

## [git patches] net driver updates for 2.6.29

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2008-11/msg01833.html>

---

- *From:* Jeff Garzik <[jeff@xxxxxxxxxx](mailto:jeff@xxxxxxxxxx)>
  - *Date:* Thu, 6 Nov 2008 01:50:59 -0500
- 

Please pull from 'davem-next' branch of  
master.kernel.org:/pub/scm/linux/kernel/git/jgarzik/netdev-2.6.git davem-next

to receive the following updates:

Documentation/networking/bonding.txt | 52 +  
MAINTAINERS | 6 +  
drivers/net/Kconfig | 15 +  
drivers/net/Makefile | 1 +  
drivers/net/bonding/Makefile | 3 +  
drivers/net/bonding/bond\_3ad.c | 326 ++++--  
drivers/net/bonding/bond\_3ad.h | 10 +-  
drivers/net/bonding/bond\_alb.c | 13 +-  
drivers/net/bonding/bond\_ipv6.c | 218 ++++  
drivers/net/bonding/bond\_main.c | 63 +-  
drivers/net/bonding/bond\_sysfs.c | 91 ++  
drivers/net/bonding/bonding.h | 35 +-  
drivers/net/ehca/ehca.h | 2 +-  
drivers/net/ehca/ehca\_qmr.c | 10 +-  
drivers/net/pcmcia/fmvj18x\_cs.c | 73 +-  
drivers/net/phy/smsc.c | 28 +  
drivers/net/sfc/Kconfig | 8 +  
drivers/net/sfc/Makefile | 1 +  
drivers/net/sfc/boards.c | 136 +++  
drivers/net/sfc/efx.c | 32 +-  
drivers/net/sfc/efx.h | 10 +  
drivers/net/sfc/enum.h | 4 +-  
drivers/net/sfc/ethtool.c | 15 +-  
drivers/net/sfc/falcon.c | 23 +-  
drivers/net/sfc/falcon\_hwdefs.h | 1 -  
drivers/net/sfc/mdio\_10g.c | 35 +  
drivers/net/sfc/mdio\_10g.h | 7 +  
drivers/net/sfc/mtd.c | 268 ++++  
drivers/net/sfc/net\_driver.h | 8 +  
drivers/net/sfc/sfe4001.c | 116 +-  
drivers/net/sfc/spi.h | 34 +-  
drivers/net/sfc/tenxpress.c | 18 +-  
drivers/net/sfc/workarounds.h | 2 +

[git patches] net driver updates for 2.6.29

```
drivers/net/sfc/xfp_phy.c | 9 +
drivers/net/smsc911x.c | 2091 +++++
drivers/net/smsc911x.h | 394 ++++++
drivers/net/usb/usbnet.c | 140 +-
include/linux/smsc911x.h | 42 +
include/linux/usb/usbnet.h | 6 +
include/net/ndisc.h | 14 +
net/ipv6/ndisc.c | 92 +-
41 files changed, 4144 insertions(+), 308 deletions(-)
create mode 100644 drivers/net/bonding/bond_ipv6.c
create mode 100644 drivers/net/sfc/mtd.c
create mode 100644 drivers/net/smsc911x.c
create mode 100644 drivers/net/smsc911x.h
create mode 100644 include/linux/smsc911x.h
```

Ben Hutchings (5):

- sfc: Correct address of gPXE boot configuration in EEPROM
- sfc: Clean up non-volatile memory partitioning
- sfc: Expose flash region storing boot code as MTD
- sfc: Use lm87 and lm90 drivers for board temperature/power monitoring
- sfc: Do not reset when hardware monitor detects a fault

Brian Haley (1):

- bonding: send IPv6 neighbor advertisement on failover

Hannes Hering (1):

- ehea: Fix some whitespace issues

Jay Vosburgh (2):

- bonding: Fix ALB mode to balance traffic on VLANs
- bonding: alternate agg selection policies for 802.3ad

Komuro (1):

- fmvj18x\_cs: write interrupt ack bit for lan and modem to work simultaneously.

Per Hallsmark (1):

- usbnet: enable more aggressive autosuspend

Steve Glendinning (1):

- SMSC LAN911x and LAN921x vendor driver

```
diff --git a/Documentation/networking/bonding.txt b/Documentation/networking/bonding.txt
index d733a42..5ede747 100644
--- a/Documentation/networking/bonding.txt
+++ b/Documentation/networking/bonding.txt
@@ -194,6 +194,48 @@ or, for backwards compatibility, the option value. E.g.,
```

The parameters are as follows:

```
+ad_select
+
```

## [git patches] net driver updates for 2.6.29

- + Specifies the 802.3ad aggregation selection logic to use. The possible values and their effects are:
  - + stable or 0
    - + The active aggregator is chosen by largest aggregate bandwidth.
    - + Reselection of the active aggregator occurs only when all slaves of the active aggregator are down or the active aggregator has no slaves.
    - + This is the default value.
  - + bandwidth or 1
    - + The active aggregator is chosen by largest aggregate bandwidth. Reselection occurs if:
      - + – A slave is added to or removed from the bond
      - + – Any slave's link state changes
      - + – Any slave's 802.3ad association state changes
      - + – The bond's administrative state changes to up
  - + count or 2
    - + The active aggregator is chosen by the largest number of ports (slaves). Reselection occurs as described under the "bandwidth" setting, above.
    - + The bandwidth and count selection policies permit failover of 802.3ad aggregations when partial failure of the active aggregator occurs. This keeps the aggregator with the highest availability (either in bandwidth or in number of ports) active at all times.
    - + This option was added in bonding version 3.4.0.
- + arp\_interval

Specifies the ARP link monitoring frequency in milliseconds.

@@ -551,6 +593,16 @@ num\_grat\_arp

affects only the active-backup mode. This option was added for bonding version 3.3.0.

+num\_unsol\_na

- + Specifies the number of unsolicited IPv6 Neighbor Advertisements to be issued after a failover event. One unsolicited NA is issued

[git patches] net driver updates for 2.6.29

+ immediately after the failover.  
+  
+ The valid range is 0 – 255; the default value is 1. This option  
+ affects only the active–backup mode. This option was added for  
+ bonding version 3.4.0.  
+  
primary

A string (eth0, eth2, etc) specifying which slave is the  
diff --git a/MAINTAINERS b/MAINTAINERS  
index 129117f..5d951f3 100644  
--- a/MAINTAINERS  
+++ b/MAINTAINERS  
@@ -3853,6 +3853,12 @@ M: mhoffman@xxxxxxxxxxxxxx  
L: lm-sensors@xxxxxxxxxxxxxx  
S: Maintained

+SMSC911x ETHERNET DRIVER  
+P: Steve Glendinning  
+M: steve.glendinning@xxxxxxxxxx  
+L: netdev@xxxxxxxxxxxxxx  
+S: Supported

+  
SMX UIO Interface  
P: Ben Nizette  
M: bn@xxxxxxxxxxxxxx  
diff --git a/drivers/net/Kconfig b/drivers/net/Kconfig  
index 0f3e6b2..74a18a7 100644  
--- a/drivers/net/Kconfig  
+++ b/drivers/net/Kconfig  
@@ -61,6 +61,7 @@ config DUMMY  
config BONDING  
tristate "Bonding driver support"  
depends on INET  
+ depends on IPV6 || IPV6=n  
---help---

Say 'Y' or 'M' if you wish to be able to 'bond' multiple Ethernet  
Channels together. This is called 'Etherchannel' by Cisco,  
@@ -978,6 +979,20 @@ config SMC911X  
called smc911x. If you want to compile it as a module, say M  
here and read <file:Documentation/kbuild/modules.txt>

+config SMSC911X  
+ tristate "SMSC LAN911x/LAN921x families embedded ethernet support"  
+ depends on ARM || SUPERH  
+ select CRC32  
+ select MII  
+ select PHYLIB  
+ ---help---  
+ Say Y here if you want support for SMSC LAN911x and LAN921x families  
+ of ethernet controllers.

## [git patches] net driver updates for 2.6.29

```
+
+ To compile this driver as a module, choose M here and read
+ <file:Documentation/networking/net-modules.txt>. The module
+ will be called smsc911x.
+
config NET_VENDOR_RACAL
bool "Racal-Interlan (Micom) NI cards"
depends on ISA
diff --git a/drivers/net/Makefile b/drivers/net/Makefile
index 657c47b..e06829a 100644
--- a/drivers/net/Makefile
+++ b/drivers/net/Makefile
@@ -219,6 +219,7 @@ obj-$(CONFIG_S2IO) += s2io.o
obj-$(CONFIG_MYRI10GE) += myri10ge/
obj-$(CONFIG_SMC91X) += smc91x.o
obj-$(CONFIG_SMC911X) += smc911x.o
+obj-$(CONFIG_SMSC911X) += smsc911x.o
obj-$(CONFIG_BFIN_MAC) += bfin_mac.o
obj-$(CONFIG_DM9000) += dm9000.o
obj-$(CONFIG_PASEMI_MAC) += pasemi_mac_driver.o
diff --git a/drivers/net/bonding/Makefile b/drivers/net/bonding/Makefile
index 5cdae2b..6f9c6fa 100644
--- a/drivers/net/bonding/Makefile
+++ b/drivers/net/bonding/Makefile
@@ -6,3 +6,6 @@ obj-$(CONFIG_BONDING) += bonding.o

bonding-objs := bond_main.o bond_3ad.o bond_alb.o bond_sysfs.o

+ipv6-$(subst m,y,$(CONFIG_IPV6)) += bond_ipv6.o
+bonding-objs += $(ipv6-y)
+
diff --git a/drivers/net/bonding/bond_3ad.c b/drivers/net/bonding/bond_3ad.c
index 6106660..ba1372f 100644
--- a/drivers/net/bonding/bond_3ad.c
+++ b/drivers/net/bonding/bond_3ad.c
@@ -27,6 +27,7 @@
#include <linux/netdevice.h>
#include <linux/spinlock.h>
#include <linux/ethtool.h>
+#include <linux/etherdevice.h>
#include <linux/if_bonding.h>
#include <linux/pkt_sched.h>
#include <net/net_namespace.h>
@@ -236,6 +237,17 @@ static inline struct aggregator *__get_next_agg(struct aggregator *aggregator)
return &(SLAVE_AD_INFO(slave->next).aggregator);
}

+/*
+ * __agg_has_partner
+ *
+ * Return nonzero if aggregator has a partner (denoted by a non-zero ether
```

## [git patches] net driver updates for 2.6.29

```
+ * address for the partner). Return 0 if not.
+ */
+static inline int __agg_has_partner(struct aggregator *agg)
+{
+ return !is_zero_ether_addr(agg->partner_system.mac_addr_value);
+}
+
+/**
+ * __disable_port - disable the port's slave
+ * @port: the port we're looking at
+ @@ -274,14 +286,14 @@ static inline int __port_is_enabled(struct port *port)
+ * __get_agg_selection_mode - get the aggregator selection mode
+ * @port: the port we're looking at
+ *
+ - * Get the aggregator selection mode. Can be %BANDWIDTH or %COUNT.
+ + * Get the aggregator selection mode. Can be %STABLE, %BANDWIDTH or %COUNT.
+ */
+static inline u32 __get_agg_selection_mode(struct port *port)
+{
+ struct bonding *bond = __get_bond_by_port(port);
+
+ if (bond == NULL) {
+ - return AD_BANDWIDTH;
+ + return BOND_AD_STABLE;
+ }
+
+ return BOND_AD_INFO(bond).agg_select_mode;
+ @@ -1414,9 +1426,82 @@ static void ad_port_selection_logic(struct port *port)
+ // else set ready=FALSE in all aggregator's ports
+ __set_agg_ports_ready(port->aggregator, __agg_ports_are_ready(port->aggregator));
+
+ - if (!__check_agg_selection_timer(port) && (aggregator = __get_first_agg(port))) {
+ - ad_agg_selection_logic(aggregator);
+ + aggregator = __get_first_agg(port);
+ + ad_agg_selection_logic(aggregator);
+ +}
+ +
+ +/**
+ + * Decide if "agg" is a better choice for the new active aggregator that
+ + * the current best, according to the ad_select policy.
+ + */
+ +static struct aggregator *ad_agg_selection_test(struct aggregator *best,
+ + struct aggregator *curr)
+ +{
+ + /**
+ + * 0. If no best, select current.
+ + *
+ + * 1. If the current agg is not individual, and the best is
+ + * individual, select current.
+ + *
+ + * 2. If current agg is individual and the best is not, keep best.
```

## [git patches] net driver updates for 2.6.29

```
+ *
+ * 3. Therefore, current and best are both individual or both not
+ * individual, so:
+ *
+ * 3a. If current agg partner replied, and best agg partner did not,
+ * select current.
+ *
+ * 3b. If current agg partner did not reply and best agg partner
+ * did reply, keep best.
+ *
+ * 4. Therefore, current and best both have partner replies or
+ * both do not, so perform selection policy:
+ *
+ * BOND_AD_COUNT: Select by count of ports. If count is equal,
+ * select by bandwidth.
+ *
+ * BOND_AD_STABLE, BOND_AD_BANDWIDTH: Select by bandwidth.
+ */
+ if (!best)
+ return curr;
+
+ if (!curr->is_individual && best->is_individual)
+ return curr;
+
+ if (curr->is_individual && !best->is_individual)
+ return best;
+
+ if (__agg_has_partner(curr) && !__agg_has_partner(best))
+ return curr;
+
+ if (!__agg_has_partner(curr) && __agg_has_partner(best))
+ return best;
+
+ switch (__get_agg_selection_mode(curr->lag_ports)) {
+ case BOND_AD_COUNT:
+ if (curr->num_of_ports > best->num_of_ports)
+ return curr;
+
+ if (curr->num_of_ports < best->num_of_ports)
+ return best;
+
+ /*FALLTHROUGH*/
+ case BOND_AD_STABLE:
+ case BOND_AD_BANDWIDTH:
+ if (__get_agg_bandwidth(curr) > __get_agg_bandwidth(best))
+ return curr;
+
+ break;
+
+ default:
+ printk(KERN_WARNING DRV_NAME
```

## [git patches] net driver updates for 2.6.29

```
+ ": %s: Impossible agg select mode %d\n",
+ curr->slave->dev->master->name,
+ __get_agg_selection_mode(curr->lag_ports));
+ break;
}
+
+ return best;
}

/**
@@ -1424,156 +1509,138 @@ static void ad_port_selection_logic(struct port *port)
* @aggregator: the aggregator we're looking at
*
* It is assumed that only one aggregator may be selected for a team.
- * The logic of this function is to select (at first time) the aggregator with
- * the most ports attached to it, and to reselect the active aggregator only if
- * the previous aggregator has no more ports related to it.
+ *
+ * The logic of this function is to select the aggregator according to
+ * the ad_select policy:
+ *
+ * BOND_AD_STABLE: select the aggregator with the most ports attached to
+ * it, and to reselect the active aggregator only if the previous
+ * aggregator has no more ports related to it.
+ *
+ * BOND_AD_BANDWIDTH: select the aggregator with the highest total
+ * bandwidth, and reselect whenever a link state change takes place or the
+ * set of slaves in the bond changes.
+ *
+ * BOND_AD_COUNT: select the aggregator with largest number of ports
+ * (slaves), and reselect whenever a link state change takes place or the
+ * set of slaves in the bond changes.
*
* FIXME: this function MUST be called with the first agg in the bond, or
* __get_active_agg() won't work correctly. This function should be better
* called with the bond itself, and retrieve the first agg from it.
*/
-static void ad_agg_selection_logic(struct aggregator *aggregator)
+static void ad_agg_selection_logic(struct aggregator *agg)
{
- struct aggregator *best_aggregator = NULL, *active_aggregator = NULL;
- struct aggregator *last_active_aggregator = NULL, *origin_aggregator;
+ struct aggregator *best, *active, *origin;
struct port *port;
- u16 num_of_aggs=0;

- origin_aggregator = aggregator;
+ origin = agg;

- //get current active aggregator
- last_active_aggregator = __get_active_agg(aggregator);
```

## [git patches] net driver updates for 2.6.29

```
+ active = __get_active_agg(agg);
+ best = active;

- // search for the aggregator with the most ports attached to it.
do {
- // count how many candidate lag's we have
- if (aggregator->lag_ports) {
- num_of_aggs++;
- }
- if (aggregator->is_active && !aggregator->is_individual && // if current aggregator is the active
aggregator
- MAC_ADDRESS_COMPARE(&(aggregator->partner_system), &(null_mac_addr))) { // and partner
answers to 802.3ad PDUs
- if (aggregator->num_of_ports) { // if any ports attached to the current aggregator
- best_aggregator=NULL; // disregard the best aggregator that was chosen by now
- break; // stop the selection of other aggregator if there are any ports attached to this active aggregator
- } else { // no ports attached to this active aggregator
- aggregator->is_active = 0; // mark this aggregator as not active anymore
+ agg->is_active = 0;
+
+ if (agg->num_of_ports)
+ best = ad_agg_selection_test(best, agg);
+
+ } while ((agg = __get_next_agg(agg)));
+
+ if (best &&
+ __get_agg_selection_mode(best->lag_ports) == BOND_AD_STABLE) {
+ /*
+ * For the STABLE policy, don't replace the old active
+ * aggregator if it's still active (it has an answering
+ * partner) or if both the best and active don't have an
+ * answering partner.
+ */
+ if (active && active->lag_ports &&
+ active->lag_ports->is_enabled &&
+ (__agg_has_partner(active) ||
+ (!__agg_has_partner(active) && !__agg_has_partner(best)))) {
+ if (!(!active->actor_oper_aggregator_key &&
+ best->actor_oper_aggregator_key)) {
+ best = NULL;
+ active->is_active = 1;
+ }
+ }
- if (aggregator->num_of_ports) { // if any ports attached
- if (best_aggregator) { // if there is a candidte aggregator
- //The reasons for choosing new best aggregator:
- // 1. if current agg is NOT individual and the best agg chosen so far is individual OR
- // current and best aggs are both individual or both not individual, AND
- // 2a. current agg partner reply but best agg partner do not reply OR
- // 2b. current agg partner reply OR current agg partner do not reply AND best agg partner also do not reply
AND
AND
```

## [git patches] net driver updates for 2.6.29

```
- // current has more ports/bandwidth, or same amount of ports but current has faster ports, THEN
- // current agg become best agg so far
-
- //if current agg is NOT individual and the best agg chosen so far is individual change best_aggregator
- if (!aggregator->is_individual && best_aggregator->is_individual) {
- best_aggregator=aggregator;
- }
- // current and best aggs are both individual or both not individual
- else if ((aggregator->is_individual && best_aggregator->is_individual) ||
- (!aggregator->is_individual && !best_aggregator->is_individual)) {
- // current and best aggs are both individual or both not individual AND
- // current agg partner reply but best agg partner do not reply
- if ((MAC_ADDRESS_COMPARE(&(aggregator->partner_system), &(null_mac_addr)) &&
- !MAC_ADDRESS_COMPARE(&(best_aggregator->partner_system), &(null_mac_addr)))) {
- best_aggregator=aggregator;
- }
- // current agg partner reply OR current agg partner do not reply AND best agg partner also do not reply
- else if (! (!MAC_ADDRESS_COMPARE(&(aggregator->partner_system), &(null_mac_addr)) &&
- MAC_ADDRESS_COMPARE(&(best_aggregator->partner_system), &(null_mac_addr)))) {
- if ((__get_agg_selection_mode(aggregator->lag_ports) == AD_BANDWIDTH)&&
- (__get_agg_bandwidth(aggregator) > __get_agg_bandwidth(best_aggregator))) {
- best_aggregator=aggregator;
- } else if (__get_agg_selection_mode(aggregator->lag_ports) == AD_COUNT) {
- if (((aggregator->num_of_ports > best_aggregator->num_of_ports) &&
- (aggregator->actor_oper_aggregator_key & AD_SPEED_KEY_BITS))||
- ((aggregator->num_of_ports == best_aggregator->num_of_ports) &&
- ((u16)(aggregator->actor_oper_aggregator_key & AD_SPEED_KEY_BITS) >
- (u16)(best_aggregator->actor_oper_aggregator_key & AD_SPEED_KEY_BITS)))) {
- best_aggregator=aggregator;
- }
- }
- }
- } else {
- best_aggregator=aggregator;
- }
- }
- aggregator->is_active = 0; // mark all aggregators as not active anymore
- } while ((aggregator = __get_next_agg(aggregator)));
-
- // if we have new aggregator selected, don't replace the old aggregator if it has an answering partner,
- // or if both old aggregator and new aggregator don't have answering partner
- if (best_aggregator) {
- if (last_active_aggregator && last_active_aggregator->lag_ports &&
- last_active_aggregator->lag_ports->is_enabled &&
- (MAC_ADDRESS_COMPARE(&(last_active_aggregator->partner_system), &(null_mac_addr)) || //
- partner answers OR
- (!MAC_ADDRESS_COMPARE(&(last_active_aggregator->partner_system), &(null_mac_addr)) && //
- both old and new
- !MAC_ADDRESS_COMPARE(&(best_aggregator->partner_system), &(null_mac_addr)))) // partner do
- not answer
```

## [git patches] net driver updates for 2.6.29

```
- ) {
- // if new aggregator has link, and old aggregator does not, replace old aggregator.(do nothing)
- // -> don't replace otherwise.
- if (!(last_active_aggregator->actor_oper_aggregator_key &&
best_aggregator->actor_oper_aggregator_key)) {
- best_aggregator=NULL;
- last_active_aggregator->is_active = 1; // don't replace good old aggregator
+ }

- }
- }
+ if (best && (best == active)) {
+ best = NULL;
+ active->is_active = 1;
+ }

// if there is new best aggregator, activate it
- if (best_aggregator) {
- for (aggregator = __get_first_agg(best_aggregator->lag_ports);
- aggregator;
- aggregator = __get_next_agg(aggregator)) {
-
- dprintk("Agg=%d; Ports=%d; a key=%d; p key=%d; Indiv=%d; Active=%d\n",
- aggregator->aggregator_identifier, aggregator->num_of_ports,
- aggregator->actor_oper_aggregator_key, aggregator->partner_oper_aggregator_key,
- aggregator->is_individual, aggregator->is_active);
+ if (best) {
+ dprintk("best Agg=%d; P=%d; a k=%d; p k=%d; Ind=%d; Act=%d\n",
+ best->aggregator_identifier, best->num_of_ports,
+ best->actor_oper_aggregator_key,
+ best->partner_oper_aggregator_key,
+ best->is_individual, best->is_active);
+ dprintk("best ports %p slave %p %s\n",
+ best->lag_ports, best->slave,
+ best->slave ? best->slave->dev->name : "NULL");
+
+ for (agg = __get_first_agg(best->lag_ports); agg;
+ agg = __get_next_agg(agg)) {
+
+ dprintk("Agg=%d; P=%d; a k=%d; p k=%d; Ind=%d; Act=%d\n",
+ agg->aggregator_identifier, agg->num_of_ports,
+ agg->actor_oper_aggregator_key,
+ agg->partner_oper_aggregator_key,
+ agg->is_individual, agg->is_active);
+ }
+ }

// check if any partner replys
- if (best_aggregator->is_individual) {
- printk(KERN_WARNING DRV_NAME ": %s: Warning: No 802.3ad response from "
- "the link partner for any adapters in the bond\n",
- best_aggregator->slave->dev->master->name);
```

## [git patches] net driver updates for 2.6.29

```
- }
-
- // check if there are more than one aggregator
- if (num_of_aggs > 1) {
- dprintk("Warning: More than one Link Aggregation Group was "
- "found in the bond. Only one group will function in the bond\n");
+ if (best->is_individual) {
+ printk(KERN_WARNING DRV_NAME ": %s: Warning: No 802.3ad"
+ " response from the link partner for any"
+ " adapters in the bond\n",
+ best->slave->dev->master->name);
}

- best_aggregator->is_active = 1;
- dprintk("LAG %d choosed as the active LAG\n", best_aggregator->aggregator_identifier);
- dprintk("Agg=%d; Ports=%d; a key=%d; p key=%d; Indiv=%d; Active=%d\n",
- best_aggregator->aggregator_identifier, best_aggregator->num_of_ports,
- best_aggregator->actor_oper_aggregator_key, best_aggregator->partner_oper_aggregator_key,
- best_aggregator->is_individual, best_aggregator->is_active);
+ best->is_active = 1;
+ dprintk("LAG %d chosen as the active LAG\n",
+ best->aggregator_identifier);
+ dprintk("Agg=%d; P=%d; a k=%d; p k=%d; Ind=%d; Act=%d\n",
+ best->aggregator_identifier, best->num_of_ports,
+ best->actor_oper_aggregator_key,
+ best->partner_oper_aggregator_key,
+ best->is_individual, best->is_active);

// disable the ports that were related to the former active_aggregator
- if (last_active_aggregator) {
- for (port=last_active_aggregator->lag_ports; port; port=port->next_port_in_aggregator) {
+ if (active) {
+ for (port = active->lag_ports; port;
+ port = port->next_port_in_aggregator) {
__disable_port(port);
}
}
}

- // if the selected aggregator is of join individuals(partner_system is NULL), enable their ports
- active_aggregator = __get_active_agg(origin_aggregator);
+ /*
+ * if the selected aggregator is of join individuals
+ * (partner_system is NULL), enable their ports
+ */
+ active = __get_active_agg(origin);

- if (active_aggregator) {
- if (!MAC_ADDRESS_COMPARE(&(active_aggregator->partner_system), &(null_mac_addr))) {
- for (port=active_aggregator->lag_ports; port; port=port->next_port_in_aggregator) {
+ if (active) {
```

## [git patches] net driver updates for 2.6.29

```
+ if (!__agg_has_partner(active)) {
+ for (port = active->lag_ports; port;
+ port = port->next_port_in_aggregator) {
+ __enable_port(port);
+ }
+ }
+
+ if (origin->slave) {
+ struct bonding *bond;
+
+ bond = bond_get_bond_by_slave(origin->slave);
+ if (bond)
+ bond_3ad_set_carrier(bond);
+ }
+ }

/**
@@ -1830,6 +1897,19 @@ static void ad_initialize_lacpdu(struct lacpdu *lacpdu)
// Check aggregators status in team every T seconds
#define AD_AGGREGATOR_SELECTION_TIMER 8

+/*
+ * bond_3ad_initiate_agg_selection(struct bonding *bond)
+ *
+ * Set the aggregation selection timer, to initiate an agg selection in
+ * the very near future. Called during first initialization, and during
+ * any down to up transitions of the bond.
+ */
+void bond_3ad_initiate_agg_selection(struct bonding *bond, int timeout)
+{
+ BOND_AD_INFO(bond).agg_select_timer = timeout;
+ BOND_AD_INFO(bond).agg_select_mode = bond->params.ad_select;
+ }
+
static u16 aggregator_identifier;

/**
@@ -1854,9 +1934,9 @@ void bond_3ad_initialize(struct bonding *bond, u16 tick_resolution, int lacp_fas
// initialize how many times this module is called in one second(should be about every 100ms)
ad_ticks_per_sec = tick_resolution;

- // initialize the aggregator selection timer(to activate an aggregation selection after initialize)
- BOND_AD_INFO(bond).agg_select_timer = (AD_AGGREGATOR_SELECTION_TIMER *
ad_ticks_per_sec);
- BOND_AD_INFO(bond).agg_select_mode = AD_BANDWIDTH;
+ bond_3ad_initiate_agg_selection(bond,
+ AD_AGGREGATOR_SELECTION_TIMER *
+ ad_ticks_per_sec);
+ }
+ }
```

## [git patches] net driver updates for 2.6.29

```
diff --git a/drivers/net/bonding/bond_3ad.h b/drivers/net/bonding/bond_3ad.h
index b5ee45f..a803fe0 100644
--- a/drivers/net/bonding/bond_3ad.h
+++ b/drivers/net/bonding/bond_3ad.h
@@ -42,10 +42,11 @@ typedef struct mac_addr {
u8 mac_addr_value[ETH_ALEN];
} mac_addr_t;

-typedef enum {
- AD_BANDWIDTH = 0,
- AD_COUNT
-} agg_selection_t;
+enum {
+ BOND_AD_STABLE = 0,
+ BOND_AD_BANDWIDTH = 1,
+ BOND_AD_COUNT = 2,
+};

// rx machine states(43.4.11 in the 802.3ad standard)
typedef enum {
@@ -277,6 +278,7 @@ void bond_3ad_initialize(struct bonding *bond, u16 tick_resolution, int lacp_fas
int bond_3ad_bind_slave(struct slave *slave);
void bond_3ad_unbind_slave(struct slave *slave);
void bond_3ad_state_machine_handler(struct work_struct *);
+void bond_3ad_initiate_agg_selection(struct bonding *bond, int timeout);
void bond_3ad_adapter_speed_changed(struct slave *slave);
void bond_3ad_adapter_duplex_changed(struct slave *slave);
void bond_3ad_handle_link_change(struct slave *slave, char link);
diff --git a/drivers/net/bonding/bond_alb.c b/drivers/net/bonding/bond_alb.c
index 87437c7..e170fa2 100644
--- a/drivers/net/bonding/bond_alb.c
+++ b/drivers/net/bonding/bond_alb.c
@@ -346,14 +346,18 @@ static void rlb_update_entry_from_arp(struct bonding *bond, struct arp_pkt *arp)

static int rlb_arp_rcv(struct sk_buff *skb, struct net_device *bond_dev, struct packet_type *ptype, struct
net_device *orig_dev)
{
- struct bonding *bond = bond_dev->priv;
+ struct bonding *bond;
struct arp_pkt *arp = (struct arp_pkt *)skb->data;
int res = NET_RX_DROP;

if (dev_net(bond_dev) != &init_net)
goto out;

- if (!(bond_dev->flags & IFF_MASTER))
+ while (bond_dev->priv_flags & IFF_802_1Q_VLAN)
+ bond_dev = vlan_dev_real_dev(bond_dev);
+
+ if (!(bond_dev->priv_flags & IFF_BONDING) ||
```

## [git patches] net driver updates for 2.6.29

```
+ !(bond_dev->flags & IFF_MASTER))
goto out;

if (!arp) {
@@ -368,6 +372,9 @@ static int rlb_arp_rcv(struct sk_buff *skb, struct net_device *bond_dev, struct

if (arp->op_code == htons(ARPOP_REPLY)) {
/* update rx hash table for this ARP */
+ printk("rar: update orig %s bond_dev %s\n", orig_dev->name,
+ bond_dev->name);
+ bond = bond_dev->priv;
rlb_update_entry_from_arp(bond, arp);
dprintk("Server received an ARP Reply from client\n");
}
@@ -818,7 +825,7 @@ static int rlb_initialize(struct bonding *bond)

/*initialize packet type*/
pk_type->type = __constant_htons(ETH_P_ARP);
- pk_type->dev = bond->dev;
+ pk_type->dev = NULL;
pk_type->func = rlb_arp_rcv;

/* register to receive ARPs */
diff --git a/drivers/net/bonding/bond_ipv6.c b/drivers/net/bonding/bond_ipv6.c
new file mode 100644
index 0000000..7c78b7b
--- /dev/null
+++ b/drivers/net/bonding/bond_ipv6.c
@@ -0,0 +1,218 @@
+/*
+ * Copyright(c) 2008 Hewlett-Packard Development Company, L.P.
+ *
+ * This program is free software; you can redistribute it and/or modify it
+ * under the terms of the GNU General Public License as published by the
+ * Free Software Foundation; either version 2 of the License, or
+ * (at your option) any later version.
+ *
+ * This program is distributed in the hope that it will be useful, but
+ * WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY
+ * or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License
+ * for more details.
+ *
+ * You should have received a copy of the GNU General Public License along
+ * with this program; if not, write to the Free Software Foundation, Inc.,
+ * 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
+ *
+ * The full GNU General Public License is included in this distribution in the
+ * file called LICENSE.
+ *
+ */
+
```

```

+//#define BONDING_DEBUG 1
+
+#include <linux/types.h>
+#include <linux/if_vlan.h>
+#include <net/ipv6.h>
+#include <net/ndisc.h>
+#include <net/addrconf.h>
+#include "bonding.h"
+
+/*
+ * Assign bond->master_ipv6 to the next IPv6 address in the list, or
+ * zero it out if there are none.
+ */
+static void bond_glean_dev_ipv6(struct net_device *dev, struct in6_addr *addr)
+{
+ struct inet6_dev *idev;
+ struct inet6_ifaddr *ifa;
+
+ if (!dev)
+ return;
+
+ idev = in6_dev_get(dev);
+ if (!idev)
+ return;
+
+ read_lock_bh(&idev->lock);
+ ifa = idev->addr_list;
+ if (ifa)
+ ipv6_addr_copy(addr, &ifa->addr);
+ else
+ ipv6_addr_set(addr, 0, 0, 0, 0);
+
+ read_unlock_bh(&idev->lock);
+
+ in6_dev_put(idev);
+}
+
+static void bond_na_send(struct net_device *slave_dev,
+ struct in6_addr *daddr,
+ int router,
+ unsigned short vlan_id)
+{
+ struct in6_addr mcaddr;
+ struct icmp6hdr icmp6h = {
+ .icmp6_type = NDISC_NEIGHBOUR_ADVERTISEMENT,
+ };
+ struct sk_buff *skb;
+
+ icmp6h.icmp6_router = router;
+ icmp6h.icmp6_solicited = 0;
+ icmp6h.icmp6_override = 1;

```

[git patches] net driver updates for 2.6.29

```
+
+ addrconf_addr_solicit_mult(daddr, &mcaddr);
+
+ dprintk("ipv6 na on slave %s: dest %pI6, src %pI6\n",
+ slave->name, &mcaddr, daddr);
+
+ skb = ndisc_build_skb(slave_dev, &mcaddr, daddr, &icmp6h, daddr,
+ ND_OPT_TARGET_LL_ADDR);
+
+ if (!skb) {
+ printk(KERN_ERR DRV_NAME ": NA packet allocation failed\n");
+ return;
+ }
+
+ if (vlan_id) {
+ skb = vlan_put_tag(skb, vlan_id);
+ if (!skb) {
+ printk(KERN_ERR DRV_NAME ": failed to insert VLAN tag\n");
+ return;
+ }
+ }
+
+ ndisc_send_skb(skb, slave_dev, NULL, &mcaddr, daddr, &icmp6h);
+}
+
+/*
+ * Kick out an unsolicited Neighbor Advertisement for an IPv6 address on
+ * the bonding master. This will help the switch learn our address
+ * if in active-backup mode.
+ *
+ * Caller must hold curr_slave_lock for read or better
+ */
+void bond_send_unsolicited_na(struct bonding *bond)
+{
+ struct slave *slave = bond->curr_active_slave;
+ struct vlan_entry *vlan;
+ struct inet6_dev *idev;
+ int is_router;
+
+ dprintk("bond_send_unsol_na: bond %s slave %s\n", bond->dev->name,
+ slave ? slave->dev->name : "NULL");
+
+ if (!slave || !bond->send_unsol_na ||
+ test_bit(__LINK_STATE_LINKWATCH_PENDING, &slave->dev->state))
+ return;
+
+ bond->send_unsol_na--;
+
+ idev = in6_dev_get(bond->dev);
+ if (!idev)
+ return;
```

```

+
+ is_router = !!idev->cnf.forwarding;
+
+ in6_dev_put(idev);
+
+ if (!ipv6_addr_any(&bond->master_ipv6))
+ bond_na_send(slave->dev, &bond->master_ipv6, is_router, 0);
+
+ list_for_each_entry(vlan, &bond->vlan_list, vlan_list) {
+ if (ipv6_addr_any(&vlan->vlan_ipv6)) {
+ bond_na_send(slave->dev, &vlan->vlan_ipv6, is_router,
+ vlan->vlan_id);
+ }
+ }
+ }
+ }
+ }
+
+ /*
+ * bond_inet6addr_event: handle inet6addr notifier chain events.
+ *
+ * We keep track of device IPv6 addresses primarily to use as source
+ * addresses in NS probes.
+ *
+ * We track one IPv6 for the main device (if it has one).
+ */
+static int bond_inet6addr_event(struct notifier_block *this,
+ unsigned long event,
+ void *ptr)
+{
+ struct inet6_ifaddr *ifa = ptr;
+ struct net_device *vlan_dev, *event_dev = ifa->idev->dev;
+ struct bonding *bond;
+ struct vlan_entry *vlan;
+
+ if (dev_net(event_dev) != &init_net)
+ return NOTIFY_DONE;
+
+ list_for_each_entry(bond, &bond_dev_list, bond_list) {
+ if (bond->dev == event_dev) {
+ switch (event) {
+ case NETDEV_UP:
+ if (ipv6_addr_any(&bond->master_ipv6))
+ ipv6_addr_copy(&bond->master_ipv6,
+ &ifa->addr);
+ return NOTIFY_OK;
+ case NETDEV_DOWN:
+ if (ipv6_addr_equal(&bond->master_ipv6,
+ &ifa->addr))
+ bond_glean_dev_ipv6(bond->dev,
+ &bond->master_ipv6);
+ return NOTIFY_OK;
+ default:

```

[git patches] net driver updates for 2.6.29

```
+ return NOTIFY_DONE;
+ }
+ }
+
+ list_for_each_entry(vlan, &bond->vlan_list, vlan_list) {
+ vlan_dev = vlan_group_get_device(bond->vlgrp,
+ vlan->vlan_id);
+ if (vlan_dev == event_dev) {
+ switch (event) {
+ case NETDEV_UP:
+ if (ipv6_addr_any(&vlan->vlan_ipv6))
+ ipv6_addr_copy(&vlan->vlan_ipv6,
+ &ifa->addr);
+ return NOTIFY_OK;
+ case NETDEV_DOWN:
+ if (ipv6_addr_equal(&vlan->vlan_ipv6,
+ &ifa->addr))
+ bond_glean_dev_ipv6(vlan_dev,
+ &vlan->vlan_ipv6);
+ return NOTIFY_OK;
+ default:
+ return NOTIFY_DONE;
+ }
+ }
+ }
+ return NOTIFY_DONE;
+}
+
+static struct notifier_block bond_inet6addr_notifier = {
+ .notifier_call = bond_inet6addr_event,
+};
+
+void bond_register_ipv6_notifier(void)
+{
+ register_inet6addr_notifier(&bond_inet6addr_notifier);
+}
+
+void bond_unregister_ipv6_notifier(void)
+{
+ unregister_inet6addr_notifier(&bond_inet6addr_notifier);
+}
+
diff --git a/drivers/net/bonding/bond_main.c b/drivers/net/bonding/bond_main.c
index 39575d7..02de3e0 100644
--- a/drivers/net/bonding/bond_main.c
+++ b/drivers/net/bonding/bond_main.c
@@ -89,6 +89,7 @@
```

```
static int max_bonds = BOND_DEFAULT_MAX_BONDS;
static int num_grat_arp = 1;
```

[git patches] net driver updates for 2.6.29

```

+static int num_unsol_na = 1;
static int miimon = BOND_LINK_MON_INTERV;
static int updelay = 0;
static int downdelay = 0;
@@ -96,6 +97,7 @@ static int use_carrier = 1;
static char *mode = NULL;
static char *primary = NULL;
static char *lacp_rate = NULL;
+static char *ad_select = NULL;
static char *xmit_hash_policy = NULL;
static int arp_interval = BOND_LINK_ARP_INTERV;
static char *arp_ip_target[BOND_MAX_ARP_TARGETS] = { NULL, };
@@ -107,6 +109,8 @@ module_param(max_bonds, int, 0);
MODULE_PARM_DESC(max_bonds, "Max number of bonded devices");
module_param(num_grat_arp, int, 0644);
MODULE_PARM_DESC(num_grat_arp, "Number of gratuitous ARP packets to send on failover event");
+module_param(num_unsol_na, int, 0644);
+MODULE_PARM_DESC(num_unsol_na, "Number of unsolicited IPv6 Neighbor Advertisements packets
to send on failover event");
module_param(miimon, int, 0);
MODULE_PARM_DESC(miimon, "Link check interval in milliseconds");
module_param(updelay, int, 0);
@@ -127,6 +131,8 @@ MODULE_PARM_DESC(primary, "Primary network device to use");
module_param(lacp_rate, charp, 0);
MODULE_PARM_DESC(lacp_rate, "LACPDU tx rate to request from 802.3ad partner "
"(slow/fast)");
+module_param(ad_select, charp, 0);
+MODULE_PARM_DESC(ad_select, "803.ad aggregation selection logic: stable (0, default), bandwidth (1),
count (2)");
module_param(xmit_hash_policy, charp, 0);
MODULE_PARM_DESC(xmit_hash_policy, "XOR hashing method: 0 for layer 2 (default)"
", 1 for layer 3+4");
@@ -197,6 +203,13 @@ struct bond_parm_tbl fail_over_mac_tbl[] = {
{ NULL, -1 },
};

+struct bond_parm_tbl ad_select_tbl[] = {
+{ "stable", BOND_AD_STABLE },
+{ "bandwidth", BOND_AD_BANDWIDTH },
+{ "count", BOND_AD_COUNT },
+{ NULL, -1 },
+};
+
+/*----- Forward declarations -----*/

static void bond_send_gratuitous_arp(struct bonding *bond);
@@ -242,14 +255,13 @@ static int bond_add_vlan(struct bonding *bond, unsigned short vlan_id)
dprintk("bond: %s, vlan id %d\n",
(bond ? bond->dev->name: "None"), vlan_id);

- vlan = kmalloc(sizeof(struct vlan_entry), GFP_KERNEL);

```

[git patches] net driver updates for 2.6.29

```
+ vlan = kzalloc(sizeof(struct vlan_entry), GFP_KERNEL);
if (!vlan) {
return -ENOMEM;
}

INIT_LIST_HEAD(&vlan->vlan_list);
vlan->vlan_id = vlan_id;
- vlan->vlan_ip = 0;

write_lock_bh(&bond->lock);

@@ -1208,6 +1220,9 @@ void bond_change_active_slave(struct bonding *bond, struct slave *new_active)
bond->send_grat_arp = bond->params.num_grat_arp;
bond_send_gratuitous_arp(bond);

+ bond->send_unsol_na = bond->params.num_unsol_na;
+ bond_send_unsolicited_na(bond);
+
write_unlock_bh(&bond->curr_slave_lock);
read_unlock(&bond->lock);

@@ -2463,6 +2478,12 @@ void bond_mii_monitor(struct work_struct *work)
read_unlock(&bond->curr_slave_lock);
}

+ if (bond->send_unsol_na) {
+ read_lock(&bond->curr_slave_lock);
+ bond_send_unsolicited_na(bond);
+ read_unlock(&bond->curr_slave_lock);
+ }
+
if (bond_miimon_inspect(bond)) {
read_unlock(&bond->lock);
rtnl_lock();
@@ -3158,6 +3179,12 @@ void bond_activebackup_arp_mon(struct work_struct *work)
read_unlock(&bond->curr_slave_lock);
}

+ if (bond->send_unsol_na) {
+ read_lock(&bond->curr_slave_lock);
+ bond_send_unsolicited_na(bond);
+ read_unlock(&bond->curr_slave_lock);
+ }
+
if (bond_ab_arp_inspect(bond, delta_in_ticks)) {
read_unlock(&bond->lock);
rtnl_lock();
@@ -3301,6 +3328,8 @@ static void bond_info_show_master(struct seq_file *seq)
seq_puts(seq, "\n802.3ad info\n");
seq_printf(seq, "LACP rate: %s\n",
(bond->params.lacp_fast) ? "fast" : "slow");
```

## [git patches] net driver updates for 2.6.29

```
+ seq_printf(seq, "Aggregator selection policy (ad_select): %s\n",
+ ad_select_tbl[bond->params.ad_select].modename);

if (bond_3ad_get_active_agg_info(bond, &ad_info) {
seq_printf(seq, "bond %s has no active aggregator\n",
@@ -3807,6 +3836,7 @@ static int bond_open(struct net_device *bond_dev)
queue_delayed_work(bond->wq, &bond->ad_work, 0);
/* register to receive LACPDU's */
bond_register_lacpdu(bond);
+ bond_3ad_initiate_agg_selection(bond, 1);
}

return 0;
@@ -3827,6 +3857,7 @@ static int bond_close(struct net_device *bond_dev)
write_lock_bh(&bond->lock);

bond->send_grat_arp = 0;
+ bond->send_unsol_na = 0;

/* signal timers not to re-arm */
bond->kill_timers = 1;
@@ -4542,6 +4573,7 @@ static int bond_init(struct net_device *bond_dev, struct bond_params *params)
bond->primary_slave = NULL;
bond->dev = bond_dev;
bond->send_grat_arp = 0;
+ bond->send_unsol_na = 0;
bond->setup_by_slave = 0;
INIT_LIST_HEAD(&bond->vlan_list);

@@ -4744,6 +4776,23 @@ static int bond_check_params(struct bond_params *params)
}
}

+ if (ad_select) {
+ params->ad_select = bond_parse_parm(ad_select, ad_select_tbl);
+ if (params->ad_select == -1) {
+ printk(KERN_ERR DRV_NAME
+ ": Error: Invalid ad_select \"%s\"\n",
+ ad_select == NULL ? "NULL" : ad_select);
+ return -EINVAL;
+ }
+
+ if (bond_mode != BOND_MODE_8023AD) {
+ printk(KERN_WARNING DRV_NAME
+ ": ad_select param only affects 802.3ad mode\n");
+ }
+ } else {
+ params->ad_select = BOND_AD_STABLE;
+ }
+
+ if (max_bonds < 0 || max_bonds > INT_MAX) {
```

## [git patches] net driver updates for 2.6.29

```
printk(KERN_WARNING DRV_NAME
": Warning: max_bonds (%d) not in range %d-%d, so it "
@@ -4791,6 +4840,13 @@ static int bond_check_params(struct bond_params *params)
num_grat_arp = 1;
}

+ if (num_unsol_na < 0 || num_unsol_na > 255) {
+ printk(KERN_WARNING DRV_NAME
+ ": Warning: num_unsol_na (%d) not in range 0-255 so it "
+ "was reset to 1 \n", num_unsol_na);
+ num_unsol_na = 1;
+ }
+
/* reset values for 802.3ad */
if (bond_mode == BOND_MODE_8023AD) {
if (!miimon) {
@@ -4992,6 +5048,7 @@ static int bond_check_params(struct bond_params *params)
params->xmit_policy = xmit_hashtype;
params->miimon = miimon;
params->num_grat_arp = num_grat_arp;
+ params->num_unsol_na = num_unsol_na;
params->arp_interval = arp_interval;
params->arp_validate = arp_validate_value;
params->updelay = updelay;
@@ -5144,6 +5201,7 @@ static int __init bonding_init(void)

register_netdevice_notifier(&bond_netdev_notifier);
register_inetaddr_notifier(&bond_inetaddr_notifier);
+ bond_register_ipv6_notifier();

goto out;
err:
@@ -5166,6 +5224,7 @@ static void __exit bonding_exit(void)
{
unregister_netdevice_notifier(&bond_netdev_notifier);
unregister_inetaddr_notifier(&bond_inetaddr_notifier);
+ bond_unregister_ipv6_notifier();

bond_destroy_sysfs());

diff --git a/drivers/net/bonding/bond_sysfs.c b/drivers/net/bonding/bond_sysfs.c
index e400d7d..aaf2927 100644
--- a/drivers/net/bonding/bond_sysfs.c
+++ b/drivers/net/bonding/bond_sysfs.c
@@ -48,6 +48,7 @@ extern struct list_head bond_dev_list;
extern struct bond_params bonding_defaults;
extern struct bond_parm_tbl bond_mode_tbl[];
extern struct bond_parm_tbl bond_lacp_tbl[];
+extern struct bond_parm_tbl ad_select_tbl[];
extern struct bond_parm_tbl xmit_hashtype_tbl[];
extern struct bond_parm_tbl arp_validate_tbl[];
```

## [git patches] net driver updates for 2.6.29

```
extern struct bond_parm_tbl fail_over_mac_tbl[];
@@ -944,6 +945,53 @@ out:
}
static DEVICE_ATTR(lacp_rate, S_IRUGO | S_IWUSR, bonding_show_lacp, bonding_store_lacp);

+static ssize_t bonding_show_ad_select(struct device *d,
+ struct device_attribute *attr,
+ char *buf)
+{
+ struct bonding *bond = to_bond(d);
+
+ return sprintf(buf, "%s %d\n",
+ ad_select_tbl[bond->params.ad_select].modename,
+ bond->params.ad_select);
+}
+
+static ssize_t bonding_store_ad_select(struct device *d,
+ struct device_attribute *attr,
+ const char *buf, size_t count)
+{
+ int new_value, ret = count;
+ struct bonding *bond = to_bond(d);
+
+ if (bond->dev->flags & IFF_UP) {
+ printk(KERN_ERR DRV_NAME
+ ": %s: Unable to update ad_select because interface "
+ "is up.\n", bond->dev->name);
+ ret = -EPERM;
+ goto out;
+ }
+
+ new_value = bond_parse_parm(buf, ad_select_tbl);
+
+ if (new_value != -1) {
+ bond->params.ad_select = new_value;
+ printk(KERN_INFO DRV_NAME
+ ": %s: Setting ad_select to %s (%d).\n",
+ bond->dev->name, ad_select_tbl[new_value].modename,
+ new_value);
+ } else {
+ printk(KERN_ERR DRV_NAME
+ ": %s: Ignoring invalid ad_select value %.*s.\n",
+ bond->dev->name, (int)strlen(buf) - 1, buf);
+ ret = -EINVAL;
+ }
+out:
+ return ret;
+}
+
+static DEVICE_ATTR(ad_select, S_IRUGO | S_IWUSR, bonding_show_ad_select,
```

## [git patches] net driver updates for 2.6.29

```
bonding_store_ad_select);
+
+/*
+ * Show and set the number of grat ARP to send after a failover event.
+ */
@@ -983,6 +1031,47 @@ out:
return ret;
}
static DEVICE_ATTR(num_grat_arp, S_IRUGO | S_IWUSR, bonding_show_n_grat_arp,
bonding_store_n_grat_arp);
+
+/*
+ * Show and set the number of unsolicited NA's to send after a failover event.
+ */
+static ssize_t bonding_show_n_unsol_na(struct device *d,
+ struct device_attribute *attr,
+ char *buf)
+{
+ struct bonding *bond = to_bond(d);
+
+ return sprintf(buf, "%d\n", bond->params.num_unsol_na);
+}
+
+static ssize_t bonding_store_n_unsol_na(struct device *d,
+ struct device_attribute *attr,
+ const char *buf, size_t count)
+{
+ int new_value, ret = count;
+ struct bonding *bond = to_bond(d);
+
+ if (sscanf(buf, "%d", &new_value) != 1) {
+ printk(KERN_ERR DRV_NAME
+ ": %s: no num_unsol_na value specified.\n",
+ bond->dev->name);
+ ret = -EINVAL;
+ goto out;
+ }
+ if (new_value < 0 || new_value > 255) {
+ printk(KERN_ERR DRV_NAME
+ ": %s: Invalid num_unsol_na value %d not in range 0-255; rejected.\n",
+ bond->dev->name, new_value);
+ ret = -EINVAL;
+ goto out;
+ } else {
+ bond->params.num_unsol_na = new_value;
+ }
+out:
+ return ret;
+}
+static DEVICE_ATTR(num_unsol_na, S_IRUGO | S_IWUSR, bonding_show_n_unsol_na,
bonding_store_n_unsol_na);
```

```

+
+/*
+ * Show and set the MII monitor interval. There are two tricky bits
+ * here. First, if MII monitoring is activated, then we must disable
@@ -1418,8 +1507,10 @@ static struct attribute *per_bond_attrs[] = {
&dev_attr_downdelay.attr,
&dev_attr_updelay.attr,
&dev_attr_lacp_rate.attr,
+ &dev_attr_ad_select.attr,
&dev_attr_xmit_hash_policy.attr,
&dev_attr_num_grat_arp.attr,
+ &dev_attr_num_unsol_na.attr,
&dev_attr_miimon.attr,
&dev_attr_primary.attr,
&dev_attr_use_carrier.attr,
diff --git a/drivers/net/bonding/bonding.h b/drivers/net/bonding/bonding.h
index ffb668d..b5eb8e6 100644
--- a/drivers/net/bonding/bonding.h
+++ b/drivers/net/bonding/bonding.h
@@ -19,16 +19,19 @@
#include <linux/proc_fs.h>
#include <linux/if_bonding.h>
#include <linux/kobject.h>
+#include <linux/in6.h>
#include "bond_3ad.h"
#include "bond_alb.h"

-#define DRV_VERSION "3.3.0"
-#define DRV_RELDATE "June 10, 2008"
+#define DRV_VERSION "3.5.0"
+#define DRV_RELDATE "November 4, 2008"
#define DRV_NAME "bonding"
#define DRV_DESCRIPTION "Ethernet Channel Bonding Driver"

#define BOND_MAX_ARP_TARGETS 16

+extern struct list_head bond_dev_list;
+
+#ifdef BONDING_DEBUG
#define dprintk(fmt, args...) \
printk(KERN_DEBUG \
@@ -126,6 +129,7 @@ struct bond_params {
int xmit_policy;
int miimon;
int num_grat_arp;
+ int num_unsol_na;
int arp_interval;
int arp_validate;
int use_carrier;
@@ -133,6 +137,7 @@ struct bond_params {
int updelay;

```

```

int downdelay;
int lacp_fast;
+ int ad_select;
char primary[IFNAMSIZ];
__be32 arp_targets[BOND_MAX_ARP_TARGETS];
};
@@ -148,6 +153,9 @@ struct vlan_entry {
struct list_head vlan_list;
__be32 vlan_ip;
unsigned short vlan_id;
+#if defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE)
+ struct in6_addr vlan_ipv6;
+#endif
};

struct slave {
@@ -195,6 +203,7 @@ struct bonding {
rwlock_t curr_slave_lock;
s8 kill_timers;
s8 send_grat_arp;
+ s8 send_unsol_na;
s8 setup_by_slave;
struct net_device_stats stats;
#ifdef CONFIG_PROC_FS
@@ -218,6 +227,9 @@ struct bonding {
struct delayed_work arp_work;
struct delayed_work alb_work;
struct delayed_work ad_work;
+#if defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE)
+ struct in6_addr master_ipv6;
+#endif
};

/**
@@ -341,5 +353,24 @@ extern struct bond_parm_tbl xmit_hashtype_tbl[];
extern struct bond_parm_tbl arp_validate_tbl[];
extern struct bond_parm_tbl fail_over_mac_tbl[];

+#if defined(CONFIG_IPV6) || defined(CONFIG_IPV6_MODULE)
+void bond_send_unsolicited_na(struct bonding *bond);
+void bond_register_ipv6_notifier(void);
+void bond_unregister_ipv6_notifier(void);
+#else
+static inline void bond_send_unsolicited_na(struct bonding *bond)
+{
+ return;
+}
+static inline void bond_register_ipv6_notifier(void)
+{
+ return;
+}

```

## [git patches] net driver updates for 2.6.29

```
+static inline void bond_unregister_ipv6_notifier(void)
+{
+ return;
+}
+endif
+
+endif /* _LINUX_BONDING_H */
```

```
diff --git a/drivers/net/ehea/ehea.h b/drivers/net/ehea/ehea.h
```

```
index 002d918..9930d5f 100644
```

```
--- a/drivers/net/ehea/ehea.h
```

```
+++ b/drivers/net/ehea/ehea.h
```

```
@@ -40,7 +40,7 @@
```

```
#include <asm/io.h>
```

```
#define DRV_NAME "ehea"
```

```
---#define DRV_VERSION "EHEA_0095"
```

```
+#define DRV_VERSION "EHEA_0096"
```

```
/* eHEA capability flags */
```

```
#define DLPAR_PORT_ADD_REM 1
```

```
diff --git a/drivers/net/ehea/ehea_qmr.c b/drivers/net/ehea/ehea_qmr.c
```

```
index 9d00687..3c0ec82 100644
```

```
--- a/drivers/net/ehea/ehea_qmr.c
```

```
+++ b/drivers/net/ehea/ehea_qmr.c
```

```
@@ -653,7 +653,7 @@ static int ehea_update_busmap(unsigned long pfn, unsigned long nr_pages, int add
```

```
int top = ehea_calc_index(i, EHEA_TOP_INDEX_SHIFT);
```

```
int dir = ehea_calc_index(i, EHEA_DIR_INDEX_SHIFT);
```

```
int idx = i & EHEA_INDEX_MASK;
```

```
-
```

```
+
```

```
if (add) {
```

```
int ret = ehea_init_bmap(ehea_bmap, top, dir);
```

```
if (ret)
```

```
@@ -780,7 +780,7 @@ void ehea_destroy_busmap(void)
```

```
kfree(ehea_bmap);
```

```
ehea_bmap = NULL;
```

```
-out_destroy:
```

```
+out_destroy:
```

```
mutex_unlock(&ehea_busmap_mutex);
```

```
}
```

```
@@ -858,10 +858,10 @@ static u64 ehea_reg_mr_sections(int top, int dir, u64 *pt,
```

```
for (idx = 0; idx < EHEA_MAP_ENTRIES; idx++) {
```

```
if (!ehea_bmap->top[top]->dir[dir]->ent[idx])
```

```
continue;
```

```
-
```

```
+
```

```
hret = ehea_reg_mr_section(top, dir, idx, pt, adapter, mr);
```

```
if ((hret != H_SUCCESS) && (hret != H_PAGE_REGISTERED))
```

## [git patches] net driver updates for 2.6.29

```
- return hret;
+ return hret;
}
return hret;
}
@@ -879,7 +879,7 @@ static u64 ehea_reg_mr_dir_sections(int top, u64 *pt,

hret = ehea_reg_mr_sections(top, dir, pt, adapter, mr);
if ((hret != H_SUCCESS) && (hret != H_PAGE_REGISTERED))
- return hret;
+ return hret;
}
return hret;
}
diff --git a/drivers/net/pcmcia/fmvj18x_cs.c b/drivers/net/pcmcia/fmvj18x_cs.c
index e4f8fe3..69dcfbb 100644
--- a/drivers/net/pcmcia/fmvj18x_cs.c
+++ b/drivers/net/pcmcia/fmvj18x_cs.c
@@ -125,6 +125,7 @@ typedef struct local_info_t {
u_short tx_queue_len;
cardtype_t cardtype;
u_short sent;
+ u_char __iomem *base;
} local_info_t;

#define MC_FILTERBREAK 64
@@ -242,6 +243,7 @@ static int fmvj18x_probe(struct pcmcia_device *link)
lp = netdev_priv(dev);
link->priv = dev;
lp->p_dev = link;
+ lp->base = NULL;

/* The io structure describes IO port mapping */
link->io.NumPorts1 = 32;
@@ -442,8 +444,10 @@ static int fmvj18x_config(struct pcmcia_device *link)
dev->irq = link->irq.AssignedIRQ;
dev->base_addr = link->io.BasePort1;

- if (link->io.BasePort2 != 0)
- fmvj18x_setup_mfc(link);
+ if (link->io.BasePort2 != 0) {
+ ret = fmvj18x_setup_mfc(link);
+ if (ret != 0) goto failed;
+ }

ioaddr = dev->base_addr;

@@ -610,10 +614,10 @@ static int fmvj18x_setup_mfc(struct pcmcia_device *link)
{
win_req_t req;
memreq_t mem;
```

[git patches] net driver updates for 2.6.29

```
- u_char __iomem *base;
- int i, j;
+ int i;
struct net_device *dev = link->priv;
unsigned int ioaddr;
+ local_info_t *lp = netdev_priv(dev);

/* Allocate a small memory window */
req.Attributes = WIN_DATA_WIDTH_8|WIN_MEMORY_TYPE_AM|WIN_ENABLE;
@@ -625,25 +629,32 @@ static int fmvj18x_setup_mfc(struct pcmcia_device *link)
return -1;
}

- base = ioremap(req.Base, req.Size);
+ lp->base = ioremap(req.Base, req.Size);
+ if (lp->base == NULL) {
+ printk(KERN_NOTICE "fmvj18x_cs: ioremap failed\n");
+ return -1;
+ }
+
mem.Page = 0;
mem.CardOffset = 0;
- pcmcia_map_mem_page(link->win, &mem);
-
+ i = pcmcia_map_mem_page(link->win, &mem);
+ if (i != 0) {
+ iounmap(lp->base);
+ lp->base = NULL;
+ cs_error(link, MapMemPage, i);
+ return -1;
+ }
+
ioaddr = dev->base_addr;
- writeb(0x47, base+0x800); /* Config Option Register of LAN */
- writeb(0x0, base+0x802); /* Config and Status Register */
+ writeb(0x47, lp->base+0x800); /* Config Option Register of LAN */
+ writeb(0x0, lp->base+0x802); /* Config and Status Register */

- writeb(ioaddr & 0xff, base+0x80a); /* I/O Base(Low) of LAN */
- writeb((ioaddr >> 8) & 0xff, base+0x80c); /* I/O Base(High) of LAN */
+ writeb(ioaddr & 0xff, lp->base+0x80a); /* I/O Base(Low) of LAN */
+ writeb((ioaddr >> 8) & 0xff, lp->base+0x80c); /* I/O Base(High) of LAN */

- writeb(0x45, base+0x820); /* Config Option Register of Modem */
- writeb(0x8, base+0x822); /* Config and Status Register */
+ writeb(0x45, lp->base+0x820); /* Config Option Register of Modem */
+ writeb(0x8, lp->base+0x822); /* Config and Status Register */

- iounmap(base);
- j = pcmcia_release_window(link->win);
- if (j != 0)
```

## [git patches] net driver updates for 2.6.29

```
- cs_error(link, ReleaseWindow, j);
return 0;

}
@@ -651,8 +662,25 @@ static int fmvj18x_setup_mfc(struct pcmcia_device *link)

static void fmvj18x_release(struct pcmcia_device *link)
{
- DEBUG(0, "fmvj18x_release(0x%p)\n", link);
- pcmcia_disable_device(link);
+
+ struct net_device *dev = link->priv;
+ local_info_t *lp = netdev_priv(dev);
+ u_char __iomem *tmp;
+ int j;
+
+ DEBUG(0, "fmvj18x_release(0x%p)\n", link);
+
+ if (lp->base != NULL) {
+ tmp = lp->base;
+ lp->base = NULL; /* set NULL before iounmap */
+ iounmap(tmp);
+ j = pcmcia_release_window(link->win);
+ if (j != 0)
+ cs_error(link, ReleaseWindow, j);
+ }
+
+ pcmcia_disable_device(link);
+
}

static int fmvj18x_suspend(struct pcmcia_device *link)
@@ -783,6 +811,13 @@ static irqreturn_t fjn_interrupt(int dummy, void *dev_id)

outb(D_TX_INTR, iaddr + TX_INTR);
outb(D_RX_INTR, iaddr + RX_INTR);
+
+ if (lp->base != NULL) {
+ /* Ack interrupt for multifunction card */
+ writeb(0x01, lp->base+0x802);
+ writeb(0x09, lp->base+0x822);
+ }
+
return IRQ_HANDLED;

} /* fjn_interrupt */
diff --git a/drivers/net/phy/smsc.c b/drivers/net/phy/smsc.c
index 73baa7a..c05d38d 100644
--- a/drivers/net/phy/smsc.c
+++ b/drivers/net/phy/smsc.c
@@ -126,6 +126,27 @@ static struct phy_driver lan8700_driver = {
```

[git patches] net driver updates for 2.6.29

```
.driver = { .owner = THIS_MODULE, }
};

+static struct phy_driver lan911x_int_driver = {
+ .phy_id = 0x0007c0d0, /* OUI=0x00800f, Model#=0x0d */
+ .phy_id_mask = 0xfffff0,
+ .name = "SMSC LAN911x Internal PHY",
+
+ .features = (PHY_BASIC_FEATURES | SUPPORTED_Pause
+ | SUPPORTED_Asym_Pause),
+ .flags = PHY_HAS_INTERRUPT | PHY_HAS_MAGICANEG,
+
+ /* basic functions */
+ .config_aneg = genphy_config_aneg,
+ .read_status = genphy_read_status,
+ .config_init = smsc_phy_config_init,
+
+ /* IRQ related */
+ .ack_interrupt = smsc_phy_ack_interrupt,
+ .config_intr = smsc_phy_config_intr,
+
+ .driver = { .owner = THIS_MODULE, }
+};

+
static int __init smsc_init(void)
{
int ret;
@@ -142,8 +163,14 @@ static int __init smsc_init(void)
if (ret)
goto err3;

+ ret = phy_driver_register (&lan911x_int_driver);
+ if (ret)
+ goto err4;
+
return 0;

+err4:
+ phy_driver_unregister (&lan8700_driver);
err3:
phy_driver_unregister (&lan8187_driver);
err2:
@@ -154,6 +181,7 @@ err1:

static void __exit smsc_exit(void)
{
+ phy_driver_unregister (&lan911x_int_driver);
phy_driver_unregister (&lan8700_driver);
phy_driver_unregister (&lan8187_driver);
phy_driver_unregister (&lan83c185_driver);
diff --git a/drivers/net/sfc/Kconfig b/drivers/net/sfc/Kconfig
```

## [git patches] net driver updates for 2.6.29

```
index 3be13b5..3e25fb3 100644
--- a/drivers/net/sfc/Kconfig
+++ b/drivers/net/sfc/Kconfig
@@ -12,3 +12,11 @@ config SFC
```

To compile this driver as a module, choose M here. The module will be called sfc.

```
+config SFC_MTD
+ bool "Solarflare Solarstorm SFC4000 flash MTD support"
+ depends on SFC && MTD
+ default y
+ help
+ This exposes the on-board flash memory as an MTD device (e.g.
+ /dev/mtd1). This makes it possible to upload new boot code
+ to the NIC.
```

```
diff --git a/drivers/net/sfc/Makefile b/drivers/net/sfc/Makefile
index c8f5704..e507daa 100644
--- a/drivers/net/sfc/Makefile
+++ b/drivers/net/sfc/Makefile
@@ -1,5 +1,6 @@
sfc-y += efx.o falcon.o tx.o rx.o falcon_xmac.o \
selftest.o ethtool.o xfp_phy.o \
mdio_10g.o tenxpress.o boards.o sfe4001.o
+sfc-$(CONFIG_SFC_MTD) += mtd.o
```

```
obj-$(CONFIG_SFC) += sfc.o
diff --git a/drivers/net/sfc/boards.c b/drivers/net/sfc/boards.c
index 99e6023..edf0262 100644
--- a/drivers/net/sfc/boards.c
+++ b/drivers/net/sfc/boards.c
@@ -11,6 +11,7 @@
#include "phy.h"
#include "boards.h"
#include "efx.h"
+#include "workarounds.h"
```

```
/* Macros for unpacking the board revision */
/* The revision info is in host byte order. */
@@ -52,9 +53,128 @@ static void board_blink(struct efx_nic *efx, bool blink)
}
```

```
/******
+ * Support for LM87 sensor chip used on several boards
+ */
+#define LM87_REG_ALARMS1 0x41
+#define LM87_REG_ALARMS2 0x42
+#define LM87_IN_LIMITS(nr, _min, _max) \
+ 0x2B + (nr) * 2, _max, 0x2C + (nr) * 2, _min
+#define LM87_AIN_LIMITS(nr, _min, _max) \
+ 0x3B + (nr), _max, 0x1A + (nr), _min
+#define LM87_TEMP_INT_LIMITS(_min, _max) \
```

[git patches] net driver updates for 2.6.29

```
+ 0x39, _max, 0x3A, _min
+#define LM87_TEMP_EXT1_LIMITS(_min, _max) \
+ 0x37, _max, 0x38, _min
+
+#define LM87_ALARM_TEMP_INT 0x10
+#define LM87_ALARM_TEMP_EXT1 0x20
+
+#if defined(CONFIG_SENSORS_LM87) || defined(CONFIG_SENSORS_LM87_MODULE)
+
+static int efx_init_lm87(struct efx_nic *efx, struct i2c_board_info *info,
+ const u8 *reg_values)
+{
+ struct i2c_client *clien+ ctrl1 = ctrl2 = mdio_clause45_read(efx, phy,
+ mmd, MDIO_MMDREG_CTRL1);
+ if (!power)
+ ctrl2 |= (1 << MDIO_MMDREG_CTRL1_LPOWER_LBN);
+ else
+ ctrl2 &= ~(1 << MDIO_MMDREG_CTRL1_LPOWER_LBN);
+ if (ctrl1 != ctrl2)
+ mdio_clause45_write(efx, phy, mmd,
+ MDIO_MMDREG_CTRL1, ctrl2);
+ }
+}
+
+void mdio_clause45_set_mmds_lpower(struct efx_nic *efx,
+ int low_power, unsigned int mmd_mask)
+{
+ int mmd = 0;
+ while (mmd_mask) {
+ if (mmd_mask & 1)
+ mdio_clause45_set_mmd_lpower(efx, low_power, mmd);
+ mmd_mask = (mmd_mask >> 1);
+ mmd++;
+ }
+}
+
+/**
+ * mdio_clause45_get_settings – Read (some of) the PHY settings over MDIO.
+ * @efx: Efx NIC
+ diff --git a/drivers/net/sfc/mdio_10g.h b/drivers/net/sfc/mdio_10g.h
+ index 19c42ea..db9f358 100644
+ --- a/drivers/net/sfc/mdio_10g.h
+ +++ b/drivers/net/sfc/mdio_10g.h
+ @@ -54,6 +54,9 @@
+ /* Loopback bit for WIS, PCS, PHYSX and DTEXS */
+ #define MDIO_MMDREG_CTRL1_LBACK_LBN (14)
+ #define MDIO_MMDREG_CTRL1_LBACK_WIDTH (1)
+ /* Low power */
+ #define MDIO_MMDREG_CTRL1_LPOWER_LBN (11)
+ #define MDIO_MMDREG_CTRL1_LPOWER_WIDTH (1)
```

## [git patches] net driver updates for 2.6.29

```
/* Bits in MMDREG_STAT1 */
#define MDIO_MMDREG_STAT1_FAULT_LBN (7)
@@ -240,6 +243,10 @@ extern void mdio_clause45_transmit_disable(struct efx_nic *efx);
/* Generic part of reconfigure: set/clear loopback bits */
extern void mdio_clause45_phy_reconfigure(struct efx_nic *efx);

+/* Set the power state of the specified MMDs */
+extern void mdio_clause45_set_mmds_lpower(struct efx_nic *efx,
+ int low_power, unsigned int mmd_mask);
+
+/* Read (some of) the PHY settings over MDIO */
extern void mdio_clause45_get_settings(struct efx_nic *efx,
struct ethtool_cmd *ecmd);
diff --git a/drivers/net/sfc/mtd.c b/drivers/net/sfc/mtd.c
new file mode 100644
index 0000000..a1e6c28
--- /dev/null
+++ b/drivers/net/sfc/mtd.c
@@ -0,0 +1,268 @@
+/******
+ * Driver for Solarflare Solarstorm network controllers and boards
+ * Copyright 2005–2006 Fen Systems Ltd.
+ * Copyright 2006–2008 Solarflare Communications Inc.
+ *
+ * This program is free software; you can redistribute it and/or modify it
+ * under the terms of the GNU General Public License version 2 as published
+ * by the Free Software Foundation, incorporated herein by reference.
+ */
+
+#include <linux/module.h>
+#include <linux/mtd/mtd.h>
+#include <linux/delay.h>
+
+#define EFX_DRIVER_NAME "sfc_mtd"
+#include "net_driver.h"
+#include "spi.h"
+
+#define EFX_SPI_VERIFY_BUF_LEN 16
+
+struct efx_mtd {
+ const struct efx_spi_device *spi;
+ struct mtd_info mtd;
+ char name[IFNAMSIZ + 20];
+};
+
+/* SPI utilities */
+
+static int efx_spi_slow_wait(struct efx_mtd *efx_mtd, bool uninterruptible)
+{
+ const struct efx_spi_device *spi = efx_mtd->spi;
+ struct efx_nic *efx = spi->efx;
```

[git patches] net driver updates for 2.6.29

```
+ u8 status;
+ int rc, i;
+
+ /* Wait up to 4s for flash/EEPROM to finish a slow operation. */
+ for (i = 0; i < 40; i++) {
+ __set_current_state(uninterruptible ?
+ TASK_UNINTERRUPTIBLE : TASK_INTERRUPTIBLE);
+ schedule_timeout(HZ / 10);
+ rc = falcon_spi_cmd(spi, SPI_RDSR, -1, NULL,
+ &status, sizeof(status));
+ if (rc)
+ return rc;
+ if (!(status & SPI_STATUS_NRDY))
+ return 0;
+ if (signal_pending(current))
+ return -EINTR;
+ }
+ EFX_ERR(efx, "timed out waiting for %s\n", efx_mtd->name);
+ return -ETIMEDOUT;
+}
+
+static int efx_spi_unlock(const struct efx_spi_device *spi)
+{
+ const u8 unlock_mask = (SPI_STATUS_BP2 | SPI_STATUS_BP1 |
+ SPI_STATUS_BP0);
```