

## Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-02/msg00719.html>

---

- *From:* Francois Romieu <[romieu@xxxxxxxxxxxxxx](mailto:romieu@xxxxxxxxxxxxxx)>
  - *Date:* Sat, 3 Feb 2007 01:39:52 +0100
- 

<edward\_hsu@xxxxxxxxxxxxxx> :  
[...]

```
@@ -123,8 +155,8 @@ static const int multicast_filter_limit
#define MAC_ADDR_LEN 6

#define RX_FIFO_THRESH 7 /* 7 means NO threshold, Rx buffer level before
first PCI xfer. */
-#define RX_DMA_BURST 6 /* Maximum PCI burst, '6' is 1024 */
-#define TX_DMA_BURST 6 /* Maximum PCI burst, '6' is 1024 */
+#define RX_DMA_BURST 7 /* Maximum PCI burst, '7' is unlimited */
+#define TX_DMA_BURST 7 /* Maximum PCI burst, '6' is 1024 */
```

Ok.

[...]

```
@@ -179,45 +208,18 @@ static const struct {
u8 mac_version;
u32 RxConfigMask; /* Clears the bits supported by this chip */
} rtl_chip_info[] = {
- _R("RTL8169", RTL_GIGA_MAC_VER_01, 0xff7e1880),
- _R("RTL8169s/8110s", RTL_GIGA_MAC_VER_02, 0xff7e1880),
- _R("RTL8169s/8110s", RTL_GIGA_MAC_VER_03, 0xff7e1880),
- _R("RTL8169sb/8110sb", RTL_GIGA_MAC_VER_04, 0xff7e1880),
- _R("RTL8169sc/8110sc", RTL_GIGA_MAC_VER_05, 0xff7e1880),
- _R("RTL8168b/8111b", RTL_GIGA_MAC_VER_11, 0xff7e1880), // PCI-E
- _R("RTL8168b/8111b", RTL_GIGA_MAC_VER_12, 0xff7e1880), // PCI-E
- _R("RTL8101e", RTL_GIGA_MAC_VER_13, 0xff7e1880), // PCI-E 8139
- _R("RTL8100e", RTL_GIGA_MAC_VER_14, 0xff7e1880), // PCI-E 8139
- _R("RTL8100e", RTL_GIGA_MAC_VER_15, 0xff7e1880) // PCI-E 8139
+ _R("RTL8169", RTL_GIGA_MAC_VER_8169, 0xff7e1880),
+ _R("RTL8169S/8110S", RTL_GIGA_MAC_VER_8169S, 0xff7e1880),
+ _R("RTL8169S/8110S", RTL_GIGA_MAC_VER_8110S, 0xff7e1880),
+ _R("RTL8169SB/8110SB", RTL_GIGA_MAC_VER_8169SB, 0xff7e1880),
+ _R("RTL8169SC/8110SC", RTL_GIGA_MAC_VER_8110SCd, 0xff7e1880),
+ _R("RTL8169SC/8110SC", RTL_GIGA_MAC_VER_8110SCe, 0xff7e1880),
```

This part includes a rename of the RTL\_GIGA\_MAC\_VER\_XYZ which induces no user-visible changes beside a new "RTL8169SC/8110SC" identifier.

If I correctly read the patch, it should go along the 0x98000000 mask in rtl8169\_get\_mac\_version.

[...]

```
static struct pci_device_id rtl8169_pci_tbl[] = {
- { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8129), 0, 0, RTL_CFG_0 },
- { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8136), 0, 0, RTL_CFG_2 },
- { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8167), 0, 0, RTL_CFG_0 },
- { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8168), 0, 0, RTL_CFG_2 },
- { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8169), 0, 0, RTL_CFG_0 },
- { PCI_DEVICE(PCI_VENDOR_ID_DLINK, 0x4300), 0, 0, RTL_CFG_0 },
- { PCI_DEVICE(0x1259, 0xc107), 0, 0, RTL_CFG_0 },
- { PCI_DEVICE(0x16ec, 0x0116), 0, 0, RTL_CFG_0 },
- { PCI_VENDOR_ID_LINKSYS, 0x1032,
- PCI_ANY_ID, 0x0024, 0, 0, RTL_CFG_0 },
+ { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8169), },
+ { PCI_DEVICE(PCI_VENDOR_ID_REALTEK, 0x8167), },
}
```

The current driver is intended to handle the whole set of PCI IDs which would be removed by the patch. Though some combination of chipset and motherboard do not work as expected, the gigabit chipsets have been reported to work.

Please elaborate if there is a good reason to remove any ID.

```
@@ -230,8 +232,9 @@ static struct {
} debug = { -1 };

enum RTL8169_registers {
- MAC0 = 0, /* Ethernet hardware address. */
- MAR0 = 8, /* Multicast filter. */
+ MAC0 = 0x00, /* Ethernet hardware address. */
+ MAC4 = 0x04,
+ MAR0 = 0x08, /* Multicast filter. */
}
```

New register. Ok.

```
CounterAddrLow = 0x10,
CounterAddrHigh = 0x14,
TxDescStartAddrLow = 0x20,
@@ -260,6 +263,7 @@ enum RTL8169_registers {
TBI_ANAR = 0x68,
```

```
TBI_LPAR = 0x6A,  
PHYstatus = 0x6C,  
+ Offset_7Ch = 0x7C,
```

Can you provide a more descriptive name for this register ?

```
RxMaxSize = 0xDA,  
CPlusCmd = 0xE0,  
IntrMitigate = 0xE2,  
@@ -287,11 +291,10 @@ enum RTL8169_register_content {  
RxOK = 0x01,  
  
/* RxStatusDesc */  
- RxFOVF = (1 << 23),  
- RxRWT = (1 << 22),  
- RxRES = (1 << 21),  
- RxRUNT = (1 << 20),  
- RxCRC = (1 << 19),  
+ RxRES = 0x00200000,  
+ RxCRC = 0x00080000,  
+ RxRUNT = 0x00100000,  
+ RxRWT = 0x00400000,
```

This part removes RxFOVF. Please elaborate if there is a reason to do so.

```
@@ -322,6 +325,10 @@ enum RTL8169_register_content {  
/* Config1 register p.24 */  
PMEEnable = (1 << 0), /* Power Management Enable */  
  
+ /* Config2 register p.26 */  
+ PCI_Clock_66MHz = 0x01,  
+ PCI_Clock_33MHz = 0x00,  
+
```

New bits definition (p.25 of datasheet rev. 1.12 here). Ok

```
/* Config3 register p.25 */  
MagicPacket = (1 << 5), /* Wake up when receives a Magic Packet */  
LinkUp = (1 << 4), /* Wake up when the cable connection is re-established */  
@@ -357,6 +364,31 @@ enum RTL8169_register_content {  
LinkStatus = 0x02,  
FullDup = 0x01,  
  
+ /* GIGABIT_PHY_registers */
```

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

```
+ PHY_CTRL_REG = 0,  
+ PHY_STAT_REG = 1,  
+ PHY_AUTO_NEGO_REG = 4,  
+ PHY_1000_CTRL_REG = 9,  
+  
+ /* GIGABIT_PHY_REG_BIT */  
+ PHY_Restart_Auto_Nego = 0x0200,  
+ PHY_Enable_Auto_Nego = 0x1000,  
+  
+ /* PHY_STAT_REG = 1 */  
+ PHY_Auto_Neco_Comp = 0x0020,  
+  
+ /* PHY_AUTO_NEGO_REG = 4 */  
+ PHY_Cap_10_Half = 0x0020,  
+ PHY_Cap_10_Full = 0x0040,  
+ PHY_Cap_100_Half = 0x0080,  
+ PHY_Cap_100_Full = 0x0100,  
+  
+ /* PHY_1000_CTRL_REG = 9 */  
+ PHY_Cap_1000_Full = 0x0200,  
+ PHY_Cap_1000_Half = 0x0100,  
+  
+ PHY_Cap_Null = 0x0,  
+
```

This part duplicates bits already defined in include/linu/mii.h.

I will assume that it is due to a diff against an ageing version of the driver included in the kernel.

```
/* _MediaType */  
_10_Half = 0x01,  
_10_Full = 0x02,  
@@ -440,13 +472,13 @@ struct rtl8169_private {  
    dma_addr_t RxPhyAddr;  
    struct sk_buff *Rx_skbuff[NUM_RX_DESC]; /* Rx data buffers */  
    struct ring_info tx_skb[NUM_TX_DESC]; /* Tx data buffers */  
    - unsigned align;  
    unsigned rx_buf_sz;  
    struct timer_list timer;  
    u16 cp_cmd;  
    u16 intr_mask;  
    int phy_auto_nego_reg;  
    int phy_1000_ctrl_reg;  
    + uint8_t mac_addr[NODE_ADDRESS_SIZE];
```

Two points here:

1. the driver uniformly uses u{8/16/32} types. Please follow the current

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

style and avoid to add uint{8/16/32}\_t things.

2. does this new field bring something that struct net\_device.dev\_addr does not ?

```
#ifdef CONFIG_R8169_VLAN
struct vlan_group *vlgrp;
#endif
@@ -461,8 +493,14 @@ struct rtl8169_private {

MODULE_AUTHOR("Realtek and the Linux r8169 crew <netdev@xxxxxxxxxxxxxxxx>");
MODULE_DESCRIPTION("RealTek RTL-8169 Gigabit Ethernet driver");
-module_param_array(media, int, &num_media, 0);
-MODULE_PARM_DESC(media, "force phy operation. Deprecated by ethtool (8).");
+
+module_param_array(speed, int, &num_speed, 0);
+MODULE_PARM_DESC(speed, "force phy operation. Deprecated by ethtool (8).");
+module_param_array(duplex, int, &num_duplex, 0);
+MODULE_PARM_DESC(duplex, "force phy operation. Deprecated by ethtool
(8).");
+module_param_array(autoneg, int, &num_autoneg, 0);
+MODULE_PARM_DESC(autoneg, "force phy operation. Deprecated by ethtool
(8).");
+
module_param(rx_copybreak, int, 0);
MODULE_PARM_DESC(rx_copybreak, "Copy breakpoint for copy-only-tiny-frames");
module_param(use_dac, int, 0);
@@ -474,7 +512,12 @@ MODULE_VERSION(RTL8169_VERSION);

static int rtl8169_open(struct net_device *dev);
static int rtl8169_start_xmit(struct sk_buff *skb, struct net_device *dev);
+#if LINUX_VERSION_CODE < KERNEL_VERSION(2,6,19)
+static irqreturn_t rtl8169_interrupt(int irq, void *dev_instance,
struct pt_regs *regs);
+#else
static irqreturn_t rtl8169_interrupt(int irq, void *dev_instance);
+
+#endif
static int rtl8169_init_ring(struct net_device *dev);
static void rtl8169_hw_start(struct net_device *dev);
static int rtl8169_close(struct net_device *dev);
@@ -485,6 +528,8 @@ static int rtl8169_rx_interrupt(struct net_device *dev,
void __iomem *);
static int rtl8169_change_mtu(struct net_device *dev, int new_mtu);
static void rtl8169_down(struct net_device *dev);
+static int rtl8169_set_mac_address(struct net_device *dev, void *p);
+void rtl8169_rar_set(struct rtl8169_private *tp, uint8_t *addr,
uint32_t index);
```

The code can be reordered so that the forward declarations are not needed.

See below.

[...]

```
@@ -982,16 +1065,13 @@ static void rtl8169_gset_xmii(struct net
else if (status & _10bps)
cmd->speed = SPEED_10;

- if (status & TxFlowCtrl)
- cmd->advertising |= ADVERTISED_Asym_Pause;
- if (status & RxFlowCtrl)
- cmd->advertising |= ADVERTISED_Pause;
-
```

Looking at description of the PHYStatus register at p.30 in the rev. 1.21 of the datasheet, the flow control advertised before the patch seems correct. Is there a specific reason to remove this code ?

[...]

```
-static void rtl8169_get_mac_version(struct rtl8169_private *tp, void
__iomem *ioaddr)
+static void
+rtl8169_get_mac_version(struct rtl8169_private *tp,
+ void __iomem *ioaddr)
{
const struct {
u32 mask;
int mac_version;
} mac_info[] = {
- { 0x38800000, RTL_GIGA_MAC_VER_15 },
- { 0x38000000, RTL_GIGA_MAC_VER_12 },
- { 0x34000000, RTL_GIGA_MAC_VER_13 },
- { 0x30800000, RTL_GIGA_MAC_VER_14 },
- { 0x30000000, RTL_GIGA_MAC_VER_11 },
- { 0x18000000, RTL_GIGA_MAC_VER_05 },
- { 0x10000000, RTL_GIGA_MAC_VER_04 },
- { 0x04000000, RTL_GIGA_MAC_VER_03 },
- { 0x00800000, RTL_GIGA_MAC_VER_02 },
- { 0x00000000, RTL_GIGA_MAC_VER_01 } /* Catch-all */
+ { 0x18000000, RTL_GIGA_MAC_VER_8110SCd },
+ { 0x98000000, RTL_GIGA_MAC_VER_8110SCe },
+ { 0x1 << 28, RTL_GIGA_MAC_VER_8169SB },
+ { 0x1 << 26, RTL_GIGA_MAC_VER_8110S },
+ { 0x1 << 23, RTL_GIGA_MAC_VER_8169S },
+ { 0x00000000, RTL_GIGA_MAC_VER_8169 } /* Catch-all */
```

Removal apart, this change suggests:

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

(after patch) (before patch)

```
RTL_GIGA_MAC_VER_8169 <-> RTL_GIGA_MAC_VER_01
RTL_GIGA_MAC_VER_8169S <-> RTL_GIGA_MAC_VER_02
RTL_GIGA_MAC_VER_8110S <-> RTL_GIGA_MAC_VER_03
RTL_GIGA_MAC_VER_8169SB <-> RTL_GIGA_MAC_VER_04
RTL_GIGA_MAC_VER_8110SCd <-> RTL_GIGA_MAC_VER_05
RTL_GIGA_MAC_VER_8110SCe <-> *new*
```

```
}, *p = mac_info;
u32 reg;
```

```
- reg = RTL_R32(TxConfig) & 0x7c800000;
+ reg = RTL_R32(TxConfig) & 0xfc800000;
```

Change of mask to go along the new 0x98000000 id. Ok.

[...]

```
@@ -1272,7 +1386,7 @@ static void rtl8169_hw_phy_config(struct
rtl8169_print_mac_version(tp);
rtl8169_print_phy_version(tp);
```

```
- if (tp->mac_version <= RTL_GIGA_MAC_VER_01)
+ if (tp->mac_version <= RTL_GIGA_MAC_VER_8169)
```

Rename. Ok.

```
return;
if (tp->phy_version >= RTL_GIGA_PHY_VER_H)
return;
@@ -1282,7 +1396,7 @@ static void rtl8169_hw_phy_config(struct
```

```
/* Shazam ! */
```

```
- if (tp->mac_version == RTL_GIGA_MAC_VER_04) {
+ if (tp->mac_version == RTL_GIGA_MAC_VER_8169SB) {
```

Rename. Ok.

[...]

```
@@ -1352,24 +1467,26 @@ out_unlock:
spin_unlock_irq(&tp->lock);
}
```

```
-static inline void rtl8169_delete_timer(struct net_device *dev)
```

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

```
+static inline void
+rtl8169_delete_timer(struct net_device *dev)
{
struct rtl8169_private *tp = netdev_priv(dev);
struct timer_list *timer = &tp->timer;

- if ((tp->mac_version <= RTL_GIGA_MAC_VER_01) ||
+ if ((tp->mac_version <= RTL_GIGA_MAC_VER_8169) ||
(tp->phy_version >= RTL_GIGA_PHY_VER_H))
return;

del_timer_sync(timer);
}

-static inline void rtl8169_request_timer(struct net_device *dev)
+static inline void
+rtl8169_request_timer(struct net_device *dev)
{
struct rtl8169_private *tp = netdev_priv(dev);
struct timer_list *timer = &tp->timer;

- if ((tp->mac_version <= RTL_GIGA_MAC_VER_01) ||
+ if ((tp->mac_version <= RTL_GIGA_MAC_VER_8169) ||
```

Rename. Ok.

[...]

```
+/**
+ * rtl8169_rar_set - Puts an ethernet address into a receive address
register.
+ *
+ * tp - The private data structure for driver
+ * addr - Address to put into receive address register
+ * index - Receive address register to write
+
+ *****/
+void
+rtl8169_rar_set(struct rtl8169_private *tp,
+ uint8_t *addr,
+ uint32_t index)
+{
+ void __iomem *ioaddr = tp->mmio_addr;
+ uint32_t rar_low = 0;
+ uint32_t rar_high = 0;
+
+ rar_low = ((uint32_t) addr[0] |
+ ((uint32_t) addr[1] << 8) |
+ ((uint32_t) addr[2] << 16) |
+ ((uint32_t) addr[3] << 24));
```

```
- if (!netif_running(dev))
- return -ENODEV;
+ rar_high = ((uint32_t) addr[4] |
+ ((uint32_t) addr[5] << 8));

- switch (cmd) {
- case SIOCGMIIPHY:
- data->phy_id = 32; /* Internal PHY */
- return 0;
-
- case SIOCGMIIREG:
- data->val_out = mdio_read(tp->mmio_addr, data->reg_num & 0x1f);
- return 0;
-
- case SIOCSMIIREG:
- if (!capable(CAP_NET_ADMIN))
- return -EPERM;
- mdio_write(tp->mmio_addr, data->reg_num & 0x1f, data->val_in);
- return 0;
- }
- return -EOPNOTSUPP;
+ RTL_W8(Cfg9346, Cfg9346_Unlock);
+ RTL_W32(MAC0, rar_low);
+ RTL_W32(MAC4, rar_high);
+ RTL_W8(Cfg9346, Cfg9346_Lock);
}
```

rtl8169\_set\_mac\_address() part. Ok.

[...]

```
@@ -1801,102 +1965,95 @@ static void
rtl8169_hw_start(struct net_device *dev)
{
struct rtl8169_private *tp = netdev_priv(dev);
- void __iomem *ioaddr = tp->mmio_addr;
struct pci_dev *pdev = tp->pci_dev;
+ void __iomem *ioaddr = tp->mmio_addr;
u32 i;

/* Soft reset the chip. */
RTL_W8(ChipCmd, CmdReset);

/* Check that the chip has finished the reset. */
- for (i = 100; i > 0; i--) {
+ for (i = 1000; i > 0; i--) {
if ((RTL_R8(ChipCmd) & CmdReset) == 0)
break;
- msleep_interruptible(1);
```

```
+ udelay(10);
}

- if (tp->mac_version == RTL_GIGA_MAC_VER_13) {
- pci_write_config_word(pdev, 0x68, 0x00);
- pci_write_config_word(pdev, 0x69, 0x08);
- }
-
- /* Undocumented stuff. */
- if (tp->mac_version == RTL_GIGA_MAC_VER_05) {
- u16 cmd;
-
- /* Realtek's r1000_n.c driver uses '&& 0x01' here. Well... */
- if ((RTL_R8(Config2) & 0x07) & 0x01)
- RTL_W32(0x7c, 0x0007ffff);
-
- RTL_W32(0x7c, 0x0007ff00);
-
- pci_read_config_word(pdev, PCI_COMMAND, &cmd);
- cmd = cmd & 0xef;
- pci_write_config_word(pdev, PCI_COMMAND, cmd);
- }
-
-
RTL_W8(Cfg9346, Cfg9346_Unlock);
+
RTL_W8(EarlyTxThres, EarlyTxThld);
```

Lots of changes here...

```
/* Low hurts. Let's disable the filtering. */
RTL_W16(RxMaxSize, 16383);

- /* Set Rx Config register */
- i = rtl8169_rx_config |
- (RTL_R32(RxConfig) & rtl_chip_info[tp->chipset].RxConfigMask);
- RTL_W32(RxConfig, i);
+ tp->cp_cmd |= RTL_R16(CPlusCmd);
+ RTL_W16(CPlusCmd, tp->cp_cmd);

- /* Set DMA burst size and Interframe Gap Time */
- RTL_W32(TxConfig, (TX_DMA_BURST << TxDMAShift) |
- (InterFrameGap << TxInterFrameGapShift));
+ RTL_W16(CPlusCmd, RTL_R16(CPlusCmd) | PCIMulRW);
+ pci_write_config_byte(pdev, PCI_CACHE_LINE_SIZE, 0x08);

- tp->cp_cmd |= RTL_R16(CPlusCmd) | PCIMulRW;
+ pci_write_config_byte(pdev, PCI_LATENCY_TIMER, 0x40);
```

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

```
- if ((tp->mac_version == RTL_GIGA_MAC_VER_02) ||
- (tp->mac_version == RTL_GIGA_MAC_VER_03)) {
+ if (RTL_R8(Config2) & PCI_Clock_66MHz) {
+ if (tp->mac_version == RTL_GIGA_MAC_VER_8110SCd)
+ RTL_W32(Offset_7Ch, 0x000FFFFFF);
+ else if (tp->mac_version == RTL_GIGA_MAC_VER_8110SCe)
+ RTL_W32(Offset_7Ch, 0x00FFFFFFF);
+ } else {
+ if (tp->mac_version == RTL_GIGA_MAC_VER_8110SCd)
+ RTL_W32(Offset_7Ch, 0x00FFF00);
+ else if (tp->mac_version == RTL_GIGA_MAC_VER_8110SCe)
+ RTL_W32(Offset_7Ch, 0x00FFF00);
```

Undocumented magic. Ok.

[...]

```
@@ -2529,6 +2726,17 @@ rtl8169_rx_interrupt(struct net_device *
tp->stats.rx_bytes += pkt_size;
tp->stats.rx_packets++;
}
+
+ //Work around for AMD plateform
+ if (((desc->opts2 & 0xFFFE000) != 0) &&
+ (tp->mac_version == RTL_GIGA_MAC_VER_8110SCd)) {
+ printk("%s: vlan_tag:%x\n", dev->name, desc->opts2);
+
+ desc->opts2 = 0;
+ cur_rx = cur_rx + 2;
+ } else {
+ cur_rx++;
+ }
```

Quirk. Ok.

Patch against the current in-kernel driver is included below.  
Feel free to comment if I have neglected something or send a  
Signed-off-by: line if the changes seems fine.

I have tried to sync the registers init sequence in rtl8169\_hw\_start  
with the code that you sent. I have kept a few parts inherited from  
the code in v1.05 of the r1000 driver (which was reported to work  
rather well despite some strange code).

Merge of r8169 6.001.00.

Signed-off-by: Francois Romieu <romieu@xxxxxxxxxxxx>

-----  
drivers/net/r8169.c | 112 ++++++

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

1 files changed, 74 insertions(+), 38 deletions(-)

```
diff --git a/drivers/net/r8169.c b/drivers/net/r8169.c
index fe03cd0..caaf15a 100644
--- a/drivers/net/r8169.c
+++ b/drivers/net/r8169.c
@@ -123,8 +123,8 @@ static const int multicast_filter_limit = 32;
#define MAC_ADDR_LEN 6

#define RX_FIFO_THRESH 7 /* 7 means NO threshold, Rx buffer level before first PCI xfer. */
-#define RX_DMA_BURST 6 /* Maximum PCI burst, '6' is 1024 */
-#define TX_DMA_BURST 6 /* Maximum PCI burst, '6' is 1024 */
+#define RX_DMA_BURST 7 /* Maximum PCI burst, '7' is unlimited */
+#define TX_DMA_BURST 7 /* Maximum PCI burst, '6' is 1024 */
#define EarlyTxThld 0x3F /* 0x3F means NO early transmit */
#define RxPacketMaxSize 0x3FE8 /* 16K - 1 - ETH_HLEN - VLAN - CRC... */
#define SafeMtu 0x1c20 /* ... actually life sucks beyond ~7k */
@@ -150,11 +150,12 @@ static const int multicast_filter_limit = 32;
#define RTL_R32(reg) ((unsigned long) readl (ioaddr + (reg)))

enum mac_version {
- RTL_GIGA_MAC_VER_01 = 0x00,
- RTL_GIGA_MAC_VER_02 = 0x01,
- RTL_GIGA_MAC_VER_03 = 0x02,
- RTL_GIGA_MAC_VER_04 = 0x03,
- RTL_GIGA_MAC_VER_05 = 0x04,
+ RTL_GIGA_MAC_VER_01 = 0x01, // 8169
+ RTL_GIGA_MAC_VER_02 = 0x02, // 8169S
+ RTL_GIGA_MAC_VER_03 = 0x03, // 8110S
+ RTL_GIGA_MAC_VER_04 = 0x04, // 8169SB
+ RTL_GIGA_MAC_VER_05 = 0x05, // 8169SCd
+ RTL_GIGA_MAC_VER_06 = 0x06, // 8169SCe
RTL_GIGA_MAC_VER_11 = 0x0b,
RTL_GIGA_MAC_VER_12 = 0x0c,
RTL_GIGA_MAC_VER_13 = 0x0d,
@@ -184,6 +185,7 @@ static const struct {
_R("RTL8169s/8110s", RTL_GIGA_MAC_VER_03, 0xff7e1880),
_R("RTL8169sb/8110sb", RTL_GIGA_MAC_VER_04, 0xff7e1880),
_R("RTL8169sc/8110sc", RTL_GIGA_MAC_VER_05, 0xff7e1880),
+ _R("RTL8169sc/8110sc", RTL_GIGA_MAC_VER_06, 0xff7e1880),
_R("RTL8168b/8111b", RTL_GIGA_MAC_VER_11, 0xff7e1880), // PCI-E
_R("RTL8168b/8111b", RTL_GIGA_MAC_VER_12, 0xff7e1880), // PCI-E
_R("RTL8101e", RTL_GIGA_MAC_VER_13, 0xff7e1880), // PCI-E 8139
@@ -323,6 +325,10 @@ enum RTL8169_register_content {
/* Config1 register p.24 */
PMEnable = (1 << 0), /* Power Management Enable */

+ /* Config2 register p. 25 */
+ PCI_Clock_66MHz = 0x01,
+ PCI_Clock_33MHz = 0x00,
+

```

```

/* Config3 register p.25 */
MagicPacket = (1 << 5), /* Wake up when receives a Magic Packet */
LinkUp = (1 << 4), /* Wake up when the cable connection is re-established */
@@ -1173,15 +1179,16 @@ static void rtl8169_get_mac_version(struct rtl8169_private *tp,
{ 0x34000000, RTL_GIGA_MAC_VER_13 },
{ 0x30800000, RTL_GIGA_MAC_VER_14 },
{ 0x30000000, RTL_GIGA_MAC_VER_11 },
- { 0x18000000, RTL_GIGA_MAC_VER_05 },
- { 0x10000000, RTL_GIGA_MAC_VER_04 },
- { 0x04000000, RTL_GIGA_MAC_VER_03 },
- { 0x00800000, RTL_GIGA_MAC_VER_02 },
- { 0x00000000, RTL_GIGA_MAC_VER_01 } /* Catch-all */
+ { 0x98000000, RTL_GIGA_MAC_VER_06 }, // 8110SCe
+ { 0x18000000, RTL_GIGA_MAC_VER_05 }, // 8110SCd
+ { 0x10000000, RTL_GIGA_MAC_VER_04 }, // 8169SB
+ { 0x04000000, RTL_GIGA_MAC_VER_03 }, // 8110S
+ { 0x00800000, RTL_GIGA_MAC_VER_02 }, // 8169S
+ { 0x00000000, RTL_GIGA_MAC_VER_01 } // 8169 - catch-all
}, *p = mac_info;
u32 reg;

- reg = RTL_R32(TxConfig) & 0x7c800000;
+ reg = RTL_R32(TxConfig) & 0xfc800000;
while ((reg & p->mask) != p->mask)
p++;
tp->mac_version = p->mac_version;
@@ -1420,6 +1427,16 @@ static void rtl8169_phy_reset(struct net_device *dev,
printk(KERN_ERR "%s: PHY reset failed.\n", dev->name);
}

+static bool rtl8169_match_asic_8169(unsigned int mac_version)
+{
+ return ((mac_version == RTL_GIGA_MAC_VER_01) ||
+ (mac_version == RTL_GIGA_MAC_VER_02) ||
+ (mac_version == RTL_GIGA_MAC_VER_03) ||
+ (mac_version == RTL_GIGA_MAC_VER_04) ||
+ (mac_version == RTL_GIGA_MAC_VER_05) ||
+ (mac_version == RTL_GIGA_MAC_VER_06)) ? true : false;
+}
+
static void rtl8169_init_phy(struct net_device *dev, struct rtl8169_private *tp)
{
void __iomem *ioaddr = tp->mmio_addr;
@@ -1434,10 +1451,10 @@ static void rtl8169_init_phy(struct net_device *dev, struct rtl8169_private *tp)
dprintk("Set MAC Reg C+CR Offset 0x82h = 0x01h\n");
RTL_W8(0x82, 0x01);

- if (tp->mac_version < RTL_GIGA_MAC_VER_03) {
- dprintk("Set PCI Latency=0x40\n");
- pci_write_config_byte(tp->pci_dev, PCI_LATENCY_TIMER, 0x40);
- }

```

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

```
+ pci_write_config_byte(tp->pci_dev, PCI_LATENCY_TIMER, 0x40);
+
+ if (rtl8169_match_asic_8169(tp->mac_version))
+ pci_write_config_byte(tp->pci_dev, PCI_CACHE_LINE_SIZE, 0x08);

if (tp->mac_version == RTL_GIGA_MAC_VER_02) {
dprintk("Set MAC Reg C+CR Offset 0x82h = 0x01h\n");
@@ -1857,6 +1874,30 @@ static void rtl8169_set_rx_tx_config_registers(struct rtl8169_private *tp)
(InterFrameGap << TxInterFrameGapShift));
}

+static void rtl8169_set_magic_reg(void __iomem *ioaddr, unsigned mac_version)
+{
+ struct {
+ u32 mac_version;
+ u32 clk;
+ u32 val;
+ } cfg2_info [] = {
+ { RTL_GIGA_MAC_VER_05, PCI_Clock_33MHz, 0x000fff00 }, // 8169SCd
+ { RTL_GIGA_MAC_VER_05, PCI_Clock_66MHz, 0x000fffff },
+ { RTL_GIGA_MAC_VER_06, PCI_Clock_33MHz, 0x00ffff00 }, // 8169SCe
+ { RTL_GIGA_MAC_VER_06, PCI_Clock_66MHz, 0x00ffffff }
+ }, *p = cfg2_info;
+ unsigned int i;
+ u32 clk;
+
+ clk = RTL_R8(Config2) & PCI_Clock_66MHz;
+ for (i = 0; i < ARRAY_SIZE(cfg2_info); i++) {
+ if ((p->mac_version == mac_version) && (p->clk == clk)) {
+ RTL_W32(0x7c, p->val);
+ break;
+ }
+ }
+
+ static void rtl8169_hw_start(struct net_device *dev)
+ {
+ struct rtl8169_private *tp = netdev_priv(dev);
+ @@ -1885,19 +1926,6 @@ static void rtl8169_hw_start(struct net_device *dev)
+ pci_write_config_word(pdev, 0x69, 0x08);
+ }

- /* Undocumented stuff. */
- if (tp->mac_version == RTL_GIGA_MAC_VER_05) {
- /* Realtek's r1000_n.c driver uses '&& 0x01' here. Well... */
- if ((RTL_R8(Config2) & 0x07) & 0x01)
- RTL_W32(0x7c, 0x0007ffff);
-
- RTL_W32(0x7c, 0x0007ff00);
-
- pci_read_config_word(pdev, PCI_COMMAND, &cmd);
```

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

```
- cmd = cmd & 0xef;
- pci_write_config_word(pdev, PCI_COMMAND, cmd);
- }
-
RTL_W8(Cfg9346, Cfg9346_Unlock);
if ((tp->mac_version == RTL_GIGA_MAC_VER_01) ||
(tp->mac_version == RTL_GIGA_MAC_VER_02) ||
@@ -1910,10 +1938,7 @@ static void rtl8169_hw_start(struct net_device *dev)
/* Low hurts. Let's disable the filtering. */
RTL_W16(RxMaxSize, 16383);

- if ((tp->mac_version == RTL_GIGA_MAC_VER_01) ||
- (tp->mac_version == RTL_GIGA_MAC_VER_02) ||
- (tp->mac_version == RTL_GIGA_MAC_VER_03) ||
- (tp->mac_version == RTL_GIGA_MAC_VER_04))
+ if (rtl8169_match_asic_8169(tp->mac_version))
rtl8169_set_rx_tx_config_registers(tp);

cmd = RTL_R16(CPlusCmd);
@@ -1921,6 +1946,8 @@ static void rtl8169_hw_start(struct net_device *dev)

tp->cp_cmd |= cmd | PCIMulRW;

+ rtl8169_set_magic_reg(ioaddr, tp->mac_version);
+
if ((tp->mac_version == RTL_GIGA_MAC_VER_02) ||
(tp->mac_version == RTL_GIGA_MAC_VER_03)) {
dprintk(KERN_INFO PFX "Set MAC Reg C+CR Offset 0xE0. "
@@ -1946,10 +1973,7 @@ static void rtl8169_hw_start(struct net_device *dev)
RTL_W32(RxDescAddrHigh, ((u64) tp->RxPhyAddr >> 32));
RTL_W32(RxDescAddrLow, ((u64) tp->RxPhyAddr & DMA_32BIT_MASK));

- if ((tp->mac_version != RTL_GIGA_MAC_VER_01) &&
- (tp->mac_version != RTL_GIGA_MAC_VER_02) &&
- (tp->mac_version != RTL_GIGA_MAC_VER_03) &&
- (tp->mac_version != RTL_GIGA_MAC_VER_04)) {
+ if (!rtl8169_match_asic_8169(tp->mac_version)) {
RTL_W8(ChipCmd, CmdTxEnb | CmdRxEnb);
rtl8169_set_rx_tx_config_registers(tp);
}
@@ -1969,6 +1993,10 @@ static void rtl8169_hw_start(struct net_device *dev)
/* Enable all known interrupts by setting the interrupt mask. */
RTL_W16(IntrMask, rtl8169_intr_mask);

+ if ((tp->mac_version == RTL_GIGA_MAC_VER_05) ||
+ (tp->mac_version == RTL_GIGA_MAC_VER_06))
+ RTL_W8(ChipCmd, CmdTxEnb | CmdRxEnb);
+
netif_start_queue(dev);
}
```

Re: [PATCH 2.6.19.2] r8169: support RTL8169SC/8110SC

```
@@ -2614,6 +2642,14 @@ static int rtl8169_rx_interrupt(struct net_device *dev,
tp->stats.rx_bytes += pkt_size;
tp->stats.rx_packets++;
}
+
+ /* Work around for AMD plateform. */
+ if ((desc->opts2 & 0xfffe000) &&
+ (tp->mac_version == RTL_GIGA_MAC_VER_05)) {
+ desc->opts2 = 0;
+ cur_rx++;
+ }
+
}
```

```
count = cur_rx - tp->cur_rx;
```

```
---
```

1.4.4.4

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>