Subject: [PATCH] e1000: e1000_probe resources cleanup
Posted by vaverin on Sun, 20 Aug 2006 07:00:15 GMT
View Forum Message <> Reply to Message

this patch fixes resources cleanup in e1000_probe()

Signed-off-by: Vasily Averin <vvs@sw.ru>

Thank you,
Vasily Averin

SWsoft Virtuozzo/OpenVZ Linux kernel team

```
--- linux-2.6.18-rc4/drivers/net/e1000/e1000_main.c.eprcln 2006-08-19 23:02:45.000000000
+0400
+++ linux-2.6.18-rc4/drivers/net/e1000/e1000_main.c 2006-08-20 10:48:36.000000000 +0400
@@ -696,21 +696,20 @@ e1000_probe(struct pci_dev *pdev,
   if ((err = pci_set_dma_mask(pdev, DMA_32BIT_MASK)) &&
       (err = pci_set_consistent_dma_mask(pdev, DMA_32BIT_MASK))) {
    E1000_ERR("No usable DMA configuration, aborting\n");
-   return err;
+   goto err_dma;
   }
   pci_using_dac = 0;
  }

  if ((err = pci_request_regions(pdev, e1000_driver_name)))
-  return err;
+  goto err_pci_reg;

  pci_set_master(pdev);

+ err = -ENOMEM;
  netdev = alloc_etherdev(sizeof(struct e1000_adapter));
- if (!netdev) {
-  err = -ENOMEM;
+ if (!netdev)
   goto err_alloc_etherdev;
- }

  SET_MODULE_OWNER(netdev);
  SET_NETDEV_DEV(netdev, &pdev->dev);
@@ -725,11 +724,10 @@ e1000_probe(struct pci_dev *pdev,
  mmio_start = pci_resource_start(pdev, BAR_0);
  mmio_len = pci_resource_len(pdev, BAR_0);

+ err = -EIO;
```

```
  adapter->hw.hw_addr = ioremap(mmio_start, mmio_len);
- if (!adapter->hw.hw_addr) {
-  err = -EIO;
+ if (!adapter->hw.hw_addr)
   goto err_ioremap;
- }

  for (i = BAR_1; i <= BAR_5; i++) {
   if (pci_resource_len(pdev, i) == 0)
@@ -774,6 +772,7 @@ e1000_probe(struct pci_dev *pdev,
  if ((err = e1000_sw_init(adapter)))
   goto err_sw_init;

+ err = -EIO;
  /* Flash BAR mapping must happen after e1000_sw_init
   * because it depends on mac_type */
  if ((adapter->hw.mac_type == e1000_ich8lan) &&
@@ -781,13 +780,11 @@ e1000_probe(struct pci_dev *pdev,
   flash_start = pci_resource_start(pdev, 1);
   flash_len = pci_resource_len(pdev, 1);
   adapter->hw.flash_address = ioremap(flash_start, flash_len);
-  if (!adapter->hw.flash_address) {
-   err = -EIO;
+  if (!adapter->hw.flash_address)
    goto err_flashmap;
-  }
  }

- if ((err = e1000_check_phy_reset_block(&adapter->hw)))
+ if (e1000_check_phy_reset_block(&adapter->hw))
   DPRINTK(PROBE, INFO, "PHY reset is blocked due to SOL/IDER session.\n");

  /* if ksp3, indicate if it's port a being setup */
@@ -830,7 +827,7 @@ e1000_probe(struct pci_dev *pdev,

  if (e1000_init_eeprom_params(&adapter->hw)) {
   E1000_ERR("EEPROM initialization failed\n");
-  return -EIO;
+  goto err_eeprom;
  }

  /* before reading the EEPROM, reset the controller to
@@ -842,7 +839,6 @@ e1000_probe(struct pci_dev *pdev,

  if (e1000_validate_eeprom_checksum(&adapter->hw) < 0) {
   DPRINTK(PROBE, ERR, "The EEPROM Checksum Is Not Valid\n");
-  err = -EIO;
   goto err_eeprom;
```

```
  }

@@ -855,7 +851,6 @@ e1000_probe(struct pci_dev *pdev,

   if (!is_valid_ether_addr(netdev->perm_addr)) {
    DPRINTK(PROBE, ERR, "Invalid MAC Address\n");
-   err = -EIO;
    goto err_eeprom;
   }

@@ -964,16 +959,33 @@ e1000_probe(struct pci_dev *pdev,
   return 0;

  err_register:
+ e1000_release_hw_control(adapter);
+err_eeprom:
+ if (!e1000_check_phy_reset_block(&adapter->hw))
+   e1000_phy_hw_reset(&adapter->hw);
+
   if (adapter->hw.flash_address)
    iounmap(adapter->hw.flash_address);
  err_flashmap:
+#ifdef CONFIG_E1000_NAPI
+ for (i = 0; i < adapter->num_rx_queues; i++)
+   dev_put(&adapter->polling_netdev[i]);
+#endif
+
+ kfree(adapter->tx_ring);
+ kfree(adapter->rx_ring);
+#ifdef CONFIG_E1000_NAPI
+ kfree(adapter->polling_netdev);
+#endif
  err_sw_init:
-err_eeprom:
   iounmap(adapter->hw.hw_addr);
  err_ioremap:
   free_netdev(netdev);
  err_alloc_etherdev:
   pci_release_regions(pdev);
+err_pci_reg:
+err_dma:
+ pci_disable_device(pdev);
   return err;
  }
```