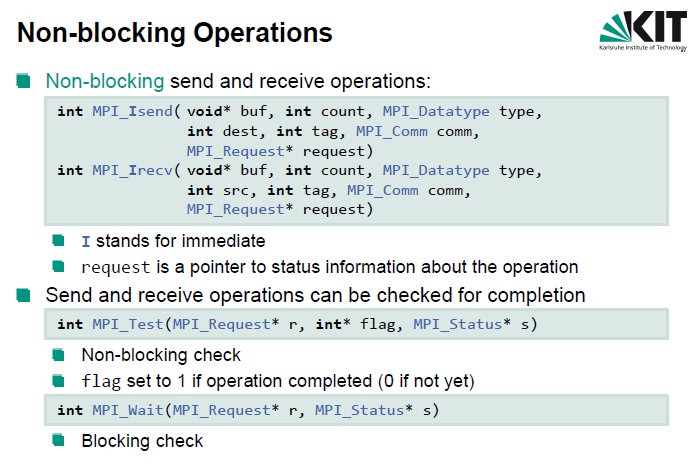
### MPI Send Operation Modes

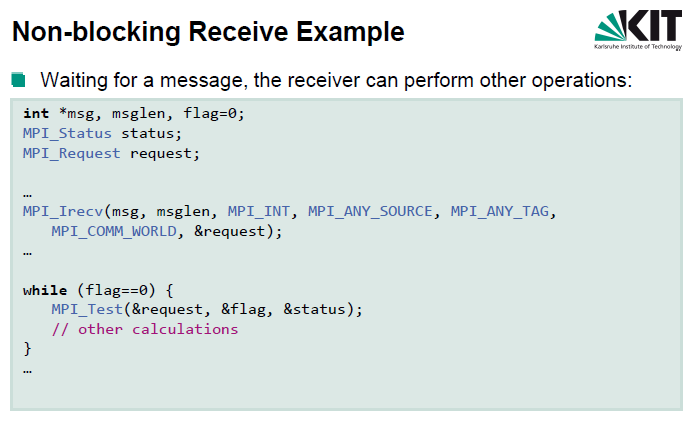


## MPI\_Recv

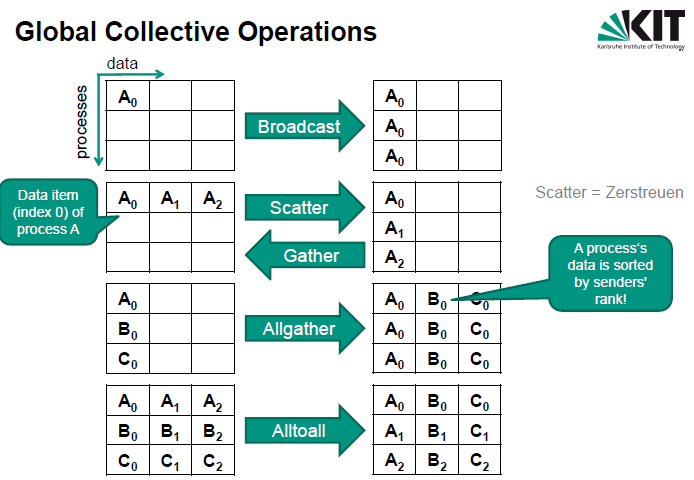


## Non-blocking Operations





# Global Collective Operations



## MPI\_Bcast

Root send his data from buffer to all processes

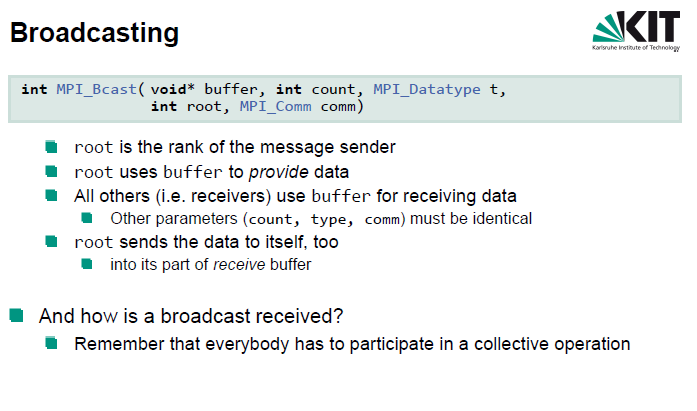
MPI\_Comm comm;

int array[100];

int root=0;

...

MPI\_Bcast( array, 100, MPI\_INT, root, comm);



## MPI\_Scatter

“Streue” Daten von root an alle Prozesse

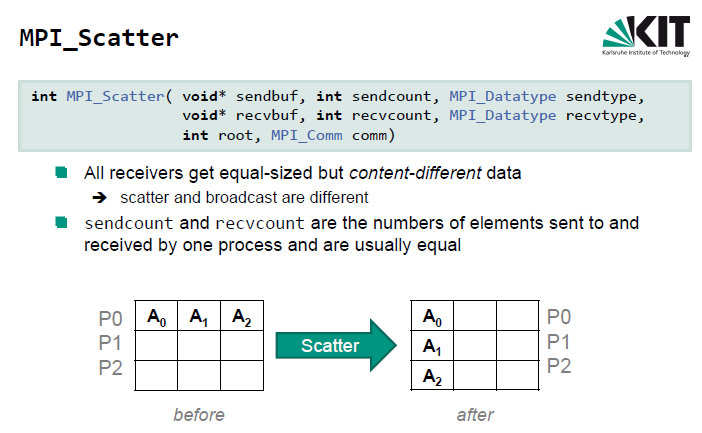
Root ist hier Rang vom Sender, muss nicht 0 sein

**sendcount:** number of elements sent to each process (integer, significant only at root)

**recvcount:**number of elements in receive buffer (integer)

**root:** rank of sending process (integer)

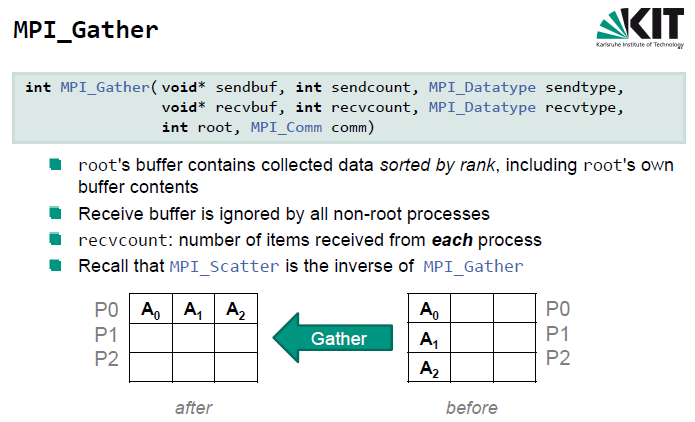
**Es werden bei Scatter alle Werte aus sendbuf geschickt**

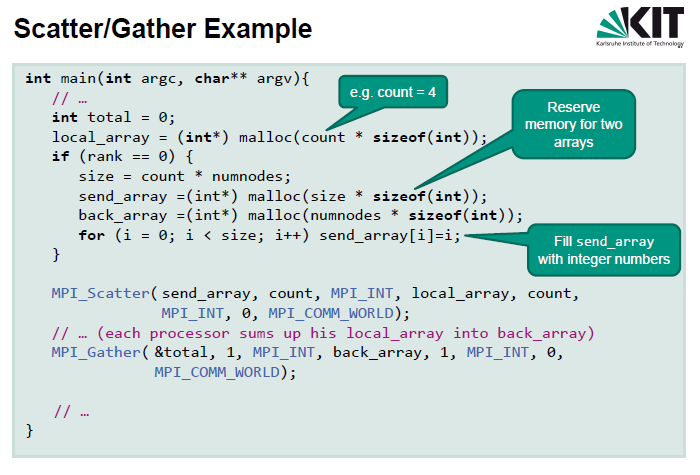


## MPI\_Gather

Sammele Daten von allen Prozessen bei Root

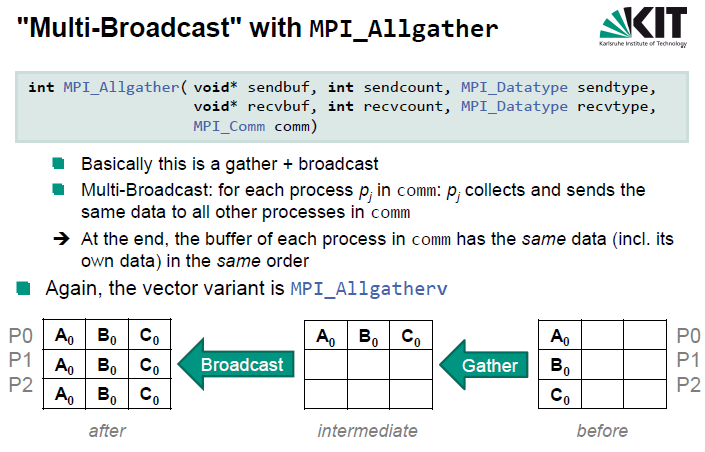
sendcount, revcount, root - wie bei Scatter





## MPI\_Allgather

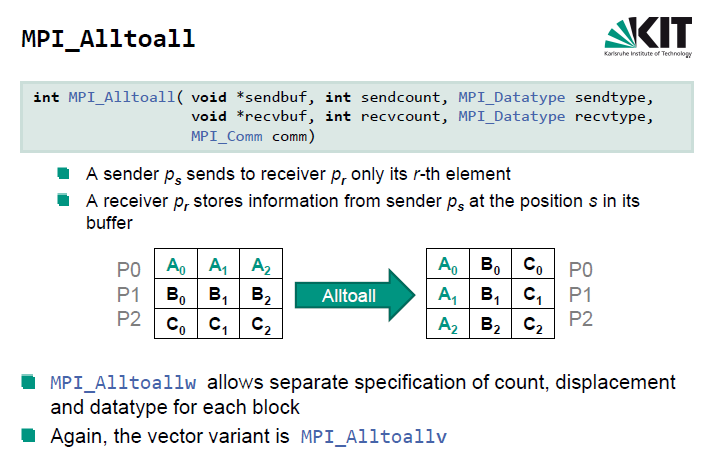
Daten sammeln mit Gather, dann die Kopien der „ganzen“ Daten verteilen



## MPI\_Alltoall

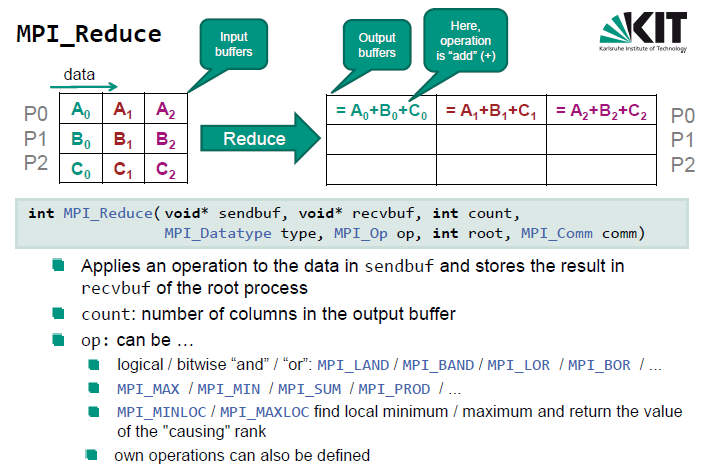
Prozess mit Rang i bekommt alle i-te Datenpakete

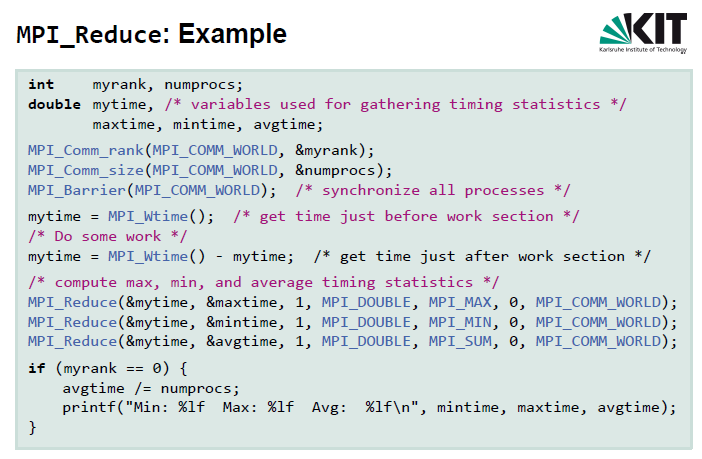
Kann als „Shuffle“ dienen



## MPI\_Reduce

Root bekommt die Daten von allen Prozessen + Reduce mit einer math. Funktion (wie bei fold)





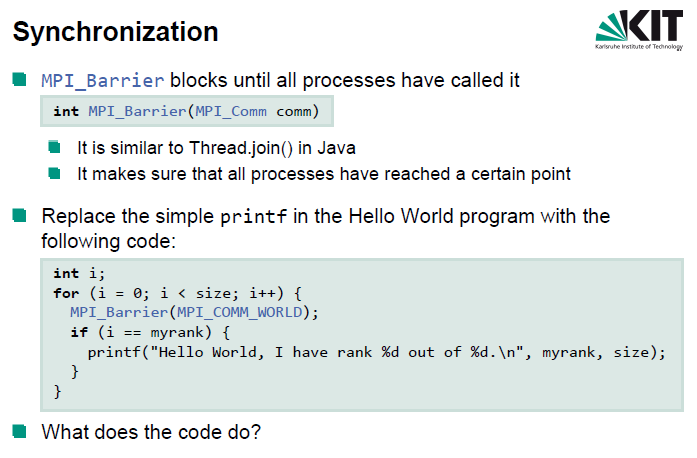
## Vektor-Varianten

Alle MPI-Befehle haben auch eine Vektor-Variante, die mit \*v endet:

MPI\_Gatherv, MPI\_Scatterv, MPI\_Allgatherv...

# Synchronization

## MPI\_Barrier

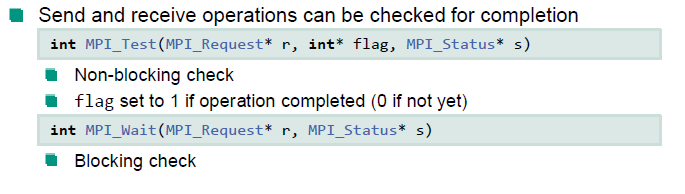


## MPI\_Test und MPI\_Wait

MPI\_Wait Waits for an MPI request to complete

request: [in] request (handle)

status: [out] status object (Status). May be MPI\_STATUS\_IGNORE.



# Aufgaben

## AllToAll

### Implementieren

for (int sender = 0; sender < size; sender++) {

MPI\_Scatter(sendBuffer, 1, MPI\_INT, recvBuffer + sender, 1,

MPI\_INT, sender, MPI\_COMM\_WORLD);

}

### Tabelle füllen

