**Agenda**

|  |  |  |  |
| --- | --- | --- | --- |
| 8:00 | :30 | Gathering |  |
| 8:30 | :15 | Introductions, ground rules, agenda bashing |  |
| 8:45 | :45 | Timeline to 1st release | What has to happen before rel 1.0?  Intermediate releases leading to rel 1.0? |
| 9:30 | :60 | Simple walk through: verbs, libfabrics | Frank Berry |
| 10:30 | :15 | Break |  |
| 10:45 | :75 | Set-up a pseudo-code exercise | Assign small groups |
| 12:15 | :45 | lunch |  |

**Agenda bashing**

Simple walkthrus – an hour should be enough

Pseudo-code

* For MPI, normally have broad topics such as tools, point to point, collectives, one-sided ops, etc. essentially broken down by chapter.

Liran Liss – cannot attend, but Rich Graham will be present on Tuesday only. MLNX therefore requests the verbs extension topic be moved to Tuesday. Purpose is to show how the basic concepts raised over the past year can be exposed using the verbs interface. Plan to touch on key issues such as efficient APIs, address vectors, progress model, etc. Suggestion was made to contact Roland Drier (kernel maintainer) to see how receptive he is toward the sort of approaches being proposed.

Are there IP-related issues? Example was given of RAMBus, which introduced some patented items to a standard thus forcing adopters to license the items.

AR – Paul to follow up with the Board of Directors. This will probably wait until October when Jim Ryan returns from sabbatical.

Models to guide implementation – the objective is to identify ways of creating models that can be used to guide those accessing the APIs, both from the application layer looking down and the provider layer looking up. Currently, things like ib microbenchmarks are used as (inadequate) stand-ins.

* Howard will support the discussion from the provider level.
* Sean will lead the discussion from the app layer looking down.

Management and addressing assumptions – We haven’t focused on management interactions yet, but clearly it will impact the APIs and the way applications use the API. Move this to Wednesday at 9:00am to line up with a scalable SA meeting that regularly occurs on Wednesday.

Need more than an one hour for Requirements wrap up.

**Timeline discussion**

What is 1.0? Or is it really 0.x?

* Identify features needed.
* At least one reference provider (user space, but maybe both?)
  + Should be able to stimulate unexpected behaviors causing common application errors. “semantic stress test”
  + Nice to have a provider that would run on any POSIX provider.
* Is this 1.0 for libfabrics (including providers?) or 1.0 for the APIs?
* Compatible with RDMA CM, and/or management infrastructure (may be provider-based)
* Does not have to be performant.
* Must have a query facility, given that not all APIs will be available to begin.
* Unit test - Test suite for APIs
* Functional test - At least one ‘reference’ application, driven by the ‘minimum APIs discussion’
* Delay support for verbs extensions
* Mechanism to allow vendor extensions
* A versioning mechanism
* Code snippets/examples
* A defined release mechanism
* Fully documented, “Google-able” man pages
* Needs to have:
  + Upstream repo location.
  + Upstream tarball location for source code releases.
  + Independent release schedule, not tied to OFED.
* To Github or not to Github…that is the question.
  + Bug tracking mechanism
  + Volunteers: Jeff S., Jeff Hammond, Sean Hefty, Ken R.

IP question – This is a question for the board of directors.

AR- Paul – raise this at the BoD on Jim Ryan’s return from sabbatical (10/4/14).

**Simple Code Walkthrough – Arun Ilango / Frank Berry / Sean Hefty**

rc\_pingpong.c will be posted to the OFI WG website

* Initialization section:
  + Expectation is that there will be two types of event queues, a ‘fast path’ eq and an eq reserved for control type methods.
  + Connect is expected to be a non-blocking operation, requiring binding to an eq/cq of some sort.
  + There is a tradeoff between a ‘unified queue’ and separated queues.
    - Separate queues may mean that connection events (polled only infrequently) may be missed.
    - Unified queue requires a branch to go check for connection completions, reducing the efficiency of the fast path.
* Fi\_send – some question about passing in pointers to the fi\_send call. It is recognized that the current structure may be problematical for certain architectures (ARM?), but there is currently no alternative proposal on the table.
* Completion events
  + Currently there is no completion event that does callbacks.

**Application Mapping Over Libfabrics walkthrough – Charles Archer**

* + Query interface - For tagged messages
  + Address vector – two options
    - Addr
    - Addr\_index
  + Completions – multiple types
    - Locally complete
    - Remote complete (operation finished on the remote side
    - Remote write to RAM
    - Others?
    - Distinguish between the API and the application. Application only needs to know when the buffer can be re-used.
  + Query interface – RMA/Atomics
  + Query interface – Event Queue
  + Query limits
  + Short Send
  + Large Message Send
  + RMA Completions