```
**********************************************************
  443575 Tue Dec  4 16:29:33 2012
new/usr/src/uts/common/io/scsi/adapters/mpt_sas/mptsas.c
re #6833 rb1771 less verbosity in mptsas
**********************************************************
______unchanged_portion_omitted_

12695 /*
12696  * mptsas_add_intrs:
12697  *
12698  * Register FIXED or MSI interrupts.
12699  */
12700 static int
12701 mptsas_add_intrs(mptsas_t *mpt, int intr_type)
12702 {
12703         dev_info_t      *dip = mpt->m_dip;
12704         int             avail, actual, count = 0;
12705         int             i, flag, ret;

12707         NDBG6(("mptsas_add_intrs:interrupt type 0x%x", intr_type));

12709         /* Get number of interrupts */
12710         ret = ddi_intr_get_nintrs(dip, intr_type, &count);
12711         if ((ret != DDI_SUCCESS) || (count <= 0)) {
12712                 mptsas_log(mpt, CE_WARN, "ddi_intr_get_nintrs() failed, "
12713                     "ret %d count %d\n", ret, count);

12715                 return (DDI_FAILURE);
12716         }

12718         /* Get number of available interrupts */
12719         ret = ddi_intr_get_navail(dip, intr_type, &avail);
12720         if ((ret != DDI_SUCCESS) || (avail == 0)) {
12721                 mptsas_log(mpt, CE_WARN, "ddi_intr_get_navail() failed, "
12722                     "ret %d avail %d\n", ret, avail);

12724                 return (DDI_FAILURE);
12725         }

12727         if (0 && avail < count) {
12727         if (avail < count) {
12728                 mptsas_log(mpt, CE_NOTE, "ddi_intr_get_nvail returned %d, "
12729                     "navail() returned %d", count, avail);
12730         }

12732         /* Mpt only have one interrupt routine */
12733         if ((intr_type == DDI_INTR_TYPE_MSI) && (count > 1)) {
12734                 count = 1;
12735         }

12737         /* Allocate an array of interrupt handles */
12738         mpt->m_intr_size = count * sizeof (ddi_intr_handle_t);
12739         mpt->m_htable = kmem_alloc(mpt->m_intr_size, KM_SLEEP);

12741         flag = DDI_INTR_ALLOC_NORMAL;

12743         /* call ddi_intr_alloc() */
12744         ret = ddi_intr_alloc(dip, mpt->m_htable, intr_type, 0,
12745             count, &actual, flag);

12747         if ((ret != DDI_SUCCESS) || (actual == 0)) {
12748                 mptsas_log(mpt, CE_WARN, "ddi_intr_alloc() failed, ret %d\n",
12749                     ret);
12750                 kmem_free(mpt->m_htable, mpt->m_intr_size);
12751                 return (DDI_FAILURE);
12752         }
```



```
12754         /* use interrupt count returned or abort? */
12755         if (actual < count) {
12756                 mptsas_log(mpt, CE_NOTE, "Requested: %d, Received: %d\n",
12757                     count, actual);
12758         }

12760         mpt->m_intr_cnt = actual;

12762         /*
12763          * Get priority for first msi, assume remaining are all the same
12764          */
12765         if ((ret = ddi_intr_get_pri(mpt->m_htable[0],
12766             &mpt->m_intr_pri)) != DDI_SUCCESS) {
12767                 mptsas_log(mpt, CE_WARN, "ddi_intr_get_pri() failed %d\n", ret);

12769                 /* Free already allocated intr */
12770                 for (i = 0; i < actual; i++) {
12771                         (void) ddi_intr_free(mpt->m_htable[i]);
12772                 }

12774                 kmem_free(mpt->m_htable, mpt->m_intr_size);
12775                 return (DDI_FAILURE);
12776         }

12778         /* Test for high level mutex */
12779         if (mpt->m_intr_pri >= ddi_intr_get_hilevel_pri()) {
12780                 mptsas_log(mpt, CE_WARN, "mptsas_add_intrs: "
12781                     "Hi level interrupt not supported\n");

12783                 /* Free already allocated intr */
12784                 for (i = 0; i < actual; i++) {
12785                         (void) ddi_intr_free(mpt->m_htable[i]);
12786                 }

12788                 kmem_free(mpt->m_htable, mpt->m_intr_size);
12789                 return (DDI_FAILURE);
12790         }

12792         /* Call ddi_intr_add_handler() */
12793         for (i = 0; i < actual; i++) {
12794                 if ((ret = ddi_intr_add_handler(mpt->m_htable[i], mptsas_intr,
12795                     (caddr_t)mpt, (caddr_t)(uintptr_t)i)) != DDI_SUCCESS) {
12796                         mptsas_log(mpt, CE_WARN, "ddi_intr_add_handler() "
12797                             "failed %d\n", ret);

12799                         /* Free already allocated intr */
12800                         for (i = 0; i < actual; i++) {
12801                                 (void) ddi_intr_free(mpt->m_htable[i]);
12802                         }

12804                         kmem_free(mpt->m_htable, mpt->m_intr_size);
12805                         return (DDI_FAILURE);
12806                 }
12807         }

12809         if ((ret = ddi_intr_get_cap(mpt->m_htable[0], &mpt->m_intr_cap))
12810             != DDI_SUCCESS) {
12811                 mptsas_log(mpt, CE_WARN, "ddi_intr_get_cap() failed %d\n", ret);

12813                 /* Free already allocated intr */
12814                 for (i = 0; i < actual; i++) {
12815                         (void) ddi_intr_free(mpt->m_htable[i]);
12816                 }

12818                 kmem_free(mpt->m_htable, mpt->m_intr_size);
```

```
12819                   return (DDI_FAILURE);
12820           }

12822           /*
12823            * Enable interrupts
12824            */
12825           if (mpt->m_intr_cap & DDI_INTR_FLAG_BLOCK) {
12826                   /* Call ddi_intr_block_enable() for MSI interrupts */
12827                   (void) ddi_intr_block_enable(mpt->m_htable, mpt->m_intr_cnt);
12828           } else {
12829                   /* Call ddi_intr_enable for MSI or FIXED interrupts */
12830                   for (i = 0; i < mpt->m_intr_cnt; i++) {
12831                           (void) ddi_intr_enable(mpt->m_htable[i]);
12832                   }
12833           }
12834           return (DDI_SUCCESS);
12835 }
```

_____***unchanged_portion_omitted_***