MPI — Message Passing Interface

- File I/O: (VE-V2-Chapter 10-268)
- Note that these are binary files.
- Acts like a traditional file handle, in that there are open, close, read/write, and seek operations on it. Unlike traditional file handling, which in parallel would mean having one handle per process, this handle is collective: MPI processes act as if they share one file handle.
  Although this is a single file handle shared between the processes, each process holds an individual file pointer that it can individually position somewhere in the shared file.
  `MPI_File` type, `MPI_File_open`, access modes, `MPI_File_close`, (VE-V2-269)
- One convenient set of commands are `MPI_File_read_at` and `MPI_File_write_at` that combine seeks and reads or writes.
  Example: Files001
  Example: Files002
  Example: Files003

MPI — Message Passing Interface

- MPI also offers `MPI_File_seek`, `MPI_File_read` and `MPI_File_write` that follow traditional file handling. (VE-V2-270–271)
  Example: Files004
- With this mode it helps to use the file like you would a collective communication between nodes and an array on rank 0, that is, a "shared" location on the control node. Since the file is shared between the processes, we are back to race conditions.
- If all processes execute a read at the same logical time, it is better to use the collective call `MPI_File_read_all`. This is "equivalent" to reading the file in one process and broadcasting it to all the other processes.
  Example: Files005
- There is also a collective write `MPI_File_write_all` that comes in handy when all process are to write at the same time.
  Example: Files006
- There are also non-blocking counterparts to these. As usual, these include an 'i' in the name and return an `MPI_Request` that will correspond to an `MPI_Wait` command for process synchronization.

Reference Roadmap

The slide outlines contain references to the main course materials. Not everything has a reference but nearly all the materials can be found in the following references. The references are of the form `(<text>-<pages>)` so (PM-123) means page 123 of An Introduction to Parallel Programming by Peter S. Pacheco and Matthew Malensek.

- (VE-V1) — The Science of Computing: The Art of High Performance Computing, Vol 1 by Victor Eijkhout
- (VE-V2) — Parallel Programming in MPI and OpenMP: The Art of HPC, Vol 2 by Victor Eijkhout
- (VE-V3) — Introduction to Scientific Programming in C++17/Fortran2008: The Art of HPC, Vol 3 by Victor Eijkhout
- (VE-V4) — Tutorials for High Performance Scientific Computing: The Art of HPC, Vol 4 by Victor Eijkhout
- (PM) — An Introduction to Parallel Programming by Peter S. Pacheco and Matthew Malensek.
- (KH) — Programming Massively Parallel Processors: A Hands-on Approach by David B. Kirk and Wen-mei W. Hwu.
- (SAB) — High Performance Computing Modern Systems and Practices by Thomas Sterling, Matthew Anderson, and Maciej Brodowicz.
- (GLS) — Using MPI: Portable Parallel Programming with the Message-Passing Interface by William Gropp, Ewing Lusk, and Anthony Skjellum.
- (GHTL) — Using Advanced MPI: Modern Features of the Message-Passing Interface by William Gropp, Torsten Hoefler, Rajeev Thakur, and Ewing Lusk.