Figure 6.2  Blocking buffered transfer protocols: (a) in the presence of communication hardware with buffers at send and receive ends; and (b) in the absence of communication hardware, sender interrupts receiver and deposits data in buffer at receiver end.
Non-Buffered

Blocking Operations
- Sending process returns after data has been copied into communication buffer

Non-Blocking Operations
- Sending process returns after initiating DMA transfer to buffer. This operation may not be completed on return

Buffered

Non-Buffered

Send and Receive semantics assured by corresponding operation

Programmer must explicitly ensure semantics by polling to verify completion

Figure 6.3 Space of possible protocols for send and receive operations.
Figure 6.4  Non-blocking non-buffered send and receive operations (a) in absence of communication hardware; (b) in presence of communication hardware.
Figure 6.5  Different ways to map a set of processes to a two-dimensional grid. (a) and (b) show a row- and column-wise mapping of these processes, (c) shows a mapping that follows a space-filling curve (dotted line), and (d) shows a mapping in which neighboring processes are directly connected in a hypercube.
MinLoc(Value, Process) = (11, 2)
MaxLoc(Value, Process) = (17, 1)

Figure 6.6  An example use of the MPI_MINLOC and MPI_MAXLOC operators.
Figure 6.7 Using MPI_Comm_split to split a group of processes in a communicator into subgroups.
Figure 6.8 Splitting a Cartesian topology of size $2 \times 4 \times 7$ into (a) four subgroups of size $2 \times 1 \times 7$, and (b) eight subgroups of size $1 \times 1 \times 7$. 

keep_dims[] = (false, false, true)

keep_dims[] = (false, false, true)