

Data management

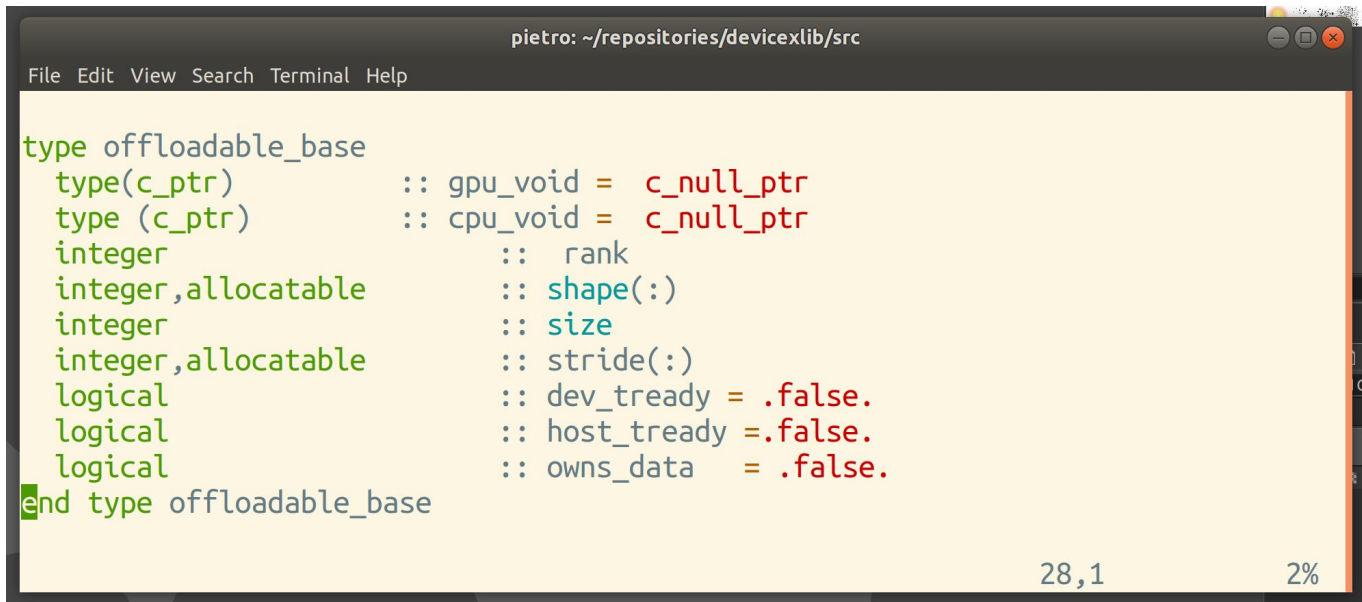
- Local data: avoid allocations using the buffer manager provided by deviceXlib allows to use a preallocated space
- Global data:
 - space on host and device allocated at initialization
 - we need to synchronize data
 - we want to avoid redundant updates
 - we would like to keep the same source code
- Examples of global variables: evc, gg, nhtongl, deeq, becp, vkb, mill , ...

Current cuda Fortran version approach

- device twin variables `_d` are defined independently in the specific `_gpu` modules
- we use `using_var` subroutines to:
 - checks and update the content of the variable before using it
 - keeps track of modifications of variables content

The offloadable type

- provides a general tool for managing global variables which must be accessed by the host and the device



```
pietro: ~/repositories/devicecplib/src
File Edit View Search Terminal Help

type offloadable_base
  type(c_ptr)      :: gpu_void = c_null_ptr
  type(c_ptr)      :: cpu_void = c_null_ptr
  integer          :: rank
  integer,allocatable :: shape(:)
  integer          :: size
  integer,allocatable :: stride(:)
  logical          :: dev_tready = .false.
  logical          :: host_tready = .false.
  logical          :: owns_data  = .false.
end type offloadable_base
```

28,1 2%

Type specific extension

```
pietro: ~/repositories/devicexlib/src
File Edit View Search Terminal Help

type offloadable_complex
  type(offloadable_base)      :: base
  complex(dp), pointer       :: gpu_pt(:) => NULL()
  complex(dp), pointer       :: cpu_pt(:) => NULL()
#if defined(__CUDA)
  attributes, device         :: gpu_pt
#endif
end type offloadable_complex

type offloadable_real
  type(offloadable_base)      :: base
  real(dp), pointer           :: gpu_pt(:) => NULL()
  real(dp), pointer           :: cpu_pt(:) => NULL()
#if defined(__CUDA)
  attributes, device         :: gpu_pt
#endif
end type offloadable_real
```

46,1 4%

Interfaces: initialization

```
pdelugas@brenta:/scratch/pdelugas/devicexlib/src
File Edit View Search Terminal Help

subroutine init_data_complex(a_manager, a, a_d, rank, shape, descsys)
  implicit none
  complex(dp),target :: a(:)
  complex(dp),target :: a_d(:)
  type(offloadable_complex),intent(out) :: a_manager
#if defined(__CUDA)
  attributes(device) :: a_d
#endif
  integer :: rank
  integer :: shape(rank)
  type(data_flags) :: descsys
  !
```

Interfaces: pointer assignement

File Edit View Search Terminal Help

```
interface
  subroutine associate_write (a_manager, mode, ptr, descsys)
    implicit none
    type(offloadable_<T>)      :: a_manager
    <T>,pointer,[device]       :: ptr(:)  !! define a procedure for the device ptr as well
    character                  :: mode
    type(data_flags)           :: descsys
  end subroutine associate_write
end interface
!
```

~
~
~
~
~
~

Plans.

- Replace the twinned variables and their management with offloadable variables.
- Include these features in the main code base.
- Is it possible to implement the same features with openMP ?