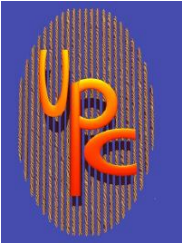


5th UPC Developers Workshop

UPC Collectives Extensions Proposals

Paul Hargrove	Berkeley
Dan Bonachea	LBNL
Rajesh Nishtala	Berkeley
Brian Wibecan	HP
Steve Seidel	MTU
Zinnu Ryne	MTU

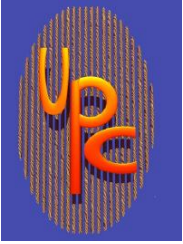


MTU extensions

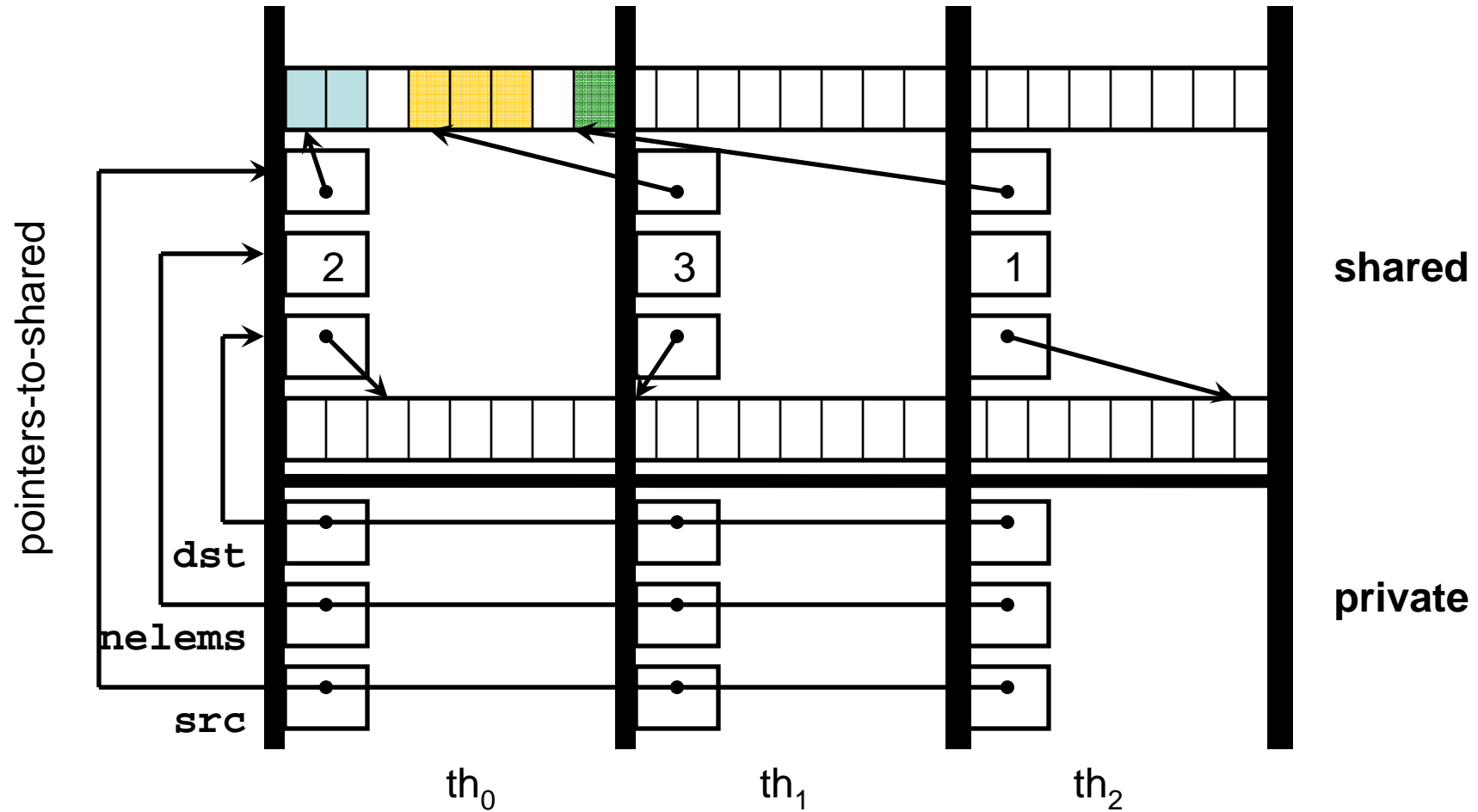
```
upc_all_relocalize_x (  
    shared void * shared * restrict dst,  
    shared const void * shared * restrict src,  
    shared size_t * nbytes, upc_flag_t mode );
```

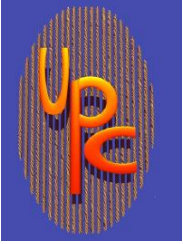
where *relocalize* is

- broadcast
 - scatter
 - gather
 - gather_all
 - exchange
- (Individual signatures vary)



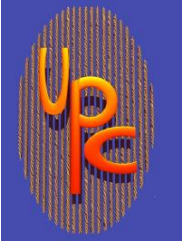
upc_all_scatter_x()





Asynchronous collectives

- Functions return `upc_coll_handle_t`
- Compatible with UPC v1.2 standard
- Specified by `upc_flag_t UPC_ASYNC`
- Various completion functions provided
- Out-of-order completion is allowed
- Completion function respects `...OUTSYNC` mode
- `UPC_HANDLE_COMPLETE` is the “null” handle



Async Completion Functions

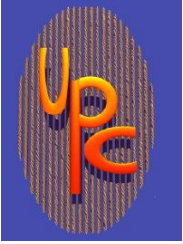
```
upc_coll_handle_t handle;
```

```
...
```

```
handle = upc_all_broadcast(...);
```

```
...
```

```
upc_wait(handle);
```



Async Completion Functions

```
void upc_wait(upc_handle_t h)
```

blocks until complete

```
int upc_test(upc_handle_t h)
```

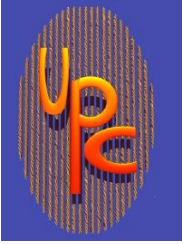
returns 0 immediately if not complete, otherwise
completes and returns nonzero

```
int upc_waitany(count, upc_handle_t * H)
```

blocks until any collective is complete, returns index

```
int upc_testany(count, upc_handle_t * H)
```

returns 0 immediately if none complete, otherwise
completes one function and returns index



Async Completion Functions

`void upc_waitall(count, upc_handle_t * H)`

waits for all calls with handles in array to complete

`int upc_testall(count, upc_handle_t * H)`

returns positive int and completes all handles if all calls are complete; else returns negative int

`int upc_wait_some(count upc_handle_t * H)`

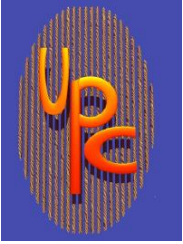
completes at least one call, completes all currently complete calls, returns count of completed call(s)

`int upc_test_some(count upc_handle_t * H)`

completes all currently complete calls (if there are some) and returns count of completed call(s)

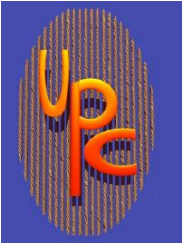
`int upc_get_status(upc_handle_t handle)`

returns positive int if call is complete; else returns 0



Berkeley extensions

- Reduce and prefix reduce
 - support missing C99 types
 - `signed long long` `unsigned long long`
`_Complex float` `_Complex double`
`_Complex long double` `_Bool`
(not `_Imaginary`)
 - adds an “exclusive” prefix reduce
 - adds `upc_all_reduce_all()`
as if `upc_all_reduce` followed by `upc_all_broadcast`



Berkeley extensions

- Proposes several possible “in-place” extensions for relocalization collectives

- Favored proposal replaces `src` argument with

UPC_IN_PLACE:

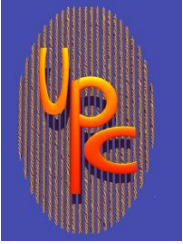
`upc_all_exchange` as if `src==dst`

`upc_all_permute` as if `src==dst`

`upc_all_prefix_reduce` as if `src==dst`

`upc_all_gather_all` as if `src==dst`

- For `upc_all_broadcast` replace `src` with `UPC_IN_PLACE(root)` as if `src==&dst[root]`



Berkeley extensions

- Non-single-valued arguments, variable **nbytes**
 - The MTU proposal addresses this, though the `upc_all_relocalize_x` extensions do require `src`, `dst`, and `nbytes` pointers (to arrays) to be single-valued.
- Non-shared (private) arguments
 - A proposal is needed
- Teams
 - *“At this time we ask that you turn off and stow all electronic equipment ...”*