Proposed CCSM API

Proposed CCSM API:
General concepts:

- interfaces for inquire/put_attribute/get_attribute/etc will be supported but are not discussed here.
- pio will have some default (machine dependent?) io decomp/read/write settings, users can change the defaults via calls into pio
- users will have to declare pio datatypes, but not know anything about them. as a result, some memory will be allocated at the caller level, but other memory will live only in pio.
- several pio datatypes will be used by caller. hierarchical var->decomp->grid->dim and also a filetype. maybe we should merge grid and decomp to "grid".
- pio_put_data and pio_get_data will be overloaded for integer/real/real*8 and 0-7 dim arrays. this will then be passed to the pio "work" subroutines as a 1d array.
- can handle varied axis ordering (ie. nx,nz,ny) a few ways; global index decomp info can handle this automatically if done properly but will require pio to handle non-monotonically increasing decomp info. alternatively, we could provide a integer array to the grid type to support reordering (mapping) of dimensions between the grid type and var data. there are probably other ways to do it. we'll prototype.
- could overload the pio_def_decomp call to support global_indexing (aka DOF), segmented initialization (ala gsmap start/length arrays), ndim blocked chunking defs (ala netcdf start/count arrays), etc.
- (implementation proposal) inside pio, there will be a array of datatypes containing things like rearrangers, io_decomps, etc. these will be reusable for any var with the same decomp and filehandle. we probably don't want them to be attached to any particular file, var, or decomp but exist separately to maximize reuse.

Proposed Interfaces:

- subroutine pio_open(filetype, filename, comm, iotype, num_iotasks, ...)
  - type(pio_file_type),(inout) :: filetype
  - char* ,(in) :: filename
  - integer ,(in) :: mpicom
  - integer ,(in) :: iotype ! pnetcdf, snetcdf, bin, etc
  - integer, optional ,(in) :: num_iotasks
  - integer, optional ,(in) :: IO_gi(🙂 ! formerly IODOF
  - ? , optional ,(in) :: ?
- subroutine pio_def_dim(filetype, dimtype, name, length)
  - type(pio_file_type),(in) :: filetype
  - type(pio_dim_type) ,(inout) :: dimtype
  - char* ,(in) :: name
  - integer ,(in) :: length
- subroutine pio_def_grid(gridtype, name, Ndims, dimids)
  - type(pio_grid_type),(inout) :: gridtype
  - char* ,(in) :: name
  - integer ,(in) :: Ndims
  - type(pio_dim_type) ,(in) :: dimids(Ndims)
- subroutine pio_def_decomp(decomptype, name, gridtype, globalindex)
  - type(pio_decomp_type),(inout):: decomptype
  - type(pio_grid_type) ,(in) :: gridtype
  - char* ,(in) :: name
  - integer ,(in) :: globalindex(🙂 ! was COMP_DOF
- subroutine pio_def_var(filetype, vartype, name, decomptype)
  - type(pio_file_type) ,(in) :: filetype
  - type(pio_var_type) ,(inout):: vartype
  - char* ,(in) :: name
  - type(pio_decomp_type),(in) :: decomptype
- subroutine pio_put_data(filetype, vartype, data)
  - type(pio_file_type) ,(in) :: filetype
  - type(pio_var_type) ,(in) :: vartype
  - real/integer ,(in) :: data(:,:,:,:,...) ! overloaded interface
- subroutine pio_def_fileinfo(num_iotasks, ...)
  - integer, optional ,(in) :: num_iotasks
  - integer, optional ,(in) :: IO_gi(🙂 ! formerly IODOF
  - ? , optional ,(in) :: ?

Proposed Datatypes:

- type pio_dim_type
  - char* :: name
  - integer :: len
- type pio_grid_type
  - char* :: name
  - integer :: Ndims
  - type(pio_dim_type),pointer :: dims(🙂
- type pio_decomp_type
  - char :: name
  - type(pio_grid_type),pointer :: grid
  - type❓ :: map ! gsmap or global_index array or something
- type pio_var_type
  - char :: name
  - type(pio_decomp_type), pointer :: decomp
- type pio_file_type

    - char :: filename
    - integer(i4) :: fh ! The file handle
    - integer(i4) :: IO_comm ! The IO communicator
    - integer(i4) :: comp_comm ! The Compute communicator
    - integer(i4) :: num_iotasks ! total number of IO tasks
    - integer(i4) :: stride_iotasks ! stride between IO tasks
    - integer(i4) :: comp_rank ! the computational rank
    - integer(i4) :: io_rank ! the io rank if io_rank = -1 not an IO processor
    - integer(i4) :: iotype ! Type of IO to perform see parameter statement below
    - integer(i4) :: Info ! MPI-IO info structure
    - integer(kind=PIO_OFFSET) :: offset ! offset into file
    - logical(log_kind) :: IOproc ! .true. if an IO processor
- type pio_desc_type
    - logical(log_kind) :: UseRearranger ! .true. if data rearrangement is necessary
    - integer(i4) :: glen ! global length of array in words
    - integer(i4) :: RfileTYPE, WfileTYPE &! MPI data types for file
    - integer(i4) :: RelemTYPE, WelemType ! MPI data type for read/write operations
    - integer(i4) :: n_RelemTYPE, n_WelemTYPE ! number of Read/Write elem types to read/write
    - integer(i4) :: n_Rwords,n_Wwords ! number of words to read/write
    - type(pio_decomp_type),pointer:: IOmap ! IO decomposition map
    - type(pio_decomp_type),pointer:: COMPmap ! Computational decomposition map
    - integer(i4) :: lsize_comp ! local size of GSMap for comp layout
    - integer(i4) :: lsize_io ! local size of GSMap for IO layout
    - type(Rearranger?) :: rearr_comp_to_io ! mct rearranger comp->io
    - type(Rearranger?) :: rearr_io_to_comp ! mct rearranger io->comp