MEETING 27 AGENDA:

Call to Order  UPAMD Power Subgroup meeting  ï  Paul Panepinto  5pm Pacific 3 Nov 2011

I.  Introductions/Attendance
    Edgar, Bob and Paul

II. Approval of 11/3/2011 Power Subgroup Agenda
    Motion to Approve: Bob; Seconded Edgar.

III. Presentation of 10/20/2011 Power Subgroup Meeting 26 Minutes
    Motion to Approve: Edgar; Seconded Bob.

IV.  IEEE Call for Patents. See

V.  Review of the current draft of the spec
    We reviewed the messages and noted that Bob has now grouped them according to
    socket (functional category.) We looked at the timestamp message as a mechanism
    for enabling sinks and sources to get a common understanding of realtime. We
    noted that the timestamp does not guarantee time synchronization. It just provides
    the ability for any device to get the time from or set the time to a specific time,
    accurate to a worst-case message delay of about 1 second. This is useful for sources
    and sinks to agree upon what time a device may wish to wake up or charge its
    battery, for example.

    This is also useful for testing and error reporting / debugging ï to know the time
    devices believe events occurred.

    We discussed what to do when devices have a radically different notion of time.
    Several options exist to reconcile the differences. A system monitor can impose
    time synchronization or the IPV6 NTP protocol may be implemented. For our
    purposes, this does not need to be resolved by UPAMD, at least at this time.

    We looked at Sender Class and thought through the implications of CAN-bus
    message processing order on our header definition. For example, we noted that as
    we defined it, Source being set as 1 would have a lower CAN-bus message
    processing order than a Sink being set as a 0. This is unlikely to be a problem,
    because there are several other priority bits that would likely dictate the order in
    which CAN-bus messages are processed. However, we did agree to make Sink = 1.
    We may move the Extended Sender Class indicator up because of the order in which
    CAN-bus processes messages. We didnât note any other obvious issues related to
    message order processing.

    The Initialization Message (sent from Sink to Source) contains a byte indicating the
    port on which the Sink uses to talk to the Source. It was thought this is essential for
the wormhole routing scheme under consideration for addressing through multiple hops. However, questions were raised about whether or not this information is indeed necessary for the wormhole routing scheme to work.

We discussed the need for every source and communicating sink to have a State Table. These State Tables can be used by any device to construct the network topology in some sort of discovery process. Several ideas have been discussed and we will review them and try to develop a recommended discovery process soon.

Finally, Edgar suggested we decide upon a mechanism to deal with loops. In previous discussions, 6 methods were considered. Now, each of us will rank our top 2 choices and see if we can use that to make a final recommendation for handling loops in the topology. As a reminder, a loop is a condition where a Source may request power from a connected Source to provide it to a connected Sink, and that request may be forwarded one or more times eventually back to the originating requestor, as the system tries to find available power. A mechanism is needed to avoid infinitely requesting power.

VI. New business?
We briefly reviewed new messages that provide access to register data and agreed to review which may be required, which may be optional and which can be decided upon for future revisions.

VII. Adjourn
Motion to Adjourn: Bob at 7:19pm Pacific