

Open Spec Issues: Misc

- Calling C libs that may or may not be thread-safe
- C99 for loop declaration
- Pointer equality (comparing shared and non-shared pointers with ==)
- What happens to phase during casts
 - 6.3.5: gives cases, but doesn't say what happens when none of the cases hold
 - Should say that the phase remains the same
 - Jason: send wording to Bill

Subtracting Pointers

- Subtract 2 pointers to shared that are to the same block, but are not in phase...
 - Is slow in general because of mods
 - Can be done quickly pointers are assumed to be in phase
- Proposal:
 - Make the out-of-phase case undefined
 - Would need some work to get the definition precise
 - Dan will try to do this in 3 lines of text
 - Use the general/slow case code throughout?

Collectives

- Collectives
 - Further additions/deletions: additions will be considered only with implementation experience
 - Bill will ask about deprecating `upc_sort` from collectives
 - Create words to replace examples
- Collectives short-term spec issue:
 - Mostly spec-like, but should be checked as well

Memory Consistency

- Move forward with two parts of consistency spec
 - Think about the informal words to make sure they say what we want
 - In the Annex, remove ambiguities
 - Implementers: read the full paper, not just the parts in the current spec – send feedback!
- Make it more “spec-ish” – remove footnotes
 - Bill will send his version to Dan/Chuck/Kathy
 - Don’t add local tex-isms back in (Bill removed them)

UPC I/O

- I/O issues and breadth:
 - Will have better data by SC04, and can finalize spec then.
 - 2 “notes” in I/O: Bill will delete
- Try to translate into spec-ease
 - E.g., remove: “Let us first define what we mean by...”
 - Remove duplicate paragraphs (factor out common terminology)
 - Bill will make the first pass

UPC Locks

- Short term spec issue: UPC Locks
 - Remove anthropomorphism in the spec
 - E.g., upc-unlock “frees” the lock, which it doesn’t (I.e., it doesn’t deallocate)
- Pairwise synchronization
 - Add synchronization similar to lock, but for semaphore-like behavior
 - Fetch-and-add operations; atomic with respect to other F&A, but not other stores/loads
 - Can be constructed with lock in a general way, but we could put them in
 - Beyond 1.2: work on semantics

Acknowledgements

- Bill will try to pare down
- Will attempt to avoid offending anyone
- Please politely disagree

For the record: this is not a Bad Case

- We do not currently prohibit:

```
int x;
```

```
shared int *p = & (shared int) x;
```

- Should it be a compile-time error: but the following should be legal:

```
int x;
```

```
shared int y;
```

```
y = x;
```

```
y = (shared int) x;
```

- Proposal: result of the cast is not an L-value
- This is a non-issue