545

AT893-0282 RMC
D. W. Sheffield

IBM® Technical Disclosure Bulletin Vol. 37 No. 02B February 1994

Avoid Remapping of Data Buffers in AIX Device Drivers

Disclosed is a method for a RISC System/6000* AIX* device driver to avoid remapping data buffers for DMA on each I/O operation.

Generally, for each I/O operation that involves DMA of a data buffer to or from an adapter, a device driver must call d_master with a sufficient TCW (Translation Control Word) range to map the buffer. These data buffers are often page size buffers and are generally reused quite frequently. Therefore, a device driver will often receive the same data buffer many times for different I/O operations.

Existing drivers simply ignore the fact that these buffers are being reused, and remap the buffers on each operation. Instead of this approach, a driver could use a least recently used algorithm to allocate TCWs from a table. When a buffer is passed down, the table could be checked, and if the same TCWs that were used on the previous operation are still available, they could be reused. This would eliminate the call to d_master, although the driver would still have to handle the cache management (depending on the hardware platform).

A possible problem with this scheme is that if the buffer is unpinned and then pinned again, it could have a different real address. Re-using the same TCW at the device driver would not work in this case. Unfortunately, there is no way for the driver to know if this has occurred since it last received this same buffer (virtual address).

The solution to this problem involves a simple set of kernel services. These services must maintain a flag for each driver using the services that indicates if it is safe for the driver to reuse a TCW for that page. This can be easily done using the dma channel number, since it is unique for each driver. The table should be marked invalid (for all dma channels) if the page is unpinned. This way a driver can check this table before reusing a previously mapped TCW. The buffer will have to be remapped like normal (using the d_master kernel service) if the page has been unpinned since it was last mapped.

Example kernel services:

```
    map_set (page_num, dma_channel_mask)
        if (tablefflpage_num & MASK“.page_num = = page_num)
            tablefflpage_num & MASK“.channel_nums |= dma_channel_mask;

    map_clear (page_num, dma_channel_mask)
        if (tablefflpage_num & MASK“.page_num = = page_num)
            tablefflpage_num & MASK“.channel_nums &= dma_channel_mask;

    map_query (page_num, dma_channel_mask)
```

```
rc = (tableffipage_num & MASK“.page_num = = page_num) &&
    ((tableffipage_num & MASK“.channel_nums & dma_channel_mask)
     = = dma_channel_mask);
```

The map_query() routine could be a macro instead of a function call, since this is the routine that is performance-critical. Therefore the only additional pathlength to implement this scheme in a driver is a direct table lookup, which may save the cost of mapping the buffer.

* Trademark of IBM Corp.