

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/hackers/2007-06/msg00140.html>

- *From:* Stefan Sperling <stsp@xxxxxxxxx>
 - *Date:* Thu, 14 Jun 2007 01:21:03 +0200
-

On Sun, Jun 10, 2007 at 04:45:04PM +0200, Stefan Sperling wrote:

I usually use either if_em or if_xl chipsets, so I hoped landing this code in at least -CURRENT (should go there first, I guess) would result in more chipsets supported ;)

There is code for enabling wake on lan in the Linux drivers for both if_xl and if_em cards. See drivers/net/3c59x.c and drivers/net/e1000/e1000_ethtool.c in the linux source tree.

So I don't think adding support for these cards is a problem. Just need to find some time to do it...

Updated patch attached. Also available at <http://stsp.name/wol/>

Now contains untested support for if_xl. Please test. I have no such card so I cannot test this myself. Note that apparently only 3C905B type cards support wake on lan. And they only support magic packet events, no unicast, broadcast or multicast events.

The patch is against RELENG_6_2. Sorry I have no -CURRENT system set up at the moment. I don't know if the patch even applies to -CURRENT.

Once if_xl is working I'll look into getting if_em supported.

If any nve users are reading this, I still need feedback on if_nve as well. I cannot test nve myself either.

Thanks,

--

stefan

<http://stsp.name> PGP Key: 0xF59D25F0

Index: sbin/ifconfig/Makefile

=====

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

```
RCS file: /usr/local/ncvs/src/sbin/ifconfig/Makefile,v
retrieving revision 1.29
diff -u -u -r1.29 Makefile
--- sbin/ifconfig/Makefile 5 Jun 2005 03:32:51 -0000 1.29
+++ sbin/ifconfig/Makefile 6 May 2006 11:08:41 -0000
@@ -28,6 +28,8 @@
```

SRCS+= ifbridge.c # bridge support

+SRCS+= ifwol.c # wake on lan support

+

.if !defined(RELEASE_CRUNCH)

SRCS+= af_ipx.c # IPX support

DPADD= \${LIBIPX}

Index: sbin/ifconfig/ifconfig.8

```
RCS file: /usr/local/ncvs/src/sbin/ifconfig/ifconfig.8,v
retrieving revision 1.95.2.17
diff -u -u -r1.95.2.17 ifconfig.8
--- sbin/ifconfig/ifconfig.8 3 Nov 2006 09:14:24 -0000 1.95.2.17
+++ sbin/ifconfig/ifconfig.8 13 Jan 2007 12:18:33 -0000
@@ -939,6 +939,27 @@
```

If that is the case, then the first four keys

(1-4) will be the standard temporary keys and any others will be adaptor specific keys such as permanent keys stored in NVRAM.

+.It Cm wakeon Ar events

+Enable Wake On Lan support, if available. The

+.Ar events

+argument is a comma separated list of package types that shall

+trigger wake events. The set of valid package types is

+.Dq Li unicast ,

+.Dq Li multicast ,

+.Dq Li broadcast ,

+and

+.Dq Li magic .

+These enable wake on unicast, multicast, broadcast and Magic Packet(tm),

+respectively.

+A SecureOn password, if supported, can be enabled using the

+.Dq Li sopasswd:<password>

+event.

+SecureOn passwords only work in combination with

+.Dq Li magic .

+The password must consist of 12 hexadecimal digits.

+.It Fl wakeon

+Disable Wake On Lan.

+.Pp

.It Cm wme

Enable Wireless Multimedia Extensions (WME) support, if available, for the specified interface.

@@ -946,7 +967,6 @@

efficient communication of realtime and multimedia data.

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

To disable WME support, use

.Fl wme .

-.Pp

The following parameters are meaningful only when WME support is in use.

Parameters are specified per-AC (Access Category) and

split into those that are used by a station when acting

Index: sbin/ifconfig/ifwol.c

=====
RCS file: sbin/ifconfig/ifwol.c

diff -N sbin/ifconfig/ifwol.c

--- /dev/null 1 Jan 1970 00:00:00 -0000

+++ sbin/ifconfig/ifwol.c 20 Oct 2006 10:29:10 -0000

@@ -0,0 +1,229 @@

+/* \$Id\$ */

+

+/*

+ * Copyright (c) 2005 Stefan Sperling.

+ * All rights reserved.

+ *

+ * Redistribution and use in source and binary forms, with or without

+ * modification, are permitted provided that the following conditions

+ * are met:

+ * 1. Redistributions of source code must retain the above copyright

+ * notice, this list of conditions and the following disclaimer.

+ * 2. Redistributions in binary form must reproduce the above copyright

+ * notice, this list of conditions and the following disclaimer in the

+ * documentation and/or other materials provided with the distribution.

+ * 3. The name of the author may not be used to endorse or promote products

+ * derived from this software without specific prior written permission.

+ *

+ * THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS" AND ANY EXPRESS OR

+ * IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES

+ * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

+ * IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,

+ * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,

+ * BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;

+ * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED

+ * AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,

+ * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY

+ * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF

+ * SUCH DAMAGE.

+ */

+

+#include <sys/param.h>

+#include <sys/ioctl.h>

+#include <sys/socket.h>

+#include <sys/sysctl.h>

+#include <sys/time.h>

+

+#include <net/if.h>

+#include <net/if_dl.h>

```
+#include <net/if_types.h>
+#include <net/if_media.h>
+#include <net/route.h>
+
+#include <ctype.h>
+#include <err.h>
+#include <errno.h>
+#include <fcntl.h>
+#include <stdio.h>
+#include <stdlib.h>
+#include <string.h>
+#include <unistd.h>
+#include <sysexit.h>
+
+#include "ifconfig.h"
+
+static void wol_status(int s);
+static void setwol(const char *, int, int, const struct afswtch *);
+static void parse_args(const char *, struct if_wolopts *);
+static void parse_sopasswd(const char *, u_char *);
+static void unsetwol(const char *, int, int, const struct afswtch *);
+static void print_wol_events(uint32_t events);
+
+/*
+ * Print wake on lan capabilities and events the device currently heeds.
+ */
+static void
+wol_status(int s)
+{
+ struct ifreq ifr;
+
+ memset(&ifr, 0, sizeof(ifr));
+ strncpy(ifr.ifr_name, name, IFNAMSIZ);
+
+ if (ioctl(s, SIOCGIFWOLSUPP, &ifr) < 0)
+ /* Device does not support wake on lan */
+ return;
+
+ printf("\tsupported wake events:");
+ print_wol_events(ifr.ifr_wolopts.ifwol_supported);
+ printf("\n");
+
+ if (ioctl(s, SIOCGIFWOLOPTS, &ifr) < 0)
+ err(EX_USAGE, "SIOCGIFWOLOPTS");
+
+ if (ifr.ifr_wolopts.ifwol_events == 0)
+ return;
+
+ printf("\twill wake on:");
+ print_wol_events(ifr.ifr_wolopts.ifwol_events);
+ printf("\n");
```

```
+}
+
+static void
+print_wol_events(uint32_t events)
+{
+ if (events & IFWOL_WAKE_ON_UNICAST)
+ printf(" unicast");
+ if (events & IFWOL_WAKE_ON_MULTICAST)
+ printf(" multicast");
+ if (events & IFWOL_WAKE_ON_BROADCAST)
+ printf(" broadcast");
+ if (events & IFWOL_WAKE_ON_MAGIC) {
+ printf(" magic");
+ if (events & IFWOL_ENABLE_SOPASSWD)
+ printf("[SecureOn password]");
+ }
+}
+
+/*
+ * Set wake on lan events.
+ */
+static void
+setwol(const char *val, int d, int s, const struct ifswtch *afp)
+{
+ struct ifreq ifr;
+
+ memset(&ifr, 0, sizeof(ifr));
+ strncpy(ifr.ifr_name, name, IFNAMSIZ);
+
+ if (ioctl(s, SIOCGIFWOLSUPP, &ifr) < 0)
+ err(EX_USAGE, "device does not support wake on lan");
+
+ parse_args(val, &ifr.ifr_wolopts);
+ if (ioctl(s, SIOCSIFWOLOPTS, &ifr) < 0)
+ err(EX_USAGE, "SIOCSIFWOLOPTS");
+}
+
+/*
+ * Parse the argument string, which may contain one or more of the
+ * following:
+ *
+ * unicast,multicast,broadcast,magic,sopasswd:xxxxxxxxxxxxx,
+ *
+ * and fill the wolopts structure accordingly.
+ *
+ */
+static void
+parse_args(const char* args, struct if_wolopts *wolopts)
+{
+ uint32_t wol_events = 0;
+ char* opt;
```

```

+
+ for (opt = strdup(args); (opt = strtok(opt, ",")) != NULL; opt = NULL) {
+ if (strcmp(opt, "unicast") == 0)
+ wol_events |= IFWOL_WAKE_ON_UNICAST;
+ else if (strcmp(opt, "multicast") == 0)
+ wol_events |= IFWOL_WAKE_ON_MULTICAST;
+ else if (strcmp(opt, "broadcast") == 0)
+ wol_events |= IFWOL_WAKE_ON_BROADCAST;
+ else if (strcmp(opt, "magic") == 0)
+ wol_events |= IFWOL_WAKE_ON_MAGIC;
+ else if (strcmp(opt, "sopasswd") == 0)
+ errx(EX_USAGE, "no SecureOn password specified.");
+ else if (strncmp(opt, "sopasswd:", strlen("sopasswd:")) == 0) {
+ wol_events |= IFWOL_ENABLE_SOPASSWD;
+ parse_sopasswd(opt + strlen("sopasswd:"), wlopts->ifwol_sopasswd);
+ } else {
+ errx(EX_USAGE, "unknown wake event %s", opt);
+ }
+ }
+ free(opt);
+ wlopts->ifwol_events = wol_events;
+ }
+
+ /* SecureOn passwords are not like plain text passwords. Instead, they consist
+ * of 6 bytes (ie unsigned char). Try to prevent users from giving anything other
+ * than a string of six concatenated unsigned chars in hex as password.
+ */
+static void
+parse_sopasswd(const char *pw, u_char *dest) {
+ char substr[3];
+ int len, i, n;
+
+ len = strlen(pw) / 2;
+ if (len != 6)
+ errx(EX_USAGE, "Invalid SecureOn password.");
+
+ for (i = 0; i < len; i++) {
+ (void)strncpy(substr, pw, 2);
+ substr[2] = '\0';
+ if (sscanf(substr, "%x", &n) != 1)
+ errx(EX_USAGE, "Invalid SecureOn password.");
+ if (n < 0x0 || n > 0xff)
+ /* Should never happen, but just in case... */
+ errx(EX_USAGE, "Invalid SecureOn password.");
+ *dest++ = (u_char)n;
+ pw += 2;
+ }
+ }
+
+ /* Unset all wake on lan events.

```

```
+ */
+static void
+unsetwol(const char *val, int d, int s, const struct afswtch *afp)
+{
+ struct ifreq ifr;
+
+ memset(&ifr, 0, sizeof(ifr));
+ strncpy(ifr.ifr_name, name, IFNAMSIZ);
+
+ if (ioctl(s, SIOCGIFWOLSUPP, &ifr) < 0)
+ err(EX_USAGE, "device does not support wake on lan");
+
+ ifr.ifr_wolopts.ifwol_events = IFWOL_DISABLE;
+ if (ioctl(s, SIOCSIFWOLOPTS, &ifr) < 0)
+ err(EX_USAGE, "SIOCSIFWOLOPTS");
+}
+
+static struct cmd wol_cmds[] = {
+ DEF_CMD_ARG("wakeon", setwol),
+ DEF_CMD("-wakeon", 0, unsetwol)
+};
+static struct afswtch af_wol = {
+ .af_name = "af_wol",
+ .af_af = AF_UNSPEC,
+ .af_other_status = wol_status,
+};
+
+static __constructor void
+ifwol_ctor(void)
+{
+ #define N(a) (sizeof(a) / sizeof(a[0]))
+ int i;
+
+ for (i = 0; i < N(wol_cmds); i++)
+ cmd_register(&wol_cmds[i]);
+ af_register(&af_wol);
+ #undef N
+}
```

Index: sys/dev/nve/if_nve.c

=====
RCS file: /usr/local/ncvs/src/sys/dev/nve/if_nve.c,v

retrieving revision 1.7.2.11

diff -u -r1.7.2.11 if_nve.c

--- sys/dev/nve/if_nve.c 13 Sep 2006 15:15:57 -0000 1.7.2.11

+++ sys/dev/nve/if_nve.c 13 Jan 2007 12:21:03 -0000

@@ -177,6 +177,10 @@

static NV_SINT32 nve_oslockrelease(PNV_VOID, NV_SINT32, PNV_VOID);

static PNV_VOID nve_osreturnbufvirt(PNV_VOID, PNV_VOID);

+static void nve_enable_wol(struct nve_softc *);

+static void nve_get_wolopts(struct nve_softc *, struct if_wolopts *);


```
+ if (sc->wol_events == 0)
+ return;
+
+ if (sc->wol_events & IFWOL_WAKE_ON_MAGIC) {
+ pstate.ulPowerFlags = POWER_STATE_D3;
+ pstate.ulMagicPacketWakeUpFlags = POWER_STATE_ALL;
+ pstate.ulLinkChangeWakeUpFlags = 0;
+ pstate.ulPatternWakeUpFlags = 0;
+ sc->hwapi->pfnSetPowerState(sc->hwapi->pADCX, &pstate);
+ }
+}
+
+/*
+ * Write current wake on lan settings into an if_wolopts structure.
+ */
+static void
+nve_get_wolopts(struct nve_softc *sc, struct if_wolopts *wolopts)
+{
+ wolopts->ifwol_events = sc->wol_events;
+}
+
+/*
+ * Set wake on lan options.
+ */
+static int
+nve_set_wolopts(struct nve_softc *sc, struct if_wolopts *wolopts)
+{
+ if (wolopts->ifwol_events == IFWOL_DISABLE)
+ sc->wol_events = 0;
+ else {
+ if ((wolopts->ifwol_events & ~NVE_SUPPORTED_WOL_EVENTS) != 0)
+ return EINVAL;
+ sc->wol_events = wolopts->ifwol_events;
+ }
+}
+
+ return 0;
+}
```

Index: sys/dev/nve/if_nvereg.h

=====
RCS file: /usr/local/ncvs/src/sys/dev/nve/if_nvereg.h,v

retrieving revision 1.3.2.2.2.1

diff -u -u -r1.3.2.2.2.1 if_nvereg.h

--- sys/dev/nve/if_nvereg.h 7 Dec 2006 22:28:52 -0000 1.3.2.2.2.1

+++ sys/dev/nve/if_nvereg.h 13 Jan 2007 12:21:03 -0000

@@ -69,6 +69,8 @@

#define NVE_DEBUG_MII 0x0100

#define NVE_DEBUG_ALL 0xFFFF

+#define NVE_SUPPORTED_WOL_EVENTS IFWOL_WAKE_ON_MAGIC

+

#if NVE_DEBUG

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

```
#define DEBUGOUT(level, fmt, args...) if (NVE_DEBUG & level) \  
printf(fmt, ## args)  
@@ -143,6 +145,8 @@
```

```
struct mtx mtx;
```

```
+ uint32_t wol_events;
```

```
+
```

```
/* Stuff for dealing with the NVIDIA OS API */
```

```
struct callout ostimer;
```

```
PTIMER_FUNC ostimer_func;
```

```
Index: sys/net/if.c
```

```
=====  
RCS file: /usr/local/ncvs/src/sys/net/if.c,v
```

```
retrieving revision 1.234.2.17
```

```
diff -u -u -r1.234.2.17 if.c
```

```
--- sys/net/if.c 6 Oct 2006 20:26:05 -0000 1.234.2.17
```

```
+++ sys/net/if.c 13 Jan 2007 12:23:17 -0000
```

```
@@ -1461,6 +1461,7 @@
```

```
case SIOCSLIFPHYADDR:
```

```
case SIOCSIFMEDIA:
```

```
case SIOCSIFGENERIC:
```

```
+ case SIOCSIFWOLOPTS:
```

```
error = suser(td);
```

```
if (error)
```

```
return (error);
```

```
@@ -1482,6 +1483,8 @@
```

```
case SIOCGLIFPHYADDR:
```

```
case SIOCGIFMEDIA:
```

```
case SIOCGIFGENERIC:
```

```
+ case SIOCGIFWOLOPTS:
```

```
+ case SIOCGIFWOLSUPP:
```

```
if (ifp->if_ioctl == NULL)
```

```
return (EOPNOTSUPP);
```

```
IFF_LOCKGIANT(ifp);
```

```
Index: sys/net/if.h
```

```
=====  
RCS file: /usr/local/ncvs/src/sys/net/if.h,v
```

```
retrieving revision 1.96.2.4
```

```
diff -u -u -r1.96.2.4 if.h
```

```
--- sys/net/if.h 15 Feb 2006 03:37:15 -0000 1.96.2.4
```

```
+++ sys/net/if.h 5 May 2006 23:05:57 -0000
```

```
@@ -254,6 +254,28 @@
```

```
#define IFAN_DEPARTURE 1 /* interface departure */
```

```
/*
```

```
+ * Wake on Lan related options.
```

```
+ */
```

```
+struct if_wolopts {
```

```
+ uint32_t ifwol_supported; /* indicates wol capabilities */
```

```
+ uint32_t ifwol_events; /* indicates desired wake events */
```

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

10

```

+
+ /* Supported wake on lan events.
+ * A given device may not support all of these,
+ * or even support wake events not listed here.
+ * If you add wake more events, make to sure to teach
+ * ifconfig about them too. */
+#define IFWOL_DISABLE 0x01 /* clears all other events */
+#define IFWOL_WAKE_ON_UNICAST 0x02
+#define IFWOL_WAKE_ON_MULTICAST 0x04
+#define IFWOL_WAKE_ON_BROADCAST 0x08
+#define IFWOL_WAKE_ON_MAGIC 0x10 /* wake on Magic Packet(tm) */
+#define IFWOL_ENABLE_SOPASSWD 0x20 /* whether to set SecureOn password */
+
+
+ u_char ifwol_sopasswd[6]; /* SecureOn password */
+};
+
+/*
+ * Interface request structure used for socket
+ * ioctl's. All interface ioctl's must have parameter
+ * definitions which begin with ifr_name. The
+ @@ -265,6 +287,7 @@
+ struct sockaddr ifru_addr;
+ struct sockaddr ifru_dstaddr;
+ struct sockaddr ifru_broadaddr;
+ struct if_wolopts ifru_wolopts;
+ short ifru_flags[2];
+ short ifru_index;
+ int ifru_metric;
+ @@ -277,6 +300,7 @@
+ #define ifr_addr ifr_ifru.ifru_addr /* address */
+ #define ifr_dstaddr ifr_ifru.ifru_dstaddr /* other end of p-to-p link */
+ #define ifr_broadaddr ifr_ifru.ifru_broadaddr /* broadcast address */
+ #define ifr_wolopts ifr_ifru.ifru_wolopts /* wake on lan related options */
+ #define ifr_flags ifr_ifru.ifru_flags[0] /* flags (low 16 bits) */
+ #define ifr_flagshigh ifr_ifru.ifru_flags[1] /* flags (high 16 bits) */
+ #define ifr_metric ifr_ifru.ifru_metric /* metric */
+ Index: sys/pci/if_sis.c
+
+=====
+ RCS file: /usr/local/ncvs/src/sys/pci/if_sis.c,v
+ retrieving revision 1.132.2.7
+ diff -u -u -r1.132.2.7 if_sis.c
+ --- sys/pci/if_sis.c 17 Mar 2006 21:30:57 -0000 1.132.2.7
+ +++ sys/pci/if_sis.c 5 May 2006 23:05:57 -0000
+ @@ -126,6 +126,10 @@
+ static void sis_startl(struct ifnet *);
+ static void sis_stop(struct sis_softc *);
+ static void sis_watchdog(struct ifnet *);
+ +static void sis_get_wolopts(struct sis_softc *, struct if_wolopts *);
+ +static int sis_set_wolopts(struct sis_softc *, struct if_wolopts *);
+ +static void sis_enable_wol(struct sis_softc *);
+ +static uint32_t sis_translate_wol_events(uint32_t);

```

```
#ifdef SIS_USEIOWSPACE
#define SIS_RES SYS_RES_IOPORT
@@ -170,7 +174,7 @@
static void
sis_dma_map_ring(void *arg, bus_dma_segment_t *segs, int nseg, int error)
{
- u_int32_t *p;
+ uint32_t *p;

p = arg;
*p = segs->ds_addr;
@@ -258,7 +262,7 @@
sis_eeprom_getword(struct sis_softc *sc, int addr, uint16_t *dest)
{
int i;
- u_int16_t word = 0;
+ uint16_t word = 0;

/* Force EEPROM to idle state. */
sis_eeprom_idle(sc);
@@ -301,11 +305,11 @@
sis_read_eeprom(struct sis_softc *sc, caddr_t dest, int off, int cnt, int swap)
{
int i;
- u_int16_t word = 0, *ptr;
+ uint16_t word = 0, *ptr;

for (i = 0; i < cnt; i++) {
sis_eeprom_getword(sc, off + i, &word);
- ptr = (u_int16_t *) (dest + (i * 2));
+ ptr = (uint16_t *) (dest + (i * 2));
if (swap)
*ptr = ntohs(word);
else
@@ -354,7 +358,7 @@
sis_read_cmos(struct sis_softc *sc, device_t dev, caddr_t dest, int off, int cnt)
{
device_t bridge;
- u_int8_t reg;
+ uint8_t reg;
int i;
bus_space_tag_t btag;

@@ -383,7 +387,7 @@
static void
sis_read_mac(struct sis_softc *sc, device_t dev, caddr_t dest)
{
- u_int32_t filtsave, csrsave;
+ uint32_t filtsave, csrsave;
```

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

```
filtsave = CSR_READ_4(sc, SIS_RXFILT_CTL);
csrsave = CSR_READ_4(sc, SIS_CSR);
@@ -394,11 +398,11 @@
CSR_WRITE_4(sc, SIS_RXFILT_CTL, filtsave & ~SIS_RXFILTCTL_ENABLE);

CSR_WRITE_4(sc, SIS_RXFILT_CTL, SIS_FILTADDR_PAR0);
- ((u_int16_t *)dest)[0] = CSR_READ_2(sc, SIS_RXFILT_DATA);
+ ((uint16_t *)dest)[0] = CSR_READ_2(sc, SIS_RXFILT_DATA);
CSR_WRITE_4(sc, SIS_RXFILT_CTL, SIS_FILTADDR_PAR1);
- ((u_int16_t *)dest)[1] = CSR_READ_2(sc, SIS_RXFILT_DATA);
+ ((uint16_t *)dest)[1] = CSR_READ_2(sc, SIS_RXFILT_DATA);
CSR_WRITE_4(sc, SIS_RXFILT_CTL, SIS_FILTADDR_PAR2);
- ((u_int16_t *)dest)[2] = CSR_READ_2(sc, SIS_RXFILT_DATA);
+ ((uint16_t *)dest)[2] = CSR_READ_2(sc, SIS_RXFILT_DATA);

CSR_WRITE_4(sc, SIS_RXFILT_CTL, filtsave);
CSR_WRITE_4(sc, SIS_CSR, csrsave);
@@ -731,7 +735,7 @@
{
struct ifnet *ifp;
struct ifmultiaddr *ifma;
- u_int32_t h = 0, i, filtsave;
+ uint32_t h = 0, i, filtsave;
int bit, index;

ifp = sc->sis_ifp;
@@ -782,8 +786,8 @@
{
struct ifnet *ifp;
struct ifmultiaddr *ifma;
- u_int32_t h, i, n, ctl;
- u_int16_t hashes[16];
+ uint32_t h, i, n, ctl;
+ uint16_t hashes[16];

ifp = sc->sis_ifp;

@@ -984,7 +988,7 @@
* Why? Who the hell knows.
*/
{
- u_int16_t tmp[4];
+ uint16_t tmp[4];

sis_read_eeprom(sc, (caddr_t)&tmp,
NS_EE_NODEADDR, 4, 0);
@@ -1406,7 +1410,7 @@
struct ifnet *ifp;
struct sis_desc *cur_rx;
int total_len = 0;
- u_int32_t rxstat;
```

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

```
+ uint32_t rxstat;

SIS_LOCK_ASSERT(sc);

@@ -1501,7 +1505,7 @@
sis_txeof(struct sis_softc *sc)
{
struct ifnet *ifp;
- u_int32_t idx;
+ uint32_t idx;

SIS_LOCK_ASSERT(sc);
ifp = sc->sis_ifp;
@@ -1605,7 +1609,7 @@
sis_startl(ifp);

if (sc->rxcycles > 0 || cmd == POLL_AND_CHECK_STATUS) {
- u_int32_t status;
+ uint32_t status;

/* Reading the ISR register clears all interrupts. */
status = CSR_READ_4(sc, SIS_ISR);
@@ -1631,7 +1635,7 @@
{
struct sis_softc *sc;
struct ifnet *ifp;
- u_int32_t status;
+ uint32_t status;

sc = arg;
ifp = sc->sis_ifp;
@@ -1785,7 +1789,7 @@
{
struct sis_softc *sc;
struct mbuf *m_head = NULL;
- u_int32_t idx, queued = 0;
+ uint32_t idx, queued = 0;

sc = ifp->if_softc;

@@ -1872,23 +1876,23 @@
if (sc->sis_type == SIS_TYPE_83815) {
CSR_WRITE_4(sc, SIS_RXFILT_CTL, NS_FILTADDR_PAR0);
CSR_WRITE_4(sc, SIS_RXFILT_DATA,
- ((u_int16_t *)IFP2ENADDR(sc->sis_ifp))[0]);
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[0]);
CSR_WRITE_4(sc, SIS_RXFILT_CTL, NS_FILTADDR_PAR1);
CSR_WRITE_4(sc, SIS_RXFILT_DATA,
- ((u_int16_t *)IFP2ENADDR(sc->sis_ifp))[1]);
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[1]);
CSR_WRITE_4(sc, SIS_RXFILT_CTL, NS_FILTADDR_PAR2);
```

```
CSR_WRITE_4(sc, SIS_RXFILT_DATA,
- ((u_int16_t *)IFP2ENADDR(sc->sis_ifp))[2]);
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[2]);
} else {
CSR_WRITE_4(sc, SIS_RXFILT_CTL, SIS_FILTADDR_PAR0);
CSR_WRITE_4(sc, SIS_RXFILT_DATA,
- ((u_int16_t *)IFP2ENADDR(sc->sis_ifp))[0]);
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[0]);
CSR_WRITE_4(sc, SIS_RXFILT_CTL, SIS_FILTADDR_PAR1);
CSR_WRITE_4(sc, SIS_RXFILT_DATA,
- ((u_int16_t *)IFP2ENADDR(sc->sis_ifp))[1]);
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[1]);
CSR_WRITE_4(sc, SIS_RXFILT_CTL, SIS_FILTADDR_PAR2);
CSR_WRITE_4(sc, SIS_RXFILT_DATA,
- ((u_int16_t *)IFP2ENADDR(sc->sis_ifp))[2]);
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[2]);
}

/* Init circular TX/RX lists. */
@@ -2162,6 +2166,21 @@
}
#endif /* DEVICE_POLLING */
break;
+ case SIOCGIFWOLSUPP:
+ ifr->ifr_wolopts.ifwol_supported = NS_SUPPORTED_WOL_EVENTS;
+ error = 0;
+ break;
+ case SIOCGIFWOLOPTS:
+ SIS_LOCK(sc);
+ sis_get_wolopts(sc, &ifr->ifr_wolopts);
+ SIS_UNLOCK(sc);
+ error = 0;
+ break;
+ case SIOCSIFWOLOPTS:
+ SIS_LOCK(sc);
+ error = sis_set_wolopts(sc, &ifr->ifr_wolopts);
+ SIS_UNLOCK(sc);
+ break;
default:
error = ether_ioctl(ifp, command, data);
break;
@@ -2271,9 +2290,141 @@
SIS_LOCK(sc);
sis_reset(sc);
sis_stop(sc);
+ sis_enable_wol(sc);
SIS_UNLOCK(sc);
}

+/*
+ * Translate wake on lan events defined in if.h
```

```
+ * into flags the chip understands.
+ */
+static uint32_t
+sis_translate_wol_events(uint32_t wol_events)
+{
+ uint32_t sis_wol_events = 0;
+
+ if (wol_events & IFWOL_WAKE_ON_UNICAST)
+ sis_wol_events |= NS_WCSR_WAKE_UCAST;
+ if (wol_events & IFWOL_WAKE_ON_MULTICAST)
+ sis_wol_events |= NS_WCSR_WAKE_MCAST;
+ if (wol_events & IFWOL_WAKE_ON_BROADCAST)
+ sis_wol_events |= NS_WCSR_WAKE_BCAST;
+ if (wol_events & IFWOL_WAKE_ON_MAGIC)
+ sis_wol_events |= NS_WCSR_WAKE_MAGIC;
+
+ return sis_wol_events;
+}
+
+/*
+ * Write current wake on lan settings into an if_wolopts structure.
+ * Note that the sopasswd field in the structure is cleared, because
+ * the password is confidential.
+ */
+static void
+sis_get_wolopts(struct sis_softc *sc, struct if_wolopts *wolopts)
+{
+ int i;
+
+ SIS_LOCK_ASSERT(sc);
+
+ wolopts->ifwol_events = sc->ns_wol_events;
+
+ /* Do not disclose Secure On password. */
+ #define N(a) (sizeof(a) / sizeof(a[0]))
+ for (i = 0; i < N(wolopts->ifwol_sopasswd); i++)
+ wolopts->ifwol_sopasswd[i] = '\0';
+ #undef N
+}
+
+/*
+ * Set wake on lan options.
+ */
+static int
+sis_set_wolopts(struct sis_softc *sc, struct if_wolopts *wolopts)
+{
+ SIS_LOCK_ASSERT(sc);
+
+ /* FIXME: handle sopasswd */
+
+ if (wolopts->ifwol_events == IFWOL_DISABLE)
```

```
+ sc->ns_wol_events = 0;
+ else {
+ if ((wolopts->ifwol_events & ~NS_SUPPORTED_WOL_EVENTS) != 0)
+ return EINVAL;
+ sc->ns_wol_events = wolopts->ifwol_events;
+ }
+
+ return 0;
+}
+
+/*
+ * Enable Wake On Lan on the DP83815,
+ * if any wake on lan options have been set.
+ */
+static void
+sis_enable_wol(struct sis_softc *sc)
+{
+ SIS_LOCK_ASSERT(sc);
+
+ if (sc->sis_type != SIS_TYPE_83815)
+ return;
+
+ /* Check whether any wake on lan events have been set. */
+ if (sc->ns_wol_events == 0)
+ return;
+
+ /*
+ * Configure the receive filter to accept potential wake packets,
+ * configure wake events and enter low-power state.
+ */
+
+ /* Stop receiver. */
+ SIS_SETBIT(sc, SIS_CSR, SIS_CSR_RX_DISABLE);
+
+ /* Reset receive pointer */
+ CSR_WRITE_4(sc, SIS_RX_LISTPTR, 0);
+
+ /* Re-enable receiver (now in "silent receive mode.") */
+ SIS_SETBIT(sc, SIS_CSR, SIS_CSR_RX_ENABLE);
+
+ /* Clear receive filter register, so that the enable bit is unset.
+ * Other bits in this register can only be configured while the enable
+ * bit is zero. */
+ CSR_WRITE_4(sc, SIS_RXFILT_CTL, 0);
+
+ /*
+ * Accept unicast packets. The datasheet seems to be inaccurate.
+ * It suggests simply setting the unicast bit in NS_RXFILTCTL,
+ * but this does not seem to work. Instead, we "perfect match"
+ * our own mac address, which makes the rx filter accept unicast
+ * packets. (section below copy pasted from sis_initl routine)
```

```

+ */
+ CSR_WRITE_4(sc, SIS_RXFILT_CTL, NS_FILTADDR_PAR0);
+ CSR_WRITE_4(sc, SIS_RXFILT_DATA,
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[0]);
+ CSR_WRITE_4(sc, SIS_RXFILT_CTL, NS_FILTADDR_PAR1);
+ CSR_WRITE_4(sc, SIS_RXFILT_DATA,
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[1]);
+ CSR_WRITE_4(sc, SIS_RXFILT_CTL, NS_FILTADDR_PAR2);
+ CSR_WRITE_4(sc, SIS_RXFILT_DATA,
+ ((uint16_t *)IFP2ENADDR(sc->sis_ifp))[2]);
+ SIS_SETBIT(sc, SIS_RXFILT_CTL, NS_RXFILTCTL_PERFECT);
+
+ /* Allow broadcast and multicast packets, too. */
+ SIS_SETBIT(sc, SIS_RXFILT_CTL, SIS_RXFILTCTL_BROAD);
+ SIS_SETBIT(sc, SIS_RXFILT_CTL, SIS_RXFILTCTL_ALLMULTI);
+
+ /* Re-enable RX filter. */
+ SIS_SETBIT(sc, SIS_RXFILT_CTL, SIS_RXFILTCTL_ENABLE);
+
+ /* Configure wake on lan events */
+ CSR_WRITE_4(sc, NS_WCSR, sis_translate_wol_events(sc->ns_wol_events));
+
+ /* Set appropriate power state, so the card stays active
+ * after system shutdown. */
+ CSR_WRITE_4(sc, NS_CLKRUN, NS_CLKRUN_PMESTS | NS_CLKRUN_PMEENB);
+ }
+
static device_method_t sis_methods[] = {
/* Device interface */
DEVMETHOD(device_probe, sis_probe),
Index: sys/pci/if_sisreg.h
=====
RCS file: /usr/local/ncvs/src/sys/pci/if_sisreg.h,v
retrieving revision 1.33.2.1
diff -u -r1.33.2.1 if_sisreg.h
--- sys/pci/if_sisreg.h 29 Sep 2005 18:52:21 -0000 1.33.2.1
+++ sys/pci/if_sisreg.h 6 May 2006 10:59:50 -0000
@@ -77,6 +77,7 @@
/* NS DP83815/6 registers */
#define NS_IHR 0x1C
#define NS_CLKRUN 0x3C
+#define NS_WCSR 0x40
#define NS_SRR 0x58
#define NS_BMCR 0x80
#define NS_BMSR 0x84
@@ -463,6 +464,7 @@
#endif
int in_tick;
struct mtx sis_mtx;
+ uint32_t ns_wol_events;
};

```

```
#define SIS_LOCK(_sc) mtx_lock(&(_sc)->sis_mtx)
@@ -523,3 +525,17 @@
#define SIS_PSTATE_D3 0x0003
#define SIS_PME_EN 0x0010
#define SIS_PME_STATUS 0x8000
+
+/* DP83815 pci config space power management register */
+#define NS_PMCSR 0x44
+
+/* DP83815 Wake On Lan Command/Status register */
+#define NS_WCSR_WAKE_UCAST 0x00000002
+#define NS_WCSR_WAKE_MCAST 0x00000004
+#define NS_WCSR_WAKE_BCAST 0x00000008
+#define NS_WCSR_WAKE_MAGIC 0x00000200
+
+/* FIXME: handle sopasswd */
+#define NS_SUPPORTED_WOL_EVENTS (IFWOL_WAKE_ON_UNICAST |
IFWOL_WAKE_ON_MULTICAST \
+ | IFWOL_WAKE_ON_BROADCAST | IFWOL_WAKE_ON_MAGIC)
+
Index: sys/pci/if_vr.c
```

```
=====
RCS file: /usr/local/ncvs/src/sys/pci/if_vr.c,v
retrieving revision 1.104.2.6
diff -u -u -r1.104.2.6 if_vr.c
--- sys/pci/if_vr.c 17 Mar 2006 21:30:57 -0000 1.104.2.6
+++ sys/pci/if_vr.c 13 Jun 2007 22:38:52 -0000
@@ -169,6 +169,10 @@
static int vr_list_rx_init(struct vr_softc *);
static int vr_list_tx_init(struct vr_softc *);

+static int vr_set_wolopts(struct vr_softc *, struct if_wolopts *);
+static void vr_get_wolopts(struct vr_softc *, struct if_wolopts *);
+static void vr_enable_wol(struct vr_softc *);
+
+#ifdef VR_USEIOSPACE
#define VR_RES SYS_RES_IOPORT
#define VR_RID VR_PCI_LOIO
@@ -710,7 +714,7 @@
#endif

/*
- * Windows may put the chip in suspend mode when it
+ * Windows or WOL may put the chip in suspend mode when it
 * shuts down. Be sure to kick it in the head to wake it
 * up again.
 */
@@ -761,6 +765,13 @@

sc->suspended = 0;
```

```

+ /* Check Wake on Lan support. */
+ if (sc->vr_revid >= REV_ID_VT6102 ) {
+ sc->vr_wol support = 1;
+ if (sc->vr_revid >= REV_ID_VT6105_B0)
+ sc->vr_wol6patterns = 1;
+ }
+
+ /* Hook interrupt last to avoid having to lock softc */
error = bus_setup_intr(dev, sc->vr_irq, INTR_TYPE_NET | INTR_MPSAFE,
vr_intr, sc, &sc->vr_intrhand);
@@ -1618,6 +1629,21 @@
}
#endif /* DEVICE_POLLING */
break;
+ case SIOCGIFWOLSUPP:
+ ifr->ifr_wolopts.ifwol_supported = VR_SUPPORTED_WOL_EVENTS;
+ error = 0;
+ break;
+ case SIOCGIFWOLOPTS:
+ VR_LOCK(sc);
+ vr_get_wolopts(sc, &ifr->ifr_wolopts);
+ VR_UNLOCK(sc);
+ error = 0;
+ break;
+ case SIOCSIFWOLOPTS:
+ VR_LOCK(sc);
+ error = vr_set_wolopts(sc, &ifr->ifr_wolopts);
+ VR_UNLOCK(sc);
+ break;
default:
error = ether_ioctl(ifp, command, data);
break;
@@ -1702,6 +1728,92 @@
static void
vr_shutdown(device_t dev)
{
+ struct vr_softc *sc;

+ sc = device_get_softc(dev);
+ VR_LOCK(sc);
+ vr_enable_wol(sc);
+ VR_UNLOCK(sc);
vr_detach(dev);
}
+
+static void
+vr_enable_wol(struct vr_softc *sc)
+{
+ VR_LOCK_ASSERT(sc);
+

```

```

+ /* Check whether wake on lan is available
+ * and whether events have been set. */
+ if (!sc->vr_wolsupport || sc->vr_wolevents == 0)
+ return;
+
+ /* Set the chip to power state D0 */
+ VR_CLRBIT(sc, VR_STICKHW, (VR_STICKHW_DS0|VR_STICKHW_DS1));
+
+ /* Clear WOL configuration */
+ CSR_WRITE_1(sc, VR_WOLCRCLR, 0xFF);
+ if (sc->vr_wol6patterns)
+ CSR_WRITE_1(sc, VR_WOLCRCLR1, 0x03);
+
+ /* Clear power-event status. */
+ CSR_WRITE_1(sc, VR_PWRCSRCLR, 0xFF);
+
+ /* Don't use extra patterns. */
+ if (sc->vr_wol6patterns)
+ CSR_WRITE_1(sc, VR_WOLCGCLR, 0x04);
+
+ /* Set unicast wake event if applicable. */
+ if (sc->vr_wolevents & IFWOL_WAKE_ON_UNICAST)
+ VR_SETBIT(sc, VR_WOLCRSET, VR_WAKE_UCAST);
+
+ /* Set magic wake event if applicable. */
+ if (sc->vr_wolevents & IFWOL_WAKE_ON_MAGIC) {
+ VR_SETBIT(sc, VR_WOLCRSET, VR_WAKE_MAGIC);
+ /* enable EEPROM-controlled wake-up */
+ VR_SETBIT(sc, VR_CONFIG, 0x03);
+ }
+ #if 0
+ /* Set broadcast/multicast wake event if applicable. */
+ /* Does not work for some reason :( */
+ if (sc->vr_wolevents & IFWOL_WAKE_ON_BROADCAST ||
+ sc->vr_wolevents & IFWOL_WAKE_ON_MULTICAST)
+ CSR_WRITE_1(sc, VR_WOLCGSET, VR_WAKE_BMCASET);
+ #endif
+ /* Enable Wake On Lan. */
+ CSR_WRITE_1(sc, VR_PWCFGSET, 0x01);
+ VR_SETBIT(sc, VR_STICKHW, VR_STICKHW_WOL_ENB);
+
+ /* Set power state to D3 */
+ VR_SETBIT(sc, VR_STICKHW, (VR_STICKHW_DS0|VR_STICKHW_DS1));
+ }
+
+ /*
+ * Write current wake on lan settings into an if_wolopts structure.
+ */
+ static void
+ vr_get_wolopts(struct vr_softc *sc, struct if_wolopts *wolopts)

```

```
+{
+ VR_LOCK_ASSERT(sc);
+ wolopts->ifwol_events = sc->vr_wolevents;
+}
+
+/*
+ * Set wake on lan options.
+ */
+static int
+vr_set_wolopts(struct vr_softc *sc, struct if_wolopts *wolopts)
+{
+ VR_LOCK_ASSERT(sc);
+
+ if (wolopts->ifwol_events == IFWOL_DISABLE)
+ sc->vr_wolevents = 0;
+ else {
+ if ((wolopts->ifwol_events & ~VR_SUPPORTED_WOL_EVENTS) != 0)
+ return EINVAL;
+ sc->vr_wolevents = wolopts->ifwol_events;
+ }
+
+ return 0;
+}
+
```

Index: sys/pci/if_vrreg.h

=====
RCS file: /usr/local/ncvs/src/sys/pci/if_vrreg.h,v

retrieving revision 1.22.2.1

diff -u -u -r1.22.2.1 if_vrreg.h

--- sys/pci/if_vrreg.h 8 Nov 2005 16:05:56 -0000 1.22.2.1

+++ sys/pci/if_vrreg.h 13 Jun 2007 22:38:19 -0000

@@ -283,6 +283,21 @@

#define VR_STICKHW_WOL_STS 0x08

#define VR_STICKHW_LEGWOL_ENB 0x80

/* Wake on Lan definitions (snooped from Linux driver) */

+#define VR_WOLCRSET 0xA0

+#define VR_PWCFGSET 0xA1

+#define VR_WOLCGSET 0xA3

+#define VR_WOLCRCLR 0xA4

+#define VR_WOLCRCLR1 0xA6

+#define VR_WOLCGCLR 0xA7

+#define VR_PWRCSRCLR 0xAC

+#define VR_WAKE_UCAST 0x10

+#define VR_WAKE_MAGIC 0x20

+#define VR_WAKE_BMCAST 0x30

+#define VR_WAKE_LINKON 0x40

+#define VR_WAKE_LINKOFF 0x80

+#define VR_SUPPORTED_WOL_EVENTS (IFWOL_WAKE_ON_UNICAST |
IFWOL_WAKE_ON_MAGIC)

+

```

/*
 * BCR0 register bits. (At least for the VT6102 chip.)
 */
@@ -471,6 +486,10 @@
#ifdef DEVICE_POLLING
int rxcycles;
#endif
+ int vr_wol_support; /* Chip supports WOL. */
+ uint32_t vr_wol_events; /* Wake on Lan status */
+ /* some chips have 6 "patterns" for WOL instead of 4 */
+ int vr_wol6_patterns;
};

#define VR_F_RESTART 0x01 /* Restart unit on next tick */
@@ -545,10 +564,14 @@
#define REV_ID_VT3065_A 0x40
#define REV_ID_VT3065_B 0x41
#define REV_ID_VT3065_C 0x42
+#define REV_ID_VT6102 0x40
#define REV_ID_VT6102_APOLLO 0x74
#define REV_ID_VT3106 0x80
#define REV_ID_VT3106_J 0x80 /* 0x80-0x8F */
#define REV_ID_VT3106_S 0x90 /* 0x90-0xA0 */
+#define REV_ID_VT6105 0x80
+#define REV_ID_VT6105_B0 0x83
+

/*
 * PCI low memory base and low I/O base register, and
Index: sys/pci/if_xl.c
=====
RCS file: /usr/local/ncvs/src/sys/pci/if_xl.c,v
retrieving revision 1.190.2.10
diff -u -u -r1.190.2.10 if_xl.c
--- sys/pci/if_xl.c 17 Aug 2006 00:13:07 -0000 1.190.2.10
+++ sys/pci/if_xl.c 13 Jun 2007 23:00:35 -0000
@@ -249,6 +249,9 @@
static void xl_shutdown(device_t);
static int xl_suspend(device_t);
static int xl_resume(device_t);
+static void xl_get_wolopts(struct xl_softc *, struct if_wolopts *);
+static int xl_set_wolopts(struct xl_softc *, struct if_wolopts *);
+static void xl_enable_wol(device_t dev);

#ifdef DEVICE_POLLING
static void xl_poll(struct ifnet *ifp, enum poll_cmd cmd, int count);
@@ -3218,6 +3221,25 @@
ifp->if_hwassist = 0;
XL_UNLOCK(sc);
break;
+ case SIOCGIFWOLSUPP:

```

```
+ if (sc->xl_type == XL_TYPE_905B)
+ ifr->ifr_wolopts.ifwol_supported =
+ XL_SUPPORTED_WOL_EVENTS;
+ else
+ ifr->ifr_wolopts.ifwol_supported = 0;
+ error = 0;
+ break;
+ case SIOCGIFWOLOPTS:
+ XL_LOCK(sc);
+ xl_get_wolopts(sc, &ifr->ifr_wolopts);
+ XL_UNLOCK(sc);
+ error = 0;
+ break;
+ case SIOCSIFWOLOPTS:
+ XL_LOCK(sc);
+ error = xl_set_wolopts(sc, &ifr->ifr_wolopts);
+ XL_UNLOCK(sc);
+ break;
default:
error = ether_ioctl(ifp, command, data);
break;
@@ -3348,6 +3370,7 @@
XL_LOCK(sc);
xl_reset(sc);
xl_stop(sc);
+ xl_enable_wol(dev);
XL_UNLOCK(sc);
}

@@ -3384,3 +3407,80 @@

return (0);
}
+
+/*
+ * Write current wake on lan settings into an if_wolopts structure.
+ */
+static void
+xl_get_wolopts(struct xl_softc *sc, struct if_wolopts *wolopts)
+{
+ XL_LOCK_ASSERT(sc);
+
+ if (sc->xl_type == XL_TYPE_905B)
+ wolopts->ifwol_events = sc->xl_wol_events;
+ else
+ wolopts->ifwol_events = 0;
+}
+
+/*
+ * Set wake on lan options.
+ */
```

```

+static int
+xl_set_wolopts(struct xl_softc *sc, struct if_wolopts *wolopts)
+{
+ XL_LOCK_ASSERT(sc);
+
+ if (sc->xl_type != XL_TYPE_905B)
+ return ENOTSUP;
+
+ if (wolopts->ifwol_events == IFWOL_DISABLE)
+ sc->xl_wol_events = 0;
+ else {
+ if ((wolopts->ifwol_events & ~XL_SUPPORTED_WOL_EVENTS) != 0)
+ return EINVAL;
+ sc->xl_wol_events = wolopts->ifwol_events;
+ }
+
+ return 0;
+}
+
+/*
+ * Enable Wake On Lan if any wake on lan options have been set.
+ */
+static void
+xl_enable_wol(device_t dev)
+{
+ u_int8_t rxfilt;
+ struct xl_softc *sc;
+
+ sc = device_get_softc(dev);
+
+ XL_LOCK_ASSERT(sc);
+
+ if (sc->xl_type != XL_TYPE_905B)
+ return;
+
+ /* Check whether any wake on lan events have been set. */
+ if (sc->xl_wol_events == 0)
+ return;
+
+ /* Configure wake on lan events. */
+ XL_SEL_WIN(7);
+ if (sc->xl_wol_events & IFWOL_WAKE_ON_MAGIC)
+ CSR_WRITE_2(sc, XL_W7_BM_WOL, XL_WAKE_ON_MAGIC);
+
+ /* Configure the receive filter to accept WOL packets.
+ * We want to receive everything. */
+ XL_SEL_WIN(5);
+ rxfilt = CSR_READ_1(sc, XL_W5_RX_FILTER);
+ CSR_WRITE_1(sc, XL_W5_RX_FILTER,
+ rxfilt | XL_RXFILTER_INDIVIDUAL | XL_RXFILTER_ALLMULTI
+ | XL_RXFILTER_BROADCAST | XL_RXFILTER_ALLFRAMES);

```

```
+
+ /* Make sure reciever is enabled. */
+ CSR_WRITE_2(sc, XL_COMMAND, XL_CMD_RX_ENABLE);
+
+ /* Set appropriate power state, so the card stays active
+ * after system shutdown. */
+ pci_set_powerstate(dev, PCI_POWERSTATE_D3);
+}
Index: sys/pci/if_xlreg.h
```

RCS file: /usr/local/ncvs/src/sys/pci/if_xlreg.h,v

retrieving revision 1.55.2.1

diff -u -u -r1.55.2.1 if_xlreg.h

--- sys/pci/if_xlreg.h 26 Aug 2005 14:46:22 -0000 1.55.2.1

+++ sys/pci/if_xlreg.h 13 Jun 2007 23:12:49 -0000

@@ -408,6 +408,7 @@

#define XL_W7_BM_LEN 0x06

#define XL_W7_BM_STATUS 0x0B

#define XL_W7_BM_TIMER 0x0A

+#define XL_W7_BM_WOL 0x0C

/*

* bus master control registers

@@ -611,6 +612,7 @@

#ifdef DEVICE_POLLING

int rxcycles;

#endif

+ uint32_t xl_wol_events;

};

#define XL_LOCK(_sc) mtx_lock(&(_sc)->xl_mtx)

@@ -739,3 +741,7 @@

#ifndef IFM_10_FL

#define IFM_10_FL 13 /* 10baseFL - Fiber */

#endif

+

+#define XL_WAKE_ON_MAGIC 0x0002

+#define XL_SUPPORTED_WOL_EVENTS IFWOL_WAKE_ON_MAGIC

+

Index: sys/sys/sockio.h

RCS file: /usr/local/ncvs/src/sys/sys/sockio.h,v

retrieving revision 1.28.2.1

diff -u -u -r1.28.2.1 sockio.h

--- sys/sys/sockio.h 15 Feb 2006 03:37:15 -0000 1.28.2.1

+++ sys/sys/sockio.h 5 May 2006 23:05:58 -0000

@@ -117,4 +117,11 @@

#define SIOCIFDESTROY_IOWR('i', 121, struct ifreq) /* destroy clone if */

#define SIOCIFGCLONERS_IOWR('i', 120, struct if_clonereq) /* get cloners */

+#define SIOCGIFWOLOPTS_IOWR('i', 124, struct ifreq) /* get wake on lan

Re: kern/83807: [sis] [patch] if_sis: Wake On Lan support for FreeBSD

```
+ options */
+#define SIOCSIFWOLOPTS _IOW('i', 125, struct ifreq) /* set wake on lan
+ options */
+#define SIOCGIFWOLSUPP _IOWR('i', 126, struct ifreq) /* get wake on lan
+ modes supported by
+ device */
#endif /* !_SYS_SOCKIO_H_ */
```

Attachment: pgpJG8LfAVeey.pgp

Description: PGP signature