

ICSC ITWG Meeting Minutes, 23. Nov. 2004
Taking minutes: IBM (fn)

Meeting attendance diagram (if you have more than 1 minus within the last four ITWG meetings, you are not eligible to vote):

mtg	date	hp	ibm	netapp	sun
---	-----	--	---	-----	---
m-3		+	+	-	-
m-2		+	+	-	-
m-1		+	+	-	-
m-0	nov 23	+	+	-	-

Present:

jim hamrick (jh) [HP]
fredy neeser (fn) [IBM]
jay rosner (jr) [HP]

Minutes to approve: Email from Jim Hamrick, subject "Draft 2 of ITWG meeting minutes for 11/09/2004", sent 11/12/04 1:40PM PT - Approved.

Action item review

AI (jr): Add text to diagram to say that TCP ACK could be piggybacked on the first FPDU. Work in progress.

jr: Will be in iWARP Implementer's Guide (separate section).

AI (jr): Add discussion of how IRD/ORD negotiation is done into app usage section of it_ep_connect man page, and reference to it from app usage section of it_ep_accept man page.

jr: Work in progress.

AI (fn): Adjust current text describing how to unbind a narrow RMR after QP disconnected.

Deallocate bound MW

Dellocate bound MR

fn: Work in progress (application usage section).

CM issues:

all: Defer discussion of CM draft man pages

MM issues - Discussion based on MM Detailed Requirements v0.97a:

Email thread started by Jay Rosser, subject "Additional MM requirement - New parameter for it_lmr_create2() for privileged consumers", sent 11/18/04:

jr: virtual address space qualifier for it_lmr_create.

jr: alternative would be a new, ITAPI-provided mapping service that takes a virtual address, a length and a space ID and returns a PBL, where PBL represents bus addresses.

jr: intent is for kernel consumer to register a pinned user mode virtual address range. (Minuter's note: Since the address range is already pinned, this must be it_lmr_create2 called with a PBL or it_lmr_link. When it_lmr_create2 is called without passing a PBL, the ITAPI1/2 Implementation makes sure that the memory region gets pinned).

jh: desire to minimize the amount of time the bus addresses are used. bus address space is often smaller than physical address space.

fn: how can this goal be achieved with an ITAPI mapping service?

jr: you don't want to commit bus address space too early.

fn: I have a two-stage model in my mind. First stage: pinning of virtual memory area (OSV-provided mechanism) results in physical addresses. Second stage: conversion from physical addresses to bus addresses.

jh: problem is that amount of registered memory may exceed bus address space. can't fail a work request with an out-of-resource completion error.

jr: layered io model. class driver level can generate a large number of operations that could possibly exceed bus address space. an entity below the class driver may need to serialize the operations.

jr: would not be problematic if class driver would use pbls to register memory. only problematic if user-mode virtual addresses are being registered.

(Minuter's comment: This problem occurs only if it_lmr_create2 is called without a PBL by a *privileged* consumer (i.e., kernel consumer). Note that the use models MM-6.2.D1.2.1/2 are intended for non-privileged consumers - should this be stated explicitly? Privileged (i.e. kernel) consumers are expected to pass a PBL when calling it_lmr_create2.)

jr: model 1: Privileged kernel consumers only:

Step 1: Privileged kernel consumer (not in user process context) takes a virtual address range and a virtual address space ID and invokes an OS-provided pinning service.

Step 2 (Later on, when the privileged kernel consumer decides to commit bus resources, and before it actually performs the I/O operation):

Privileged kernel consumer calls an ITAPI mapping service (which wraps

a corresponding OS-provided mapping service) to commit bus resources and map the virtual address range to a PBL representing bus addresses. Step 3: Privileged kernel consumer performs fast registration using PBL (bus addresses).

fn: OS-provided pinning service may perform optimizations for variable page size. Has underlying PBL (physical addresses).

all: can get an out-of-resource error when calling ITAPI mapping service contending for bus addresses.

fn: how would ITAPI mapping service be implemented?

jr: ITAPI mapping service would interact with system architecture to allocate bus resources. ITAPI mapping service wraps a typical OS service, similar to the way the `ri_vm_pin_range` call in HP's RNIC-PI proposal wraps an OS pinning service.
(Minuter's comment: I further assume that the mapping service corresponds to the service provided by the `ri_hal_map_mem` call in HP's RNIC-PI proposal. Correct?)

jr: ITAPI mapping service would typically be invoked right before fast registration.

jr: Both pinning and mapping may be expensive operations.

fn: Separate pinning and mapping service to solve the out-of-bus-address resource problem.
(Minuter's comment: I think one can benefit from this separation only if it is reasonable and/or possible to defer the mapping process relative to the pinning process. The question is why would a kernel consumer pin a user-mode memory range (Step 1 above) in advance, without wanting to perform an I/O operation at the same time? - A possible reason is that a user-space application may have posted a large Send, which cannot be converted to a single RDMA Send due to lack of bus resources. Hence, the kernel consumer would break up the large Send into several smaller RDMA Sends and commit/release bus resources on demand. Does this make sense?)

jr: I am thinking of a user application (not an ITAPI consumer) performing I/O operations. Underlying subsystem (kernel ITAPI consumer) detects that I/O operation is sufficiently large to warrant zero-copy model. Kernel consumer has to serialize the user's I/O operations, making sure that the bus address space is not exceeded.

fn: User application may use SDP?

jr: Yes, that's a possible scenario. Would have an SDP implementation as a kernel consumer.

jr: there are other minor motivations for ITAPI mapping service. Convenience for kernel API consumers to use a mapping service outside the normal device driver framework.

jh: may need to indicate the IA that is going to use the address range.

jr: AI to write a proposal for an ITAPI mapping service that maps a virtual address range (virtual address, length, space id, and possibly ia that is going to use the address range) to a bus address range. ITAPI PBL (bus addresses) would be output from this service. (Minuter's comment: I suppose an ITAPI unmap service would also be needed then.)

fn: Going directly from virtual addresses to bus addresses is not necessarily the only solution. I am considering a two-stage model. If the OS pinning service returns the PBL (physical addresses) underlying the pinned memory region, then the OS mapping service could alternatively take the PBL (physical addresses) as input and generate a PBL (bus addresses) as output.

jr: Valid point.

fn: PC architecture doesn't distinguish physical addresses and bus addresses (calling them physical addresses), while other architectures do.

jh: Question of convention. On the PC, one could have alternatively used the term bus addresses only.

fn: We need to be careful distinguishing PBLs for physical and bus addresses. Muddle of terminology. Verbs also use "physical addresses".

jh: And I think they mean "bus addresses".

jh: PBLs for ITAPI should be suitable to be used directly by the card

fn: e.g., for fast registration purposes or when using the STag of 0.

Additional agenda item by fn: Trying to close on MM Detailed Requirements.

fn: Trying to close on narrow window binding requirements first. These were already voted, but some needed to be changed. Changes are related to remotely detected errors (MM-9.5.D1 etc.).

fn: Another change is MM-9.1.D1. This change allows Implementations to support more RMR types than required. For instance, an iWARP Implementation could also support Wide RMRs. There's nothing in the DDP/RDMAP protocols that would prevent Wide RMRs. (Minuter's comment: Only by strictly following the RDMAC Verbs, a restriction to Narrow RMRs results.)

jr: This change sounds reasonable to me.

fn: Two more changes. MM-9.7.D1.2/3 are already covered by MM-9.3.D1.4/5.

fn: AI - FN to send out MM Detailed Requirements v0.98 tomorrow along with a list of changed narrow window binding requirements to be voted in the next meeting.

jr: Regarding remaining MM Detailed Requirements, still need discussion. Agenda item for next call - Narrow down the open issues / sections that are ready for vote.

Additional agenda item by fn: Local access privileges for RMRs.

fn: In ITAPI1, `it_rmr_bind` only specifies remote access privileges. However, `IT_PRIV_ALL` also includes local access privileges.

jh: `it_rmr_bind` ignores the local component of `IT_PRIV_ALL`.

fn: In ITAPI2, an RMR can be used locally as a data sink (iWARP feature). New call for this is `it_post_rdma_read_to_rmr`.

fn: It seems that binding an RMR with remote write access privilege is sufficient to use an RMR as a data sink in an RDMA read request.

fn: It seems that local access privileges of an RMR are irrelevant because an RMR can be accessed only through a remote operation. (Minuter's comment: This is even true for `it_post_rdma_read_to_rmr`, if we note that writing to the RMR sink is caused by an incoming RDMA Read Response).

Meeting adjourned about 10 minutes late.