

Surreal interconnect requirements as submitted to the p1394.1 committee

Dr. David V. James, Sony
3300 Zanker Road, MS SJ3C1
San Jose, CA 95134-4591
Phone: 408 955-6295
FAX: 408 955-4591
Email: dvj@alum.mit.edu

October 12 1999

**This contribution is one of several, presented for independent review.
An overall contribution, which provides the context for multiple contributions,
is also provided in BR047R08.**

GASP packets have the unfortunate property of being discarded under congestion conditions.
Some events, such as bus-address-changed, has significant repercussions if lost.
To avoid such event losses, events are defined to be mergable, and the merging algorithm is proposed.

High performance Serial Bus bridges

7.5.3 Event messages

TBD—It may become necessary to time-stamp events, for recovery of isochronous resources after a net_changed event.

An event message broadcasts state change information, where that state change information has the format illustrated in figure 1. If two event messages must be merged, a local-bus *source_ID* value is generated and the event-send bits $e[i]$ shall be OR'd together. The intent is to provide a reliable event notification service, to alleviate the need for excessive numbers of bus resets.

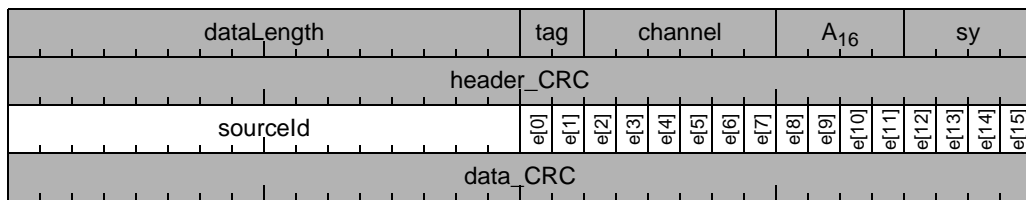


Figure 1—Event message format

The 32 $e[0]$ through $e[15]$ bits specify the event which is being indicated, as listed in table 1.

Table 1—Bridge related events

Number	Reference	Name	Description
TBD		new_node	Available event indications
TBD		new_bus	Reserved event indications
TBD		—	TBD
TBD-TBD		—	TBD
TBD		—	TBD
TBD-TBD		—	Reserved

NOTE—If possible, a Surreal interconnect should transport event packets as flow-controlled multicast packets, to avoid the necessity of merging event packets in congested bridges and/or nodes.

On Serial Bus, there is no way to flow-control GASP-packet delivery. Since queue-overflow conditions are unavoidable, events are merged when they would be otherwise discarded, as specified by equation 1.

```
// Merge events e1 and e2, return merged event in e1
MergeEvents(e1Ptr,e2Ptr,thisNodeId)
{
    e1Ptr->bits|= e2Ptr->bits;
    if (e1Ptr->sourceId != e2Ptr->sourceId) {
        // Broadcast restarts from here, with unknown localId flag
        e1Ptr->sourceId.busId= thisNodeId.busId;
        ePtr->sourceId.localId= 0X3F;
    }
    return(ePtr);
}
(1)
```