Open Source Used In IP Phone 3905 9.4(1)SR4

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.

Text Part Number: 78EE117C99-1838157167

This document contains licenses and notices for open source software used in this product. With respect to the free/open source software listed in this document, if you have any questions or wish to receive a copy of any source code to which you may be entitled under the applicable free/open source license(s) (such as the GNU Lesser/General Public License), please submit this form.

In your requests please include the following reference number 78EE117C99-1838157167

Contents

1.1 uclibc 0.9.30

1.1.1 Available under license :

```
/*
* FreeSec: libcrypt for NetBSD
*
* Copyright (c) 1994 David Burren
* All rights reserved.
*
* Adapted for FreeBSD-2.0 by Geoffrey M. Rehmet
* this file should now *only* export crypt(), in order to make
* binaries of libcrypt exportable from the USA
*
* Adapted for FreeBSD-4.0 by Mark R V Murray
* this file should now *only* export crypt_des(), in order to make
* a module that can be optionally included in libcrypt.
*
* 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. Neither the name of the author nor the names of other contributors
*    may be used to endorse or promote products derived from this software
*    without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "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 OR CONTRIBUTORS 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.
*
* This is an original implementation of the DES and the crypt(3) interfaces
* by David Burren <davidb@werj.com.au>.
*
* An excellent reference on the underlying algorithm (and related
```

```
* algorithms) is:
*
* B. Schneier, Applied Cryptography: protocols, algorithms,
* and source code in C, John Wiley & Sons, 1994.
*
* Note that in that book's description of DES the lookups for the initial,
* pbox, and final permutations are inverted (this has been brought to the
* attention of the author).  A list of errata for this book has been
* posted to the sci.crypt newsgroup by the author and is available for FTP.
*
* ARCHITECTURE ASSUMPTIONS:
* It is assumed that the 8-byte arrays passed by reference can be
* addressed as arrays of u_int32_t's (ie. the CPU is not picky about
* alignment).
*/
/*
* MD5C.C - RSA Data Security, Inc., MD5 message-digest algorithm
*
* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
* rights reserved.
*
* License to copy and use this software is granted provided that it
* is identified as the "RSA Data Security, Inc. MD5 Message-Digest
* Algorithm" in all material mentioning or referencing this software
* or this function.
*
* License is also granted to make and use derivative works provided
* that such works are identified as "derived from the RSA Data
* Security, Inc. MD5 Message-Digest Algorithm" in all material
* mentioning or referencing the derived work.
*
* RSA Data Security, Inc. makes no representations concerning either
* the merchantability of this software or the suitability of this
* software for any particular purpose. It is provided "as is"
* without express or implied warranty of any kind.
*
* These notices must be retained in any copies
 of any part of this
* documentation and/or software.
* /
/*
* Copyright (c) 1994-2000 Eric Youngdale, Peter MacDonald, David Engel,
* Hongjiu Lu and Mitch D'Souza
*
* 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. The name of the above contributors may not be
*   used to endorse or promote products derived from this software
*   without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE CONTRIBUTORS ``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 CONTRIBUTORS 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.
*/
GNU LESSER GENERAL PUBLIC LICENSE
       Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
   51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

      Preamble

 The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

 This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to
 use it. You
can use it too, but we suggest you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

 When we speak of free software, we are referring to freedom of use,

not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute
copies of the library, whether gratis
or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems
that might be
introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using

a shared library, the combination of the two is legally speaking a combined
work, a derivative of the original library. The ordinary
General Public License therefore permits such linking only if the
entire combination fits its criteria of freedom. The Lesser General
Public License permits more lax criteria for linking other code with
the library.

We call this license the "Lesser" General Public License because it
does Less to protect the user's freedom than the ordinary General
Public License. It also provides other free software developers Less
of an advantage over competing non-free programs. These disadvantages
are the reason we use the ordinary General Public License for many
libraries. However, the Lesser license provides advantages in certain
special circumstances.

For example, on rare occasions, there may be a special need to
encourage the widest possible use of a certain library, so that it becomes
a de-facto standard. To achieve this, non-free programs must be
allowed to use the library. A more frequent case
is that a free
library does the same job as widely used non-free libraries. In this
case, there is little to gain by limiting the free library to free
software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free
programs enables a greater number of people to use a large body of
free software. For example, permission to use the GNU C Library in
non-free programs enables many more people to use the whole GNU
operating system, as well as its variant, the GNU/Linux operating
system.

Although the Lesser General Public License is Less protective of the
users' freedom, it does ensure that the user of a program that is
linked with the Library has the freedom and the wherewithal to run
that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and
modification follow. Pay close attention to the difference between a
"work based
on the library" and a "work that uses the library". The
former contains code derived from the library, whereas the latter must
be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other

program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1

above, provided that you also meet all of these conditions:

a) The modified work must itself
be a software library.

b) You must cause the files modified to carry prominent notices
stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no
charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a
table of data to be supplied by an application program that uses
the facility, other than as an argument passed when the facility
is invoked, then you must make a good faith effort to ensure that,
in the event an application does not supply such function or
table, the facility still operates, and performs whatever part of
its purpose remains meaningful.

(For example, a function in a library to compute square roots has
a purpose that is entirely well-defined independent of the
application. Therefore, Subsection 2d requires that any
application-supplied
function or table used by this function must
be optional: if the application does not supply it, the square
root function must still compute square roots.)

These requirements apply to the modified work as a whole. If
identifiable sections of that work are not derived from the Library,
and can be reasonably considered independent and separate works in
themselves, then this License, and its terms, do not apply to those
sections when you distribute them as separate works. But when you
distribute the same sections as part of a whole which is a work based
on the Library, the distribution of the whole must be on the terms of
this License, whose permissions for other licensees extend to the
entire whole, and thus to each and every part regardless of who wrote
it.

Thus, it is not the intent of this section to claim rights or contest
your rights to work written entirely by you; rather, the intent is to
exercise the right to control the distribution
of derivative or
collective works based on the Library.

In addition, mere aggregation of another work not based on the Library
with the Library (or with a work based on the Library) on a volume of
a storage or distribution medium does not bring the other work under
the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and
derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work
with the Library by being compiled or
linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be

linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system,

rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining

where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any

patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO

WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED
TO LOSS OF DATA OR DATA BEING
RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library 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 Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your
employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.2 unzip 6.0

1.2.1 Available under license :

This is version 2009-Jan-02 of the Info-ZIP license.
The definitive version of this document should be available at
ftp://ftp.info-zip.org/pub/infozip/license.html indefinitely and
a copy at http://www.info-zip.org/pub/infozip/license.html.

Copyright (c) 1990-2009 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, "Info-ZIP" is defined as
the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois,
Jean-loup Gailly, Hunter Goatley, Ed Gordon, Ian Gorman, Chris Herborth,
Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz,
David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko,
Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs,
Kai Uwe Rommel, Steve Salisbury, Dave Smith, Steven M. Schweda,
Christian Spieler, Cosmin Truta, Antoine Verheijen, Paul von Behren,
Rich Wales, Mike White.

This software is provided "as is," without warranty of
any kind, express
or implied. In no event shall Info-ZIP or its contributors be held liable
for any direct, indirect, incidental, special or consequential damages

arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the above disclaimer and the following restrictions:

1. Redistributions of source code (in whole or in part) must retain
the above copyright notice, definition, disclaimer, and this list
of conditions.

2. Redistributions in binary form (compiled executables and libraries)
must reproduce the above copyright notice, definition, disclaimer,
and this list of conditions in documentation and/or other materials
provided with the distribution. Additional documentation is not needed
for executables where a command line license option provides these and
a
note regarding this option is in the executable's startup banner. The
sole exception to this condition is redistribution of a standard
UnZipSFX binary (including SFXWiz) as part of a self-extracting archive;
that is permitted without inclusion of this license, as long as the
normal SFX banner has not been removed from the binary or disabled.

3. Altered versions--including, but not limited to, ports to new operating
systems, existing ports with new graphical interfaces, versions with
modified or added functionality, and dynamic, shared, or static library
versions not from Info-ZIP--must be plainly marked as such and must not
be misrepresented as being the original source or, if binaries,
compiled from the original source. Such altered versions also must not
be misrepresented as being Info-ZIP releases--including, but not
limited to, labeling of the altered versions with the names "Info-ZIP"
(or any
variation thereof, including, but not limited to, different
capitalizations), "Pocket UnZip," "WiZ" or "MacZip" without the
explicit permission of Info-ZIP. Such altered versions are further
prohibited from misrepresentative use of the Zip-Bugs or Info-ZIP
e-mail addresses or the Info-ZIP URL(s), such as to imply Info-ZIP
will provide support for the altered versions.

4. Info-ZIP retains the right to use the names "Info-ZIP," "Zip," "UnZip,"
"UnZipSFX," "WiZ," "Pocket UnZip," "Pocket Zip," and "MacZip" for its
own source and binary releases.

This is the Info-ZIP file COPYING (for UnZip), last updated 17 Jul 2000.

FIRST NOTE:
This file contains some details about the copyright history of

contributions to the UnZip project.
Additionally, it summarises some exceptions to the general BSD-like
copyright found in LICENSE that covers our generic code and most of
the system specific ports.
Please read LICENSE first to find out what is allowed to do with
Info-ZIP's UnZip code.

- -

There are currently two explicit copyrights on portions of UnZip
code (at least, of which Info-ZIP is aware):
Jim Luther's Mac OS File Manager interface code; and Christopher Evans'
MacBinaryIII coding code (for the MacOS port).. These copyrights
are discussed in more detail below.

All remaining
code is now (starting with UnZip version 5.41) covered
by the new Info-ZIP license. For details, please read the acompaning
file LICENSE. The terms and conditions in this license supersede the
copyright conditions of the contributions by Igor Mandrichenko
(vms/vms.c), Greg Roelofs (zipinfo.c, new version of unshrink.c),
Mike White (Windows DLL code in "windll/*"), Steve P. Miller (Pocket
UnZip GUI "wince/*"), and Mark Adler (inflate/explode decompresseion
core routines, previously put into the public domain). All these
Info-ZIP contributors (or "primary" authors) have permitted us to
replace their copyright notes by the Info-ZIP License.

Frequently Asked Questions regarding (re)distribution of Zip and UnZip
are near the end of this file.

There are no known patents on any of the code in UnZip. Unisys
claims a patent on LZW encoding and on LZW decoding _in an apparatus
that performs LZW encoding_, but the patent appears to exempt a stand-
alone decoder (as in UnZip's unshrink.c). Unisys has publicly claimed
otherwise, but the issue has never been tested in court. Since this
point is unclear, unshrinking is not enabled by default. It is the
responsibility of the user to make his or her peace with Unisys and
its licensing requirements. (unshrink.c may be removed from future
releases altogether.)

The original unzip source code has been extensively modified and
almost entirely rewritten (changes include random zipfile access
rather than sequential; replacement of unimplode() with explode();
replacement of old unshrink() with new (unrelated) unshrink(); re-
placement of output routines; addition of inflate(), wildcards,
filename-mapping, text translation, ...; etc.). As far as we can
tell, only the core code of the unreduce method remained substantially

similar to Mr. Smith's original source. As of UnZip
5.42, the complete
core code is now covered by the Info-ZIP Licence. Therefore, support
for the reduce method has been removed.
The drop of the reduce method should only affect some test archives,
reducing was never used in any publically distributed Zip program.
For pathologic cases where support for reduced archive entries is
needed, the unreduce code copyrighted by Samuel H. Smith is available
as a separate distribution (the restricted copyright of this code is
cited below in the "historical" section).

The following copyright applies to the Mac OS File Manager interface code
(macos/source/macstuff.[ch]), distributed with UnZip 5.4 and later:

```
  * MoreFiles
  *
  * A collection of File Manager and related routines
  *
  * by Jim Luther (Apple Macintosh Developer Technical Support Emeritus)
  * with significant code contributions by Nitin Ganatra
  * (Apple Macintosh Developer Technical Support Emeritus)
  * Copyright  1992-1998
Apple Computer, Inc.
  * Portions copyright  1995 Jim Luther
  * All rights reserved.
  * The Package "More Files" is distributed under the following
  * license terms:
  *
  *         "You may incorporate this sample code into your
  *          applications without restriction, though the
  *          sample code has been provided "AS IS" and the
  *          responsibility for its operation is 100% yours.
  *          However, what you are not permitted to do is to
  *          redistribute the source as "DSC Sample Code" after
  *          having made changes. If you're going to
  *          redistribute the source, we require that you make
  *          it clear in the source that the code was descended
  *          from Apple Sample Code, but that you've made
  *          changes."
```

The usage terms of this copyright note are compatible with the
Info-ZIP license, they do not add further restrictions.

The following
copyright applies to the Mac OS "macbin3" decoding code
(extra field compatibility with ZipIt):

```
  *  MacBinaryIII.h
  *
  *  Copyright 1997 Christopher Evans (cevans@poppybank.com)
  *
  *  Basic encoding and decoding of Macintosh files to the
  *  MacBinary III spec.
  * ---------------------------------------------------------------------
  * This source is copyrighted by Christopher Evans (cevans@poppybank.com)
  * (available at ftp://ftp.lazerware.com/MacBinaryIII_src_C.sit
  * homepage of Leonard Rosenthol  leonardr@netcom.com)
```

This copyright note does not contain any usage terms. So, we assume that this code is freely reusable until we are proved wrong...

The remaining copyright notes have been superseeded by the new Info-ZIP license, with explicit permission from the respective original authors. They are cited here for historical reasons, only:

The following copyright applies to the full-featured unreduce.c (now distributed separately):

```
  * Copyright 1989 Samuel H. Smith;  All rights reserved
  *
  * Do not distribute modified versions without my permission.
  * Do not remove or alter this notice or any other copyright notice.
  * If you use this in your own program you must distribute source code.
  * Do not use any of this in a commercial product.
```

Regarding the first stipulation, Mr. Smith was tracked down in southern California some years back [Samuel H. Smith, The Tool Shop; as of mid-May 1994, (213) 851-9969 (voice), (213) 887-2127(?) (subscription BBS), 71150.2731@compuserve.com]:

"He says that he thought that whoever contacted him understood that he has no objection to the Info-ZIP group's inclusion of his code. His primary concern is that it remain freely distributable, he said."

Despite the fact that our "normal" code has been entirely rewritten and by default no longer contains any of Mr. Smith's code, Info-ZIP remains indebted and grateful to him. We hope he finds our contributions as useful as we have his.

Note that the third and fourth stipulations still apply to any company that wishes to incorporate the unreduce code into its products;

if you wish to do so, you must contact Mr. Smith directly regarding licensing.

The following copyright applied to most of the VMS code in vms.c, distributed with UnZip version 4.2 and later:

```
  * Copyright (c) 1992-93 Igor Mandrichenko.
  * Permission is granted to any individual or institution to use, copy,
  * or redistribute this software so long as all of the original files
  * are included unmodified and that this copyright notice is retained.
```

The following copyright applied to the new version of unshrink.c, distributed with UnZip version 5.2 and later:

```
  * Copyright (c) 1994 Greg Roelofs.
  * Permission is granted to any individual/institution/corporate
   * entity to use, copy, redistribute or modify this software for
  * any purpose whatsoever, subject to the conditions noted in the
  * Frequently Asked Questions section below, plus one additional
  * condition:  namely, that my name not be removed from the source
  * code.  (Other names may, of course, be added as modifications
  * are made.)  Corporate legal staff (like at IBM :-) ) who have
  * problems understanding this can contact me through Zip-Bugs...
```

The following copyright applied to the Windows DLL code (windll/*), distributed with UnZip version 5.2 and later:

```
  * Copyright (c) 1996 Mike White.
  * Permission is granted to any individual or institution to use,
  * copy, or redistribute this software so long as all of the original
  * files are included, that it is not sold for profit, and that this
  * copyright notice is retained.
```

The following copyright applied to the Windows CE GUI port, ``Pocket UnZip," distributed with UnZip version 5.3 and later:

```
  * All the source files for Pocket UnZip, except for components
  * written by the Info-ZIP group, are copyrighted 1997 by Steve P.
  * Miller.  The product "Pocket UnZip" itself is property of the
```

* author and cannot be altered in any way without written consent
* from Steve P. Miller.

The remaining code was written by many people associated with the
Info-ZIP group, with large contributions from (but not limited to):
Greg Roelofs (overall program logic, ZipInfo, unshrink, filename
mapping/portability, etc.), Mark Adler (inflate, explode, funzip),
Kai Uwe Rommel (OS/2), John Bush and Paul Kienitz (Amiga), Antoine
Verheijen (Macintosh), Hunter Goatley (more VMS), Mike White (Windows
DLLs), Christian Spieler (overall logic, optimization, VMS, etc.) and
others. See the file CONTRIBS in the source distribution for a much
more complete list of contributors.
The decompression core code for the deflate method (inflate.[ch],
explode.c) was originally written by Mark Adler who submitted it
as public domain code.

1.3 expat 2.0.1

1.3.1 Available under license :

Copyright (c) 1998-2000 Thai Open Source Software Center Ltd and Clark Cooper
Copyright (c) 2001-2017 Expat maintainers

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
"Software"), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN
ACTION OF CONTRACT,
TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.4 linux-kernel 2.6.19

1.4.1 Available under license :

Copyright (c) 2003-2006 QLogic Corporation
QLogic Linux Networking HBA Driver

This program includes a device driver for Linux 2.6 that may be
distributed with QLogic hardware specific firmware binary file.
You may modify and redistribute the device driver code under the
GNU General Public License as published by the Free Software
Foundation (version 2 or a later version).

You may redistribute the hardware specific firmware binary file
under the following terms:

1. Redistribution of source code (only if applicable),
   must retain the above copyright notice, this list of
   conditions and the following disclaimer.

2. Redistribution 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 QLogic Corporation may not be used to
   endorse or promote products derived from this software
   without specific prior written permission

REGARDLESS
 OF WHAT LICENSING MECHANISM IS USED OR APPLICABLE,
THIS PROGRAM IS PROVIDED BY QLOGIC CORPORATION "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.

USER ACKNOWLEDGES AND AGREES THAT USE OF THIS PROGRAM WILL NOT
CREATE OR GIVE GROUNDS FOR A LICENSE BY IMPLICATION, ESTOPPEL, OR
OTHERWISE IN ANY INTELLECTUAL PROPERTY RIGHTS (PATENT, COPYRIGHT,
TRADE SECRET, MASK WORK,

OR OTHER PROPRIETARY RIGHT) EMBODIED IN
ANY OTHER QLOGIC HARDWARE OR SOFTWARE EITHER SOLELY OR IN
COMBINATION WITH THIS PROGRAM.
Contributors to bttv:

Michael Chu <mmchu@pobox.com>
AverMedia fix and more flexible card recognition

Alan Cox <alan@redhat.com>
Video4Linux interface and 2.1.x kernel adaptation

Chris Kleitsch
Hardware I2C

Gerd Knorr <kraxel@cs.tu-berlin.de>
Radio card (ITT sound processor)

bigfoot <bigfoot@net-way.net>
Ragnar Hojland Espinosa <ragnar@macula.net>
ConferenceTV card

+ many more (please mail me if you are missing in this list and would
like to be mentioned)
Copyright (c) 2003-2005 QLogic Corporation
QLogic Linux Fibre Channel HBA Driver

This program includes a device driver for Linux 2.6 that may be
distributed with QLogic hardware specific firmware binary file.
You may modify and redistribute the device driver code under the
GNU General Public License as published by the Free Software
Foundation (version 2 or a later version).

You may redistribute the hardware specific firmware binary file
under the following terms:

1. Redistribution of source code (only if applicable),
must retain the above copyright notice, this list of
conditions and the following disclaimer.

2. Redistribution 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 QLogic Corporation may not be used to
endorse or promote products derived from this software
without specific prior written permission

REGARDLESS
OF WHAT LICENSING MECHANISM IS USED OR APPLICABLE,
THIS PROGRAM IS PROVIDED BY QLOGIC CORPORATION "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.

USER ACKNOWLEDGES AND AGREES THAT USE OF THIS PROGRAM WILL NOT
CREATE OR GIVE GROUNDS FOR A LICENSE BY IMPLICATION, ESTOPPEL, OR
OTHERWISE IN ANY INTELLECTUAL PROPERTY RIGHTS (PATENT, COPYRIGHT,
TRADE SECRET, MASK WORK,
OR OTHER PROPRIETARY RIGHT) EMBODIED IN
ANY OTHER QLOGIC HARDWARE OR SOFTWARE EITHER SOLELY OR IN
COMBINATION WITH THIS PROGRAM.
The files in this directory and elsewhere which refer to this LICENCE
file are part of JFFS2, the Journalling Flash File System v2.

Copyright (C) 2001, 2002 Red Hat, Inc.

JFFS2 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 or (at your option) any later
version.

JFFS2 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 JFFS2; if not, write to the Free Software Foundation, Inc.,
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

As a special exception, if other files instantiate templates or use
macros or inline functions from these files, or you compile these
files and link them with other works to produce a work
based on these
files, these files do not by themselves cause the resulting work to be
covered by the GNU General Public License. However the source code for

these files must still be made available in accordance with section (3)
of the GNU General Public License.

This exception does not invalidate any other reasons why a work based on
this file might be covered by the GNU General Public License.

For information on obtaining alternative licences for JFFS2, see
http://sources.redhat.com/jffs2/jffs2-licence.html

$Id: LICENCE,v 1.1 2002/05/20 14:56:37 dwmw2 Exp $

These microcode data are placed under the terms of the GNU General Public License.

We would prefer you not to distribute modified versions of it and not to ask
for assembly or other microcode source.

/* nicstar.c v0.22 Jawaid Bazyar (bazyar@hypermall.com)
* nicstar.c, M. Welsh (matt.welsh@cl.cam.ac.uk)
*
* Hacked October, 1997 by Jawaid Bazyar, Interlink Advertising Services Inc.
* http://www.hypermall.com/
* 10/1/97 - commented out CFG_PHYIE bit - we don't care when the PHY

```
* interrupts us (except possibly for removal/insertion of the cable?)
* 10/4/97 - began heavy inline documentation of the code. Corrected typos
* and spelling mistakes.
* 10/5/97 - added code to handle PHY interrupts, disable PHY on
* loss of link, and correctly re-enable PHY when link is
* re-established. (put back CFG_PHYIE)
*
*   Modified to work with the IDT7721 nicstar -- AAL5 (tested) only.
*
* R. D. Rechenmacher <ron@fnal.gov>, Aug. 6, 1997 $Revision: 1.1 $ $Date: 1999/08/20 11:00:11 $
*
* Linux driver for the IDT77201 NICStAR PCI ATM controller.
* PHY component is expected to be 155 Mbps S/UNI-Lite or IDT 77155;
* see init_nicstar() for PHY initialization to change this. This driver
 * expects the Linux ATM stack to support scatter-gather lists
* (skb->atm.iovcnt != 0) for Rx skb's passed to vcc->push.
*
* Implementing minimal-copy of received data:
*   IDT always receives data into a small buffer, then large buffers
*    as needed. This means that data must always be copied to create
*    the linear buffer needed by most non-ATM protocol stacks (e.g. IP)
*    Fix is simple: make large buffers large enough to hold entire
*    SDU, and leave <small_buffer_data> bytes empty at the start. Then
*    copy small buffer contents to head of large buffer.
*   Trick is to avoid fragmenting Linux, due to need for a lot of large
*    buffers. This is done by 2 things:
*      1) skb->destructor / skb->atm.recycle_buffer
*          combined, allow nicstar_free_rx_skb to be called to
*          recycle large data buffers
*      2) skb_clone of received buffers
*   See nicstar_free_rx_skb and linearize_buffer for implementation
*    details.
*
*
 *
* Copyright (c) 1996 University of Cambridge Computer Laboratory
*
*   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., 675 Mass Ave, Cambridge, MA 02139, USA.
*
* M. Welsh, 6 July 1996
*
*
*/
Thanks go to the following people for patches and contributions:

Michael Hunold <m.hunold@gmx.de>
 for the initial saa7146 driver and it's recent overhaul

Christian Theiss
 for his work on the initial Linux DVB driver

Marcus Metzler <mocm@metzlerbros.de>
Ralph Metzler <rjkm@metzlerbros.de>
 for their continuing work on the DVB driver

Michael Holzt <kju@debian.org>
 for his contributions to the dvb-net driver

Diego Picciani <d.picciani@novacomp.it>
 for CyberLogin for Linux which allows logging onto EON
 (in case you are wondering where CyberLogin is, EON changed its login
 procedure and CyberLogin is no longer used.)

Martin Schaller <martin@smurf.franken.de>
 for patching the cable card decoder driver

Klaus Schmidinger <Klaus.Schmidinger@cadsoft.de>
 for various fixes regarding tuning, OSD and CI stuff and his work on VDR

Steve Brown <sbrown@cortland.com>
 for his AFC kernel thread

Christoph Martin <martin@uni-mainz.de>
 for his LIRC infrared handler

Andreas Oberritter
 <obi@linuxtv.org>
Dennis Noermann <dennis.noermann@noernet.de>
Felix Domke <tmbinc@elitedvb.net>
Florian Schirmer <jolt@tuxbox.org>
Ronny Strutz <3des@elitedvb.de>
Wolfram Joost <dbox2@frokaschwei.de>
...and all the other dbox2 people
 for many bugfixes in the generic DVB Core, frontend drivers and

their work on the dbox2 port of the DVB driver

Oliver Endriss <o.endriss@gmx.de>
for many bugfixes

Andrew de Quincey <adq_dvb@lidskialf.net>
for the tda1004x frontend driver, and various bugfixes

Peter Schildmann <peter.schildmann@web.de>
for the driver for the Technisat SkyStar2 PCI DVB card

Vadim Catana <skystar@moldova.cc>
Roberto Ragusa <r.ragusa@libero.it>
Augusto Cardoso <augusto@carhil.net>
for all the work for the FlexCopII chipset by B2C2,Inc.

Davor Emard <emard@softhome.net>
for his work on the budget drivers, the demux code,
the module unloading problems, ...

Hans-Frieder Vogt <hfvogt@arcor.de>
for his work on calculating and checking the crc's for the
TechnoTrend/Hauppauge DEC driver firmware

Michael Dreher <michael@5dot1.de>
Andreas 'randy' Weinberger
for the support of the Fujitsu-Siemens Activy budget DVB-S

Kenneth Aafly <ke-aa@frisurf.no>
for adding support for Typhoon DVB-S budget card

Ernst Peinlich <e.peinlich@inode.at>
for tuning/DiSEqC support for the DEC 3000-s

Peter Beutner <p.beutner@gmx.net>
for the IR code for the ttusb-dec driver

Wilson Michaels <wilsonmichaels@earthlink.net>
for the lgdt330x frontend driver, and various bugfixes

Michael Krufky <mkrufky@m1k.net>
for maintaining v4l/dvb inter-tree dependencies

Taylor Jacob <rtjacob@earthlink.net>
for the nxt2002 frontend driver

Jean-Francois Thibert <jeanfrancois@sagetv.com>
for the nxt2004 frontend driver

Kirk Lapray <kirk.lapray@gmail.com>
 for the or51211 and or51132 frontend drivers, and
 for merging the nxt2002 and nxt2004 modules into a
 single nxt200x frontend driver.

(If you think you should be in this list, but you are not, drop a
 line to the DVB mailing list)
Code in this directory written at the IDA Supercomputing Research Center
carries the following copyright and license.

  Copyright 1993 United States Government as represented by the
  Director, National Security Agency. This software may be used
  and distributed according to the terms of the GNU General Public License,
  incorporated herein by reference.

  In addition to the disclaimers in the GPL, SRC expressly disclaims any
  and all warranties, expressed or implied, concerning the enclosed software.
  This software was developed at SRC for use in internal research, and the
  intent in sharing this software is to promote the productive interchange
  of ideas throughout the research community.   All software is furnished
  on an "as-is" basis. No further updates to this software should be
  expected. Although updates may occur, no commitment exists.

 NOTE! This copyright does *not* cover user programs that use kernel
services by normal system calls - this is merely considered normal use
of the kernel, and does *not* fall under the heading of "derived work".
Also note that the GPL below is copyrighted by the Free Software
Foundation, but the instance of code that it refers to (the Linux
kernel) is copyrighted by me and others who actually wrote it.

Also note that the only valid version of the GPL as far as the kernel
is concerned is _this_ particular version of the license (ie v2, not
v2.2 or v3.x or whatever), unless explicitly otherwise stated.

  Linus Torvalds

    GNU GENERAL PUBLIC LICENSE
      Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
                51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so
that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is
to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any
other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium

customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface
definition files, plus the scripts used to
control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License.
However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further

restrictions on the recipients' exercise
of the rights granted herein.
You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular
circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted
interfaces, the
original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions

of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this
is to make it
free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest
to attach them to the start of each source file to most effectively
convey the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

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., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this
when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer)
or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General
Public License instead of this License.

FlashPoint Driver Developer's Kit
Version 1.0

This program is free software; you may redistribute and/or modify it under
the terms of either:

a) the GNU General Public License as published by the Free Software
Foundation; either version 2, or (at your option) any later version,

or

b) the "BSD-style License" included below.

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 either the GNU General Public
License or the BSD-style License below 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.,
675 Mass Ave, Cambridge, MA 02139, USA.

The BSD-style License is as follows:

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 this LICENSE.FlashPoint
file, without modification, this list of conditions, and the following
disclaimer. The following copyright notice must appear immediately at
the beginning of all source files:

Copyright 1995-1996 by Mylex Corporation. All Rights Reserved

This file is available under both the GNU General Public License
and a BSD-style copyright; see LICENSE.FlashPoint for details.

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 Mylex Corporation may not be used to endorse or promote
products derived from this software without specific prior written
permission.

THIS SOFTWARE
IS PROVIDED BY MYLEX CORP. ``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 OR CONTRIBUTORS 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.

GNU LIBRARY GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1991 Free Software Foundation, Inc.
675 Mass Ave, Cambridge, MA 02139, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Library General Public License, applies to some
specially designated Free Software Foundation software, and to any
other libraries whose authors decide to use it. You can use it for
your libraries, too.

When we speak of free software, we are referring to freedom,
not
price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes
to the library and recompiling
it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed
for everyone's
free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

GNU LIBRARY GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed
under these terms. A "work based on the
Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of
the use of the Library in a tool for
writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must
itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no

charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the

Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

c) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

d) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for

copying, distributing or modifying
the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute
the Library at all. For example, if a patent
license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system;
it is up to the author/donor to decide if he or she is willing
to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries,

so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each
version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS
LICENSED FREE OF CHARGE, THERE IS NO
WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES

SUSTAINED BY YOU OR THIRD PARTIES OR A
FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF
SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH
DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest
possible use to the public, we recommend making it free software that
everyone can redistribute and change. You can do so by permitting
redistribution under these terms (or, alternatively, under the terms of the
ordinary General Public License).

To apply these terms, attach the following notices to the library. It is
safest to attach them to the start of each source file to most effectively
convey the exclusion of warranty; and each file should have at least the
"copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year>
<name of author>

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Library General Public
License as published by the Free Software Foundation; either
version 2 of the License, or (at your option) any later version.

This library 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
Library General Public License for more details.

You should have received a copy of the GNU Library General Public
License along with this library; if not, write to the Free
Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter
the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.5 pjnath 1.8.1

1.5.1 Available under license :

No license file was found, but licenses were detected in source scan.

/*
* Copyright (C) 2008-2009 Teluu Inc. (http://www.teluu.com)
*
* 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
*/

Found in path(s):
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/docs/doc_samples.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/docs/doc_turn.h
*
/opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/docs/doc_stun.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/docs/doc_ice.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/docs/doc_mainpage.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/docs/doc_nat.h
No license file was found, but licenses were detected in source scan.

/*
* Copyright (C) 2008-2009 Teluu Inc. (http://www.teluu.com)
* Copyright (C) 2003-2008 Benny Prijono <benny@prijono.org>
*
* 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
*/

Found in path(s):
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/turn_sock.c
*
 /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/allocation.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/stun_sock_test.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/ice_session.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/stun_msg_dump.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/auth.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/ice_test.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/errno.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/nat_detect.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/turn_session.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/main.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/turn_sock.h
*
 /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/server.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/nat_detect.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/server.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/main.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/stun_config.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/test.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/stun_transaction.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/test.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/stun_sock.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/ice_session.h
*
 /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/turn.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/listener_tcp.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/types.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/stun_msg.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/sess_auth.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/ice_strans.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/stun_transaction.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/stun_session.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/stun_sock.c
*

/opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-client/client_main.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/stun_auth.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/server.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/ice_strans.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/config.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/stun_msg.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/auth.h
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjturn-srv/listener_udp.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/errno.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/turn_session.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/include/pjnath/stun_session.h
*
/opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/stun.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath-test/turn_sock_test.c
* /opt/cola/permits/1606729853_1681993835.6041217/0/pjnath-zip/pjnath/src/pjnath/stun_auth.c

1.6 alsa 1.0.13

1.6.1 Available under license :

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who decide to use it. You
can use it too, but we suggest
you first think carefully about whether
this license or the ordinary General Public License is the better
strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use,

not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using

a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits
its criteria of freedom. The Lesser General
Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use
the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or

other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public

License instead of this License to
a given copy of the Library. To do
this, you must alter all the notices that refer to this License, so
that they refer to the ordinary GNU General Public License, version 2,
instead of to this License. (If a newer version than version 2 of the
ordinary GNU General Public License has appeared, then you can specify
that version instead if you wish.) Do not make any other change in
these notices.

Once this change is made in a given copy, it is irreversible for
that copy, so the ordinary GNU General Public License applies to all
subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of
the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or
derivative of it, under Section 2) in object code or executable form
under the terms of Sections 1 and 2 above provided that you accompany
it with the complete corresponding machine-readable source code, which
must be distributed
under the terms of Sections 1 and 2 above on a
medium customarily used for software interchange.

If distribution of object code is made by offering access to copy
from a designated place, then offering equivalent access to copy the
source code from the same place satisfies the requirement to
distribute the source code, even though third parties are not
compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the
Library, but is designed to work with the Library by being compiled or
linked with it, is called a "work that uses the Library". Such a
work, in isolation, is not a derivative work of the Library, and
therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library
creates an executable that is a derivative of the Library (because it
contains portions of the Library), rather than a "work that uses the
library". The executable is therefore covered by this
License.
Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file
that is part of the Library, the object code for the work may be a
derivative work of the Library even though the source code is not.
Whether this is true is especially significant if the work can be
linked without the Library, or if the work is itself a library. The

threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables
containing that work also fall under Section 6,
whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable
source code for the Library including whatever
changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if

the user
installs one, as long as the modified version is
interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at
least three years, to give the same user the materials
specified in Subsection 6a, above, for a charge no more
than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy
from a designated place, offer equivalent access to copy the above
specified materials from the same place.

e) Verify that the user has already received a copy of these
materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the
Library" must include any data and utility programs needed for
reproducing the executable from it. However, as a special exception,
the materials to be distributed need not include anything that is
normally distributed (in either source or binary form) with the
major
components (compiler, kernel, and so on) of the operating system on
which the executable runs, unless that component itself accompanies
the executable.

It may happen that this requirement contradicts the license
restrictions of other proprietary libraries that do not normally
accompany the operating system. Such a contradiction means you cannot
use both them and the Library together in an executable that you
distribute.

7. You may place library facilities that are a work based on the
Library side-by-side in a single library together with other library
facilities not covered by this License, and distribute such a combined
library, provided that the separate distribution of the work based on
the Library and of the other library facilities is otherwise
permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work
based on the Library, uncombined with any other library
facilities. This must be distributed under
the terms of the
Sections above.

b) Give prominent notice with the combined library of the fact
that part of it is a work based on the Library, and explaining
where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing
the Library (or any work based on the
Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute
so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the

integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW.

EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL
DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and
a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library 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 Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

You should also
get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users. This
General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it. (Some other Free Software Foundation software is covered by
the GNU Lesser General Public License instead.) You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses
are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid

anyone to deny you these rights or to ask you to surrender the rights.
These restrictions translate to certain responsibilities for you if you
distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must give the recipients all the rights that
you have. You must make sure that they, too, receive or can get the
source code. And you must show them these terms so they know their
rights.

We protect your rights with two steps: (1) copyright the software, and
(2) offer you this license which gives you legal permission
to copy,
distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain
that everyone understands that there is no warranty for this free
software. If the software is modified by someone else and passed on, we
want its recipients to know that what they have is not the original, so
that any problems introduced by others will not reflect on the original
authors' reputations.

Finally, any free program is threatened constantly by software
patents. We wish to avoid the danger that redistributors of a free
program will individually obtain patent licenses, in effect making the
program proprietary. To prevent this, we have made it clear that any
patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and
modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program
or other work which contains
a notice placed by the copyright holder saying it may be distributed
under the terms of this General Public License. The "Program", below,
refers to any such program or work, and a "work based on the Program"
means either the Program or any derivative work under copyright law:
that is to say, a work containing the Program or a portion of it,
either verbatim or with modifications and/or translated into another
language. (Hereinafter, translation is included without limitation in
the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not
covered by this License; they are outside its scope. The act of
running the Program is not restricted, and the output from the Program

is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based

on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled
to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based
on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit
royalty-free redistribution of the Program by
all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under

any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute
software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies
a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
APPLICABLE LAW. EXCEPT WHEN
OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

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
2.1 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

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this
when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands
`show w' and `show c' should show the appropriate
parts of the General Public License. Of course, the commands you use may
be called something other than `show w' and `show c'; they could even be
mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General
Public License instead of this License.

1.7 ortp 0.13.1

1.7.1 Available under license :

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest
you first think carefully about whether
this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients
all the rights that we gave
you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal

permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use

the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without
limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square

root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves,
then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to
a given copy of the Library. To do
this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed
under the terms of Sections 1 and 2 above on a
medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License.
Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the

Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on

which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further

restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute
so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people
have made
generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License

from time to time.
Such new versions will be similar in spirit to the present version,
but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library
specifies a version number of this License which applies to it and
"any later version", you have the option of following the terms and
conditions either of that version or of any later version published by
the Free Software Foundation. If the Library does not specify a
license version number, you may choose any version ever published by
the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free
programs whose distribution conditions are incompatible with these,
write to the author to ask for permission. For software which is
copyrighted by the Free Software Foundation, write to the Free
Software Foundation; we sometimes make exceptions for this. Our
decision will be guided by the two goals of preserving the free status
of
all derivatives of our free software and of promoting the sharing
and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO
WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW.
EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR
OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY
KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE
LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME
THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN
WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY
AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU
FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR
CONSEQUENTIAL
DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE
LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING
RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A
FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF
SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH
DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and
a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.

This library 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
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

You should also
get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the library, if
necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the
library `Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

1.8 busybox 1.2.2

1.8.1 Available under license :

The code and graphics on this website (and it's mirror sites, if any) are
Copyright (c) 1999-2004 by Erik Andersen. All rights reserved.
Copyright (c) 2005-2006 Rob Landley.

Documents on this Web site including their graphical elements, design, and
layout are protected by trade dress and other laws and MAY BE COPIED OR
IMITATED IN WHOLE OR IN PART. THIS WEBSITE IS LICENSED FREE OF CHARGE, THERE
IS NO WARRANTY FOR THE WEBSITE TO THE EXTENT PERMITTED BY APPLICABLE LAW.
SHOULD THIS WEBSITE PROVE DEFECTIVE, YOU MAY ASSUME THAT SOMEONE MIGHT GET
AROUND TO SERVICING, REPAIRING OR CORRECTING IT SOMETIME WHEN THEY HAVE NOTHING
BETTER TO DO. REGARDLESS, YOU GET TO KEEP BOTH PIECES.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY
COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THIS
WEBSITE AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR
INABILITY
TO USE THIS WEBSITE (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR
LOSS OF HAIR, LOSS OF LIFE, LOSS OF MEMORY, LOSS OF YOUR CARKEYS, MISPLACEMENT
OF YOUR PAYCHECK, OR COMMANDER DATA BEING RENDERED UNABLE TO ASSIST THE
STARFLEET OFFICERS ABORD THE STARSHIP ENTERPRISE TO RECALIBRATE THE MAIN
DEFLECTOR ARRAY, LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE
WEBSITE TO OPERATE WITH YOUR WEBBROWSER), EVEN IF SUCH HOLDER OR OTHER PARTY
HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You have been warned.

You can contact the webmaster at <rob@landley.net> if you have some sort
of problem with this.
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users. This

General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it. (Some other Free Software Foundation software is covered by
the GNU Library General Public License instead.) You can apply it to
your programs, too.

When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed
to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid
anyone to deny you these rights or to ask you to surrender the rights.
These restrictions translate to certain responsibilities for you if you
distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether
gratis or for a fee, you must give the recipients all the rights that
you have. You must make sure that they, too, receive or can get the
source code. And you must show them these terms so they know their
rights.

We protect your rights with two steps: (1) copyright the software, and
(2) offer you this license which gives you legal permission to copy,
distribute
and/or modify the software.

Also, for each author's protection and ours, we want to make certain
that everyone understands that there is no warranty for this free
software. If the software is modified by someone else and passed on, we
want its recipients to know that what they have is not the original, so
that any problems introduced by others will not reflect on the original
authors' reputations.

Finally, any free program is threatened constantly by software
patents. We wish to avoid the danger that redistributors of a free
program will individually obtain patent licenses, in effect making the
program proprietary. To prevent this, we have made it clear that any
patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and
modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains
a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's
source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole
or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an

announcement including an appropriate copyright notice and a
notice that there is no warranty (or else, saying that you provide
a warranty) and that users may redistribute the program under
these conditions, and telling the user how to view a copy of this
License. (Exception: if the Program itself is interactive but
does not normally print such an announcement, your work based on
the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If
identifiable sections of that work are not derived from the Program,
and can be reasonably considered
independent and separate works in
themselves, then this License, and its terms, do not apply to those
sections when you distribute them as separate works. But when you
distribute the same sections as part of a whole which is a work based
on the Program, the distribution of the whole must be on the terms of
this License, whose permissions for other licensees extend to the
entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest
your rights to work written entirely by you; rather, the intent is to
exercise the right to control the distribution of derivative or
collective works based on the Program.

In addition, mere aggregation of another work not based on the Program
with the Program (or with a work based on the Program) on a volume of
a storage or distribution medium does not bring the other work under
the scope of this License.

3. You may copy and distribute the Program (or a work based
on it,
under Section 2) in object code or executable form under the terms of
Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable
source code, which must be distributed under the terms of Sections
1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three
years, to give any third party, for a charge no more than your
cost of physically performing source distribution, a complete
machine-readable copy of the corresponding source code, to be
distributed under the terms of Sections 1 and 2 above on a medium
customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer
to distribute corresponding source code. (This alternative is
allowed only for noncommercial distribution and only if you

received the program in object code or
executable form with such
an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the
source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each
time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program

specifies a version number
of this License which applies to it and "any
later version", you have the option of following the terms and conditions
either of that version or of any later version published by the Free
Software Foundation. If the Program does not specify a version number of
this License, you may choose any version ever published by the Free Software
Foundation.

10. If you wish to incorporate parts of the Program into other free
programs whose distribution conditions are different, write to the author
to ask for permission. For software which is copyrighted by the Free
Software Foundation, write to the Free Software Foundation; we sometimes
make exceptions for this. Our decision will be guided by the two goals
of preserving the free status of all derivatives of our free software and
of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY
FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW.
EXCEPT WHEN
OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES
PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED
OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS
TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE
PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING,
REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR
REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES,
INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING
OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED
TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY
YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER
PROGRAMS), EVEN
IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE
POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

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

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into
proprietary programs. If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library. If this is what you want to do, use the GNU Library General
Public License instead of this License.
<!--#include file="header.html" -->

<h3>BusyBox is licensed under the GNU General Public License</h3>

BusyBox is licensed under the
GNU General Public License version 2 or later, which is generally
abbreviated as the GPL. (This is the same license the Linux kernel is under,
so you may be somewhat familiar with it by now.)

Anyone thinking of shipping BusyBox as part of a
product should be familiar with the licensing terms under which they are
allowed to use and distribute BusyBox. Read the full test of the GPL (either
through the above link, or in the file LICENSE in the busybox tarball), and
also read the Frequently
Asked Questions about the GPL.

Basically, if you distribute GPL software the license requires that you also
distribute the source code to that GPL-licensed software. So if you distribute
BusyBox without making
 the source code to the version you distribute available,
you violate the license terms, and thus infringe on the copyrights of BusyBox.
(This requirement applies whether or not you modified BusyBox; either way the
license terms still apply to you.) Read the license text for the details.

BusyBox's copyrights are enforced by the <a
href="http://www.softwarefreedom.org">Software Freedom Law Center, which
"accepts primary responsibility for enforcement of US copyrights on the
software... and coordinates international copyright enforcement efforts for
such works as necessary." If you distribute BusyBox in a way that doesn't
comply with the terms of the license BusyBox is distributed under, expect to
hear from these guys. Their entire reason for existing is to do pro-bono
legal work for free/open source software projects. (We used to list people who
violate the BusyBox license in The Hall of Shame,
but these days we find it much more effective to hand
 them over to the
lawyers.)

Our enforcement efforts are aimed at bringing people into compliance with
the BusyBox license. Open source software is under a different license from
proprietary software, but if you violate that license you're still a software

pirate and the law gives the vendor (us) some big sticks to play with. We
don't want monetary awards, injunctions, or to generate bad PR for a company,
unless that's the only way to get somebody that repeatedly ignores us to comply
with the license on our code.

<h3>A Good Example</h3>

These days, Linksys is
doing a good job at complying with the GPL, they get to be an
example of how to do things right. Please take a moment and
check out what they do with
<a
href="http://www.linksys.com/servlet/Satellite?c=L_Content_C1&childpagename=US%2FLayout&cid=1115416836002&pagename=Linksys%2FCommon%2FVisitorWrapper">
distributing the firmware for their WRT54G Router.
Following their
example would be a fine way to ensure that you
have also fulfilled your licensing obligations.

<!--#include file="footer.html" -->

1.9 pjsip 1.8.10

1.9.1 Available under license :

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
Licenses are intended to guarantee your freedom to share and change
free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some
specially designated software packages--typically libraries--of the
Free Software Foundation and other authors who

decide to use it. You
can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library,
whether gratis
or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software
patents pose a constant threat to the existence of
any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the

ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits
such linking only if the
entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting
the free library to free
software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must

be combined with the library in order
to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into
another language. (Hereinafter, translation is
included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control
compilation
and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium,
provided that
you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the
terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be
reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to

exercise the right to control the distribution of derivative or
collective works based on the Library.

In addition, mere aggregation of another work not based on the Library
with the Library (or with a work based on the Library) on a volume of
a storage or distribution medium does not bring the other work under
the scope of this License.

3. You may opt to apply the terms of the ordinary
GNU General Public
License instead of this License to a given copy of the Library. To do
this, you must alter all the notices that refer to this License, so
that they refer to the ordinary GNU General Public License, version 2,
instead of to this License. (If a newer version than version 2 of the
ordinary GNU General Public License has appeared, then you can specify
that version instead if you wish.) Do not make any other change in
these notices.

Once this change is made in a given copy, it is irreversible for
that copy, so the ordinary GNU General Public License applies to all
subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of
the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or
derivative of it, under Section 2) in object code or executable form
under the terms of Sections 1 and 2 above provided that you accompany
it with the complete corresponding
machine-readable source code, which
must be distributed under the terms of Sections 1 and 2 above on a
medium customarily used for software interchange.

If distribution of object code is made by offering access to copy
from a designated place, then offering equivalent access to copy the
source code from the same place satisfies the requirement to
distribute the source code, even though third parties are not
compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the
Library, but is designed to work with the Library by being compiled or
linked with it, is called a "work that uses the Library". Such a
work, in isolation, is not a derivative work of the Library, and
therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library
creates an executable that is a derivative of the Library (because it
contains portions of the Library), rather than a "work that

uses the
library". The executable is therefore covered by this License.
Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the
object code for the work under the terms of Section 6.
Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany
the work with the complete corresponding
machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood

that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate
properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally
distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to

refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system
which is
implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish
revised and/or new
versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free

Software Foundation; we sometimes make exceptions for this. Our decision
will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE,
BE LIABLE TO YOU
FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

```
/*
*
* Copyright (c) 2001-2006 Cisco Systems, Inc.
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
*   Redistributions of source code must retain the above copyright
*   notice, this list of conditions and the following disclaimer.
*
*   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.
*
```

* Neither the name of the Cisco Systems, Inc. nor the names of its
* contributors may be used to endorse or promote products derived
* from this software without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
* "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
* COPYRIGHT HOLDERS OR CONTRIBUTORS 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.
*
*/
Copyright 1992, 1993, 1994 by Jutta Degener and Carsten Bormann,
Technische Universitaet Berlin

Any use of this software is permitted provided that this notice is not
removed and that neither the authors nor the Technische Universitaet Berlin
are deemed to have made any representations as to the suitability of this
software for any purpose nor are held responsible for any defects of
this software. THERE IS ABSOLUTELY NO WARRANTY FOR THIS SOFTWARE.

As a matter of courtesy, the authors request to be informed about uses
this software has found, about bugs in this software, and about any
improvements that may be of general interest.

Berlin, 28.11.1994
Jutta Degener
Carsten Bormann

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble
========

The licenses for most software are designed to take away your freedom

to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our
General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives
you legal permission to copy,
distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally
print such an
announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative
or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b. Accompany it with a written offer, valid for at least three years, to give any third-party, for a charge no more than your cost of physically performing source distribution, a complete

machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that
is normally distributed (in either
source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated
so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the

Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only

in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY

OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN
ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

==========

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

ONE LINE TO GIVE THE PROGRAM'S NAME AND A BRIEF IDEA OF WHAT IT DOES.
Copyright (C) YYYY NAME OF AUTHOR

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.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19YY NAME OF AUTHOR
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions;
type `show c' for details.

The hypothetical commands `show w' and `show c' should show the

appropriate parts of the General Public License. Of course, the
commands you use may be called something other than `show w' and `show
c'; they could even be mouse-clicks or menu items--whatever suits your
program.

You should also get your employer (if you work as a programmer) or your
school, if any, to sign a "copyright disclaimer" for the program, if
necessary. Here is a sample; alter the names:

   Yoyodyne, Inc., hereby disclaims all copyright interest in the program
   `Gnomovision' (which makes passes at compilers) written by James Hacker.

   SIGNATURE OF TY COON, 1 April 1989
   Ty Coon, President of Vice

This General Public License does not permit incorporating your program
into proprietary programs. If your program is a subroutine library,
you may consider it more useful to permit linking proprietary
applications with the library. If this is what you want to do,
 use the
GNU Library General Public License instead of this License.
Portable header file to contain:
>>>>>
/*
* PortAudio Portable Real-Time Audio Library
* PortAudio API Header File
* Latest version available at: http://www.portaudio.com
*
* Copyright (c) 1999-2006 Ross Bencina and Phil Burk
*
* Permission is hereby granted, free of charge, to any person obtaining
* a copy of this software and associated documentation files
* (the "Software"), to deal in the Software without restriction,
* including without limitation the rights to use, copy, modify, merge,
* publish, distribute, sublicense, and/or sell copies of the Software,
* and to permit persons to whom the Software is furnished to do so,
* subject to the following conditions:
*
* The above copyright notice and this permission notice shall be
* included in all copies or substantial portions of the Software.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
* MERCHANTABILITY, FITNESS FOR A PARTICULAR
 PURPOSE AND NONINFRINGEMENT.
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR
* ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
* CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION

* WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
*/

/*
* The text above constitutes the entire PortAudio license; however,
* the PortAudio community also makes the following non-binding requests:
*
* Any person wishing to distribute modifications to the Software is
* requested to send the modifications to the original developer so that
* they can be incorporated into the canonical version. It is also
* requested that these non-binding requests be included along with the
* license above.
*/
<<<<<

Implementation files to contain:
>>>>>
/*
* PortAudio Portable Real-Time Audio Library
* Latest version at: http://www.portaudio.com
* <platform> Implementation
* Copyright (c) 1999-2000 <author(s)>
*
* Permission is
 hereby granted, free of charge, to any person obtaining
* a copy of this software and associated documentation files
* (the "Software"), to deal in the Software without restriction,
* including without limitation the rights to use, copy, modify, merge,
* publish, distribute, sublicense, and/or sell copies of the Software,
* and to permit persons to whom the Software is furnished to do so,
* subject to the following conditions:
*
* The above copyright notice and this permission notice shall be
* included in all copies or substantial portions of the Software.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
* EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
* MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
* IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR
* ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
* CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
* WITH THE
 SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
*/

/*
* The text above constitutes the entire PortAudio license; however,

* the PortAudio community also makes the following non-binding requests:
*
* Any person wishing to distribute modifications to the Software is
* requested to send the modifications to the original developer so that
* they can be incorporated into the canonical version. It is also
* requested that these non-binding requests be included along with the
* license above.
*/
<<<<<
Copyright 2002-2008 Xiph.org Foundation
Copyright 2002-2008 Jean-Marc Valin
Copyright 2005-2007 Analog Devices Inc.
Copyright 2005-2008 Commonwealth Scientific and Industrial Research
Organisation (CSIRO)
Copyright 1993, 2002, 2006 David Rowe
Copyright 2003 EpicGames
Copyright 1992-1994 Jutta Degener, Carsten Bormann

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

- Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.

- 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.

- Neither the name of the Xiph.org Foundation nor the names of its
contributors may be used to endorse or promote products derived from
this software without specific prior written
permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
``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 FOUNDATION OR
CONTRIBUTORS 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.

1.10 wide-dhcpv6 1.3

1.10.1 Available under license :

$KAME: COPYRIGHT,v 1.2 2004/07/29 19:02:18 jinmei Exp $

Copyright (C) 1998-2004 WIDE Project.
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. Neither the name of the project nor the names of its contributors
  may be used to endorse or promote products derived from this software
  without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE PROJECT AND CONTRIBUTORS ``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 PROJECT OR CONTRIBUTORS 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.
K 14
svn:executable
V 1
*
END

1.11 wide-dhcpv6 1.3

1.11.1 Available under license :

$KAME: COPYRIGHT,v 1.2 2004/07/29 19:02:18 jinmei Exp $

Copyright (C) 1998-2004 WIDE Project.
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. Neither the name of the project nor the names of its contributors
  may be used to endorse or promote products derived from this software
  without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE PROJECT AND CONTRIBUTORS ``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 PROJECT OR CONTRIBUTORS 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.
K 14
svn:executable
V 1
*
END

1.12 hostapd 0.4.4

1.12.1 Available under license :

No license file was found, but licenses were detected in source scan.

/* zd_mac.c
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_mac.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_mac.c
No license file was found, but licenses were detected in source scan.

/* zd_netdev.h: Header for net device related functions.
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_netdev.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_netdev.h
No license file was found, but licenses were detected in source scan.

/*
* This software may only be used and distributed
* according to the terms of the GNU General Public License.
*
* For more details, see wavelan.c.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/i82586.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/wavelan.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/i82586.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wavelan.h

No license file was found, but licenses were detected in source scan.

```
/*
* Copyright 1996 The Board of Trustees of The Leland Stanford
* Junior University. All Rights Reserved.
*
* Permission to use, copy, modify, and distribute this
* software and its documentation for any purpose and without
* fee is hereby granted, provided that the above copyright
* notice appear in all copies.  Stanford University
* makes no representations about the suitability of this
* software for any purpose.  It is provided "as is" without
* express or implied warranty.
*
* strip.c This module implements Starmode Radio IP (STRIP)
*  for kernel-based devices like TTY.  It interfaces between a
*  raw TTY, and the kernel's INET protocol layers (via DDI).
*
* Version: @(#)strip.c 1.3 July 1997
*
* Author: Stuart Cheshire <cheshire@cs.stanford.edu>
*
* Fixes: v0.9 12th Feb 1996 (SC)
*  New byte stuffing (2+6 run-length encoding)
*  New watchdog timer task
*  New Protocol key (SIP0)
*
*  v0.9.1
 3rd March 1996 (SC)
*  Changed to dynamic device allocation -- no more compile
*  time (or boot time) limit on the number of STRIP devices.
*
*  v0.9.2 13th March 1996 (SC)
*  Uses arp cache lookups (but doesn't send arp packets yet)
*
*  v0.9.3 17th April 1996 (SC)
*  Fixed bug where STR_ERROR flag was getting set unneccessarily
*  (causing otherwise good packets to be unneccessarily dropped)
*
*  v0.9.4 27th April 1996 (SC)
*  First attempt at using "&COMMAND" Starmode AT commands
*
*  v0.9.5 29th May 1996 (SC)
*  First attempt at sending (unicast) ARP packets
*
*  v0.9.6 5th June 1996 (Elliot)
*  Put "message level" tags in every "printk" statement
*
```

```
*  v0.9.7 13th June 1996 (laik)
*  Added support for the /proc fs
*
*           v0.9.8 July 1996 (Mema)
*           Added packet logging
*
*           v1.0 November 1996 (SC)
*           Fixed (severe) memory leaks in the /proc fs code
*           Fixed race conditions in the
 logging code
*
*           v1.1 January 1997 (SC)
*           Deleted packet logging (use tcpdump instead)
*           Added support for Metricom Firmware v204 features
*           (like message checksums)
*
*           v1.2 January 1997 (SC)
*           Put portables list back in
*
*           v1.3 July 1997 (SC)
*           Made STRIP driver set the radio's baud rate automatically.
*           It is no longer necessarily to manually set the radio's
*           rate permanently to 115200 -- the driver handles setting
*           the rate automatically.
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/strip.c
No license file was found, but licenses were detected in source scan.

```
/*
* Host AP (software wireless LAN access point) driver for
* Intersil Prism2/2.5/3 - hostap.o module, common routines
*
* Copyright (c) 2001-2002, SSH Communications Security Corp and Jouni Malinen
* <jkmaline@cc.hut.fi>
* Copyright (c) 2002-2005, Jouni Malinen <jkmaline@cc.hut.fi>
*
* 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. See README and COPYING for
* more details.
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/.svn/text-base/hostap_main.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/hostap_main.c

No license file was found, but licenses were detected in source scan.

/**

Copyright(c) 2003 - 2006 Intel Corporation. All rights reserved.

This program is free software; you can redistribute it and/or modify it
under the terms of version 2 of the GNU General Public License as
published by the Free Software Foundation.

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.

Contact Information:
James P. Ketrenos
<ipw2100-admin@linux.intel.com>
Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124-6497

Portions of this file are based on the sample_* files provided by Wireless
Extensions 0.26 package and copyright (c) 1997-2003 Jean Tourrilhes
<jt@hpl.hp.com>

Portions of this file are based on the Host AP project,
Copyright (c) 2001-2002, SSH Communications Security Corp and Jouni Malinen
  <jkmaline@cc.hut.fi>
Copyright (c) 2002-2003, Jouni Malinen <jkmaline@cc.hut.fi>

Portions of ipw2100_mod_firmware_load, ipw2100_do_mod_firmware_load, and
ipw2100_fw_load are loosely based on drivers/sound/sound_firmware.c
available in the 2.4.25 kernel sources, and are copyright (c) Alan Cox

**/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/ipw2100.c
No license file was found, but licenses were detected in source scan.

/* zd_def.h
*

* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_def.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_def.h
No license file was found, but licenses were detected in source scan.

/*
* (C) 2004 Margit Schubert-While <margitsw@t-online.de>
*
* 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
*
* 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
*
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/prismcompat.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/prismcompat.h.svn-base
No license file was found, but licenses were detected in source scan.

/*===

  Aironet driver for 4500 and 4800 series cards

This code is released under both the GPL version 2 and BSD licenses.
Either license may be used. The respective licenses are found at
the end of this file.

This code was developed by Benjamin Reed <breed@users.sourceforge.net>
including portions of which come from the Aironet PC4500
Developer's Reference Manual and used with permission. Copyright
(C) 1999 Benjamin Reed. All Rights Reserved. Permission to use
code in the Developer's manual was granted for this driver by
Aironet.

In addition this module was derived from dummy_cs.
The initial developer of dummy_cs is David A. Hinds
<dahinds@users.sourceforge.net>. Portions created by David A. Hinds
are Copyright (C) 1999 David A. Hinds. All Rights Reserved.

==========*/

/*

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.

In addition:

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.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/airo_cs.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/airo_cs.c
No license file was found, but licenses were detected in source scan.

/*
* Copyright (C) 1997 Cullen Jennings
* Copyright (C) 1998 Elmer.Joandi@ut.ee, +37-255-13500
* GNU General Public License applies
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/arlan.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/arlan.h.svn-base
No license file was found, but licenses were detected in source scan.

/**

Copyright(c) 2003 - 2006 Intel Corporation. All rights reserved.

This program is free software; you can redistribute it and/or modify it
under the terms of version 2 of the GNU General Public License as
published by the Free Software Foundation.

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.

Contact Information:
James P. Ketrenos
<ipw2100-admin@linux.intel.com>
Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124-6497

Portions of this file are based on the sample_* files provided by Wireless
Extensions 0.26 package and copyright (c) 1997-2003 Jean Tourrilhes
<jt@hpl.hp.com>

Portions of this file are based on the Host AP project,
Copyright (c) 2001-2002, SSH Communications Security Corp and Jouni Malinen
<jkmaline@cc.hut.fi>
Copyright (c) 2002-2003, Jouni Malinen <jkmaline@cc.hut.fi>

Portions of ipw2100_mod_firmware_load, ipw2100_do_mod_firmware_load, and
ipw2100_fw_load are loosely based on drivers/sound/sound_firmware.c
available in the 2.4.25 kernel sources, and are copyright (c) Alan Cox

**********/
/*

Initial driver on which this is based was developed by Janusz Gorycki,
Maciej Urbaniak, and Maciej Sosnowski.

Promiscuous mode support added by Jacek Wysoczynski and Maciej Urbaniak.

Theory
of Operation

Tx - Commands and Data

Firmware and host share a circular queue of Transmit Buffer Descriptors (TBDs)
Each TBD contains a pointer to the physical (dma_addr_t) address of data being
sent to the firmware as well as the length of the data.

The host writes to the TBD queue at the WRITE index. The WRITE index points
to the _next_ packet to be written and is advanced when after the TBD has been
filled.

The firmware pulls from the TBD queue at the READ index. The READ index points
to the currently being read entry, and is advanced once the firmware is
done with a packet.

When data is sent to the firmware, the first TBD is used to indicate to the
firmware if a Command or Data is being sent. If it is Command, all of the
command information is contained within the physical address referred to by the
TBD. If it is Data, the first TBD indicates the type of data packet, number

of fragments, etc. The next TBD then referrs to the actual packet location.

The Tx flow cycle is as
 follows:

1) ipw2100_tx() is called by kernel with SKB to transmit
2) Packet is move from the tx_free_list and appended to the transmit pending
  list (tx_pend_list)
3) work is scheduled to move pending packets into the shared circular queue.
4) when placing packet in the circular queue, the incoming SKB is DMA mapped
  to a physical address. That address is entered into a TBD. Two TBDs are
  filled out. The first indicating a data packet, the second referring to the
  actual payload data.
5) the packet is removed from tx_pend_list and placed on the end of the
  firmware pending list (fw_pend_list)
6) firmware is notified that the WRITE index has
7) Once the firmware has processed the TBD, INTA is triggered.
8) For each Tx interrupt received from the firmware, the READ index is checked
  to see which TBDs are done being processed.
9) For each TBD that has been processed, the ISR pulls the oldest packet
  from the fw_pend_list.
10)The packet structure contained in the fw_pend_list
 is then used
  to unmap the DMA address and to free the SKB originally passed to the driver
  from the kernel.
11)The packet structure is placed onto the tx_free_list

The above steps are the same for commands, only the msg_free_list/msg_pend_list
are used instead of tx_free_list/tx_pend_list

...

Critical Sections / Locking :

There are two locks utilized. The first is the low level lock (priv->low_lock)
that protects the following:

- Access to the Tx/Rx queue lists via priv->low_lock. The lists are as follows:

 tx_free_list : Holds pre-allocated Tx buffers.
   TAIL modified in __ipw2100_tx_process()
   HEAD modified in ipw2100_tx()

 tx_pend_list : Holds used Tx buffers waiting to go into the TBD ring
   TAIL modified ipw2100_tx()
   HEAD modified by ipw2100_tx_send_data()

 msg_free_list : Holds pre-allocated Msg (Command) buffers

TAIL modified in __ipw2100_tx_process()
HEAD modified in ipw2100_hw_send_command()

msg_pend_list : Holds used Msg buffers waiting
to go into the TBD ring
TAIL modified in ipw2100_hw_send_command()
HEAD modified in ipw2100_tx_send_commands()

The flow of data on the TX side is as follows:

MSG_FREE_LIST + COMMAND => MSG_PEND_LIST => TBD => MSG_FREE_LIST
TX_FREE_LIST + DATA => TX_PEND_LIST => TBD => TX_FREE_LIST

The methods that work on the TBD ring are protected via priv->low_lock.

- The internal data state of the device itself
- Access to the firmware read/write indexes for the BD queues
and associated logic

All external entry functions are locked with the priv->action_lock to ensure
that only one external action is invoked at a time.

*/

```
#include <linux/compiler.h>
#include <linux/errno.h>
#include <linux/if_arp.h>
#include <linux/in6.h>
#include <linux/in.h>
#include <linux/ip.h>
#include <linux/kernel.h>
#include <linux/kmod.h>
#include <linux/module.h>
#include <linux/netdevice.h>
#include <linux/ethtool.h>
#include <linux/pci.h>
#include <linux/dma-mapping.h>
#include <linux/proc_fs.h>
#include
 <linux/skbuff.h>
#include <asm/uaccess.h>
#include <asm/io.h>
#include <linux/fs.h>
#include <linux/mm.h>
#include <linux/slab.h>
#include <linux/unistd.h>
#include <linux/stringify.h>
```

```
#include <linux/tcp.h>
#include <linux/types.h>
#include <linux/version.h>
#include <linux/time.h>
#include <linux/firmware.h>
#include <linux/acpi.h>
#include <linux/ctype.h>
#include <linux/latency.h>

#include "ipw2100.h"

#define IPW2100_VERSION "git-1.2.2"

#define DRV_NAME "ipw2100"
#define DRV_VERSION IPW2100_VERSION
#define DRV_DESCRIPTION "Intel(R) PRO/Wireless 2100 Network Driver"
#define DRV_COPYRIGHT "Copyright(c) 2003-2006 Intel Corporation"

/* Debugging stuff */
#ifdef CONFIG_IPW2100_DEBUG
#define CONFIG_IPW2100_RX_DEBUG /* Reception debugging */
#endif

MODULE_DESCRIPTION(DRV_DESCRIPTION);
MODULE_VERSION(DRV_VERSION);
MODULE_AUTHOR(DRV_COPYRIGHT);
MODULE_LICENSE("GPL");

static int debug = 0;
static int mode = 0;
static int channel = 0;
static int associate = 1;
static int
 disable = 0;
#ifdef CONFIG_PM
static struct ipw2100_fw ipw2100_firmware;
#endif

#include <linux/moduleparam.h>
module_param(debug, int, 0444);
module_param(mode, int, 0444);
module_param(channel, int, 0444);
module_param(associate, int, 0444);
module_param(disable, int, 0444);

MODULE_PARM_DESC(debug, "debug level");
MODULE_PARM_DESC(mode, "network mode (0=BSS,1=IBSS,2=Monitor)");
MODULE_PARM_DESC(channel, "channel");
```

```
MODULE_PARM_DESC(associate, "auto associate when scanning (default on)");
MODULE_PARM_DESC(disable, "manually disable the radio (default 0 [radio on])");

static u32 ipw2100_debug_level = IPW_DL_NONE;

#ifdef CONFIG_IPW2100_DEBUG
#define IPW_DEBUG(level, message...) \
do { \
 if (ipw2100_debug_level & (level)) { \
  printk(KERN_DEBUG "ipw2100: %c %s ", \
                    in_interrupt() ? 'I' : 'U',  __FUNCTION__); \
  printk(message); \
 } \
} while (0)
#else
#define IPW_DEBUG(level, message...) do {} while (0)
#endif    /* CONFIG_IPW2100_DEBUG */

#ifdef CONFIG_IPW2100_DEBUG
static
 const char *command_types[] = {
 "undefined",
 "unused",  /* HOST_ATTENTION */
 "HOST_COMPLETE",
 "unused",  /* SLEEP */
 "unused",  /* HOST_POWER_DOWN */
 "unused",
 "SYSTEM_CONFIG",
 "unused",  /* SET_IMR */
 "SSID",
 "MANDATORY_BSSID",
 "AUTHENTICATION_TYPE",
 "ADAPTER_ADDRESS",
 "PORT_TYPE",
 "INTERNATIONAL_MODE",
 "CHANNEL",
 "RTS_THRESHOLD",
 "FRAG_THRESHOLD",
 "POWER_MODE",
 "TX_RATES",
 "BASIC_TX_RATES",
 "WEP_KEY_INFO",
 "unused",
 "unused",
 "unused",
 "unused",
 "WEP_KEY_INDEX",
 "WEP_FLAGS",
```

```
 "ADD_MULTICAST",
 "CLEAR_ALL_MULTICAST",
 "BEACON_INTERVAL",
 "ATIM_WINDOW",
 "CLEAR_STATISTICS",
 "undefined",
 "undefined",
 "undefined",
 "undefined",
 "TX_POWER_INDEX",
 "undefined",
 "undefined",
 "undefined",
 "undefined",
 "undefined",
 "undefined",
 "BROADCAST_SCAN",
 "CARD_DISABLE",
 "PREFERRED_BSSID",
 "SET_SCAN_OPTIONS",
 "SCAN_DWELL_TIME",
 "SWEEP_TABLE",
 "AP_OR_STATION_TABLE",
 "GROUP_ORDINALS",
 "SHORT_RETRY_LIMIT",
 "LONG_RETRY_LIMIT",
 "unused",  /*
 SAVE_CALIBRATION */
 "unused",  /* RESTORE_CALIBRATION */
 "undefined",
 "undefined",
 "undefined",
 "HOST_PRE_POWER_DOWN",
 "unused",  /* HOST_INTERRUPT_COALESCING */
 "undefined",
 "CARD_DISABLE_PHY_OFF",
 "MSDU_TX_RATES" "undefined",
 "undefined",
 "SET_STATION_STAT_BITS",
 "CLEAR_STATIONS_STAT_BITS",
 "LEAP_ROGUE_MODE",
 "SET_SECURITY_INFORMATION",
 "DISASSOCIATION_BSSID",
 "SET_WPA_ASS_IE"
};
#endif

/* Pre-decl until we get the code solid and then we can clean it up */
```

```
static void ipw2100_tx_send_commands(struct ipw2100_priv *priv);
static void ipw2100_tx_send_data(struct ipw2100_priv *priv);
static int ipw2100_adapter_setup(struct ipw2100_priv *priv);

static void ipw2100_queues_initialize(struct ipw2100_priv *priv);
static void ipw2100_queues_free(struct ipw2100_priv *priv);
static int ipw2100_queues_allocate(struct ipw2100_priv *priv);

static int ipw2100_fw_download(struct ipw2100_priv *priv,
         struct ipw2100_fw *fw);
static int ipw2100_get_firmware(struct ipw2100_priv
 *priv,
   struct ipw2100_fw *fw);
static int ipw2100_get_fwversion(struct ipw2100_priv *priv, char *buf,
    size_t max);
static int ipw2100_get_ucodeversion(struct ipw2100_priv *priv, char *buf,
       size_t max);
static void ipw2100_release_firmware(struct ipw2100_priv *priv,
        struct ipw2100_fw *fw);
static int ipw2100_ucode_download(struct ipw2100_priv *priv,
     struct ipw2100_fw *fw);
static void ipw2100_wx_event_work(struct ipw2100_priv *priv);
static struct iw_statistics *ipw2100_wx_wireless_stats(struct net_device *dev);
static struct iw_handler_def ipw2100_wx_handler_def;

static inline void read_register(struct net_device *dev, u32 reg, u32 * val)
{
 *val = readl((void __iomem *)(dev->base_addr + reg));
 IPW_DEBUG_IO("r: 0x%08X => 0x%08X\n", reg, *val);
}

static inline void write_register(struct net_device *dev, u32 reg, u32 val)
{
 writel(val, (void __iomem *)(dev->base_addr + reg));
 IPW_DEBUG_IO("w: 0x%08X <= 0x%08X\n", reg, val);
}

static inline void read_register_word(struct
 net_device *dev, u32 reg,
         u16 * val)
{
 *val = readw((void __iomem *)(dev->base_addr + reg));
 IPW_DEBUG_IO("r: 0x%08X => %04X\n", reg, *val);
}

static inline void read_register_byte(struct net_device *dev, u32 reg, u8 * val)
{
 *val = readb((void __iomem *)(dev->base_addr + reg));
```

```
 IPW_DEBUG_IO("r: 0x%08X => %02X\n", reg, *val);
}

static inline void write_register_word(struct net_device *dev, u32 reg, u16 val)
{
 writew(val, (void __iomem *)(dev->base_addr + reg));
 IPW_DEBUG_IO("w: 0x%08X <= %04X\n", reg, val);
}

static inline void write_register_byte(struct net_device *dev, u32 reg, u8 val)
{
 writeb(val, (void __iomem *)(dev->base_addr + reg));
 IPW_DEBUG_IO("w: 0x%08X =< %02X\n", reg, val);
}

static inline void read_nic_dword(struct net_device *dev, u32 addr, u32 * val)
{
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
 read_register(dev, IPW_REG_INDIRECT_ACCESS_DATA, val);
}

static inline void
 write_nic_dword(struct net_device *dev, u32 addr, u32 val)
{
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
 write_register(dev, IPW_REG_INDIRECT_ACCESS_DATA, val);
}

static inline void read_nic_word(struct net_device *dev, u32 addr, u16 * val)
{
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
 read_register_word(dev, IPW_REG_INDIRECT_ACCESS_DATA, val);
}

static inline void write_nic_word(struct net_device *dev, u32 addr, u16 val)
{
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
 write_register_word(dev, IPW_REG_INDIRECT_ACCESS_DATA, val);
}

static inline void read_nic_byte(struct net_device *dev, u32 addr, u8 * val)
{
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
```

```
 read_register_byte(dev, IPW_REG_INDIRECT_ACCESS_DATA, val);
}

static inline void write_nic_byte(struct net_device
 *dev, u32 addr, u8 val)
{
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
 write_register_byte(dev, IPW_REG_INDIRECT_ACCESS_DATA, val);
}

static inline void write_nic_auto_inc_address(struct net_device *dev, u32 addr)
{
 write_register(dev, IPW_REG_AUTOINCREMENT_ADDRESS,
        addr & IPW_REG_INDIRECT_ADDR_MASK);
}

static inline void write_nic_dword_auto_inc(struct net_device *dev, u32 val)
{
 write_register(dev, IPW_REG_AUTOINCREMENT_DATA, val);
}

static void write_nic_memory(struct net_device *dev, u32 addr, u32 len,
       const u8 * buf)
{
 u32 aligned_addr;
 u32 aligned_len;
 u32 dif_len;
 u32 i;

 /* read first nibble byte by byte */
 aligned_addr = addr & (~0x3);
 dif_len = addr - aligned_addr;
 if (dif_len) {
  /* Start reading at aligned_addr + dif_len */
  write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
         aligned_addr);
  for (i = dif_len; i < 4; i++, buf++)
   write_register_byte(dev,
        IPW_REG_INDIRECT_ACCESS_DATA
 + i,
        *buf);

  len -= dif_len;
  aligned_addr += 4;
 }

 /* read DWs through autoincrement registers */
```

```
 write_register(dev, IPW_REG_AUTOINCREMENT_ADDRESS, aligned_addr);
 aligned_len = len & (~0x3);
 for (i = 0; i < aligned_len; i += 4, buf += 4, aligned_addr += 4)
  write_register(dev, IPW_REG_AUTOINCREMENT_DATA, *(u32 *) buf);

 /* copy the last nibble */
 dif_len = len - aligned_len;
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS, aligned_addr);
 for (i = 0; i < dif_len; i++, buf++)
  write_register_byte(dev, IPW_REG_INDIRECT_ACCESS_DATA + i,
        *buf);
}

static void read_nic_memory(struct net_device *dev, u32 addr, u32 len,
       u8 * buf)
{
 u32 aligned_addr;
 u32 aligned_len;
 u32 dif_len;
 u32 i;

 /* read first nibble byte by byte */
 aligned_addr = addr & (~0x3);
 dif_len = addr - aligned_addr;
 if (dif_len) {
  /* Start reading at aligned_addr + dif_len */
  write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS,
         aligned_addr);
  for
 (i = dif_len; i < 4; i++, buf++)
   read_register_byte(dev,
        IPW_REG_INDIRECT_ACCESS_DATA + i,
        buf);

  len -= dif_len;
  aligned_addr += 4;
 }

 /* read DWs through autoincrement registers */
 write_register(dev, IPW_REG_AUTOINCREMENT_ADDRESS, aligned_addr);
 aligned_len = len & (~0x3);
 for (i = 0; i < aligned_len; i += 4, buf += 4, aligned_addr += 4)
  read_register(dev, IPW_REG_AUTOINCREMENT_DATA, (u32 *) buf);

 /* copy the last nibble */
 dif_len = len - aligned_len;
 write_register(dev, IPW_REG_INDIRECT_ACCESS_ADDRESS, aligned_addr);
 for (i = 0; i < dif_len; i++, buf++)
```

```
 read_register_byte(dev, IPW_REG_INDIRECT_ACCESS_DATA + i, buf);
}

static inline int ipw2100_hw_is_adapter_in_system(struct net_device *dev)
{
 return (dev->base_addr &&
  (readl
   ((void __iomem *)(dev->base_addr +
       IPW_REG_DOA_DEBUG_AREA_START))
   == IPW_DATA_DOA_DEBUG_VALUE));
}

static int ipw2100_get_ordinal(struct ipw2100_priv *priv, u32 ord,
         void *val, u32 * len)
{
 struct
 ipw2100_ordinals *ordinals = &priv->ordinals;
 u32 addr;
 u32 field_info;
 u16 field_len;
 u16 field_count;
 u32 total_length;

 if (ordinals->table1_addr == 0) {
  printk(KERN_WARNING DRV_NAME ": attempt to use fw ordinals "
         "before they have been loaded.\n");
  return -EINVAL;
 }

 if (IS_ORDINAL_TABLE_ONE(ordinals, ord)) {
  if (*len < IPW_ORD_TAB_1_ENTRY_SIZE) {
   *len = IPW_ORD_TAB_1_ENTRY_SIZE;

   printk(KERN_WARNING DRV_NAME
          ": ordinal buffer length too small, need %zd\n",
          IPW_ORD_TAB_1_ENTRY_SIZE);

   return -EINVAL;
  }

  read_nic_dword(priv->net_dev,
         ordinals->table1_addr + (ord << 2), &addr);
  read_nic_dword(priv->net_dev, addr, val);

  *len = IPW_ORD_TAB_1_ENTRY_SIZE;

  return 0;
 }
```

```
if (IS_ORDINAL_TABLE_TWO(ordinals, ord)) {

 ord -= IPW_START_ORD_TAB_2;

 /* get the address of statistic */
 read_nic_dword(priv->net_dev,
        ordinals->table2_addr + (ord << 3), &addr);

 /* get the second DW of statistics ;

* two 16-bit words - first is length, second is count */
 read_nic_dword(priv->net_dev,
        ordinals->table2_addr + (ord << 3) + sizeof(u32),
        &field_info);

 /* get each entry length */
 field_len = *((u16 *) & field_info);

 /* get number of entries */
 field_count = *(((u16 *) & field_info) + 1);

 /* abort if no enought memory */
 total_length = field_len * field_count;
 if (total_length > *len) {
  *len = total_length;
  return -EINVAL;
 }

 *len = total_length;
 if (!total_length)
  return 0;

 /* read the ordinal data from the SRAM */
 read_nic_memory(priv->net_dev, addr, total_length, val);

 return 0;
}

printk(KERN_WARNING DRV_NAME ": ordinal %d neither in table 1 nor "
       "in table 2\n", ord);

return -EINVAL;
}

static int ipw2100_set_ordinal(struct ipw2100_priv *priv, u32 ord, u32 * val,
        u32 * len)
{
```

```
struct ipw2100_ordinals *ordinals = &priv->ordinals;
u32 addr;

if (IS_ORDINAL_TABLE_ONE(ordinals, ord)) {
 if (*len
!= IPW_ORD_TAB_1_ENTRY_SIZE) {
  *len = IPW_ORD_TAB_1_ENTRY_SIZE;
  IPW_DEBUG_INFO("wrong size\n");
  return -EINVAL;
 }

 read_nic_dword(priv->net_dev,
        ordinals->table1_addr + (ord << 2), &addr);

 write_nic_dword(priv->net_dev, addr, *val);

 *len = IPW_ORD_TAB_1_ENTRY_SIZE;

 return 0;
}

IPW_DEBUG_INFO("wrong table\n");
if (IS_ORDINAL_TABLE_TWO(ordinals, ord))
 return -EINVAL;

return -EINVAL;
}

static char *snprint_line(char *buf, size_t count,
     const u8 * data, u32 len, u32 ofs)
{
 int out, i, j, l;
 char c;

 out = snprintf(buf, count, "%08X", ofs);

 for (l = 0, i = 0; i < 2; i++) {
  out += snprintf(buf + out, count - out, " ");
  for (j = 0; j < 8 && l < len; j++, l++)
   out += snprintf(buf + out, count - out, "%02X ",
     data[(i * 8 + j)]);
  for (; j < 8; j++)
   out += snprintf(buf + out, count - out, "   ");
 }

 out += snprintf(buf + out, count - out, " ");
 for (l = 0, i = 0; i < 2; i++) {
  out += snprintf(buf + out, count - out, " ");
```

```
 for
(j = 0; j < 8 && l < len; j++, l++) {
  c = data[(i * 8 + j)];
  if (!isascii(c) || !isprint(c))
   c = '.';

  out += snprintf(buf + out, count - out, "%c", c);
 }

 for (; j < 8; j++)
  out += snprintf(buf + out, count - out, " ");
}

 return buf;
}

static void printk_buf(int level, const u8 * data, u32 len)
{
 char line[81];
 u32 ofs = 0;
 if (!(ipw2100_debug_level & level))
  return;

 while (len) {
  printk(KERN_DEBUG "%s\n",
         snprint_line(line, sizeof(line), &data[ofs],
        min(len, 16U), ofs));
  ofs += 16;
  len -= min(len, 16U);
 }
}

#define MAX_RESET_BACKOFF 10

static void schedule_reset(struct ipw2100_priv *priv)
{
 unsigned long now = get_seconds();

 /* If we haven't received a reset request within the backoff period,
  * then we can reset the backoff interval so this reset occurs
  * immediately */
 if (priv->reset_backoff &&
     (now - priv->last_reset > priv->reset_backoff))
  priv->reset_backoff = 0;

 priv->last_reset = get_seconds();

 if
```

```
 (!(priv->status & STATUS_RESET_PENDING)) {
  IPW_DEBUG_INFO("%s: Scheduling firmware restart (%ds).\n",
         priv->net_dev->name, priv->reset_backoff);
  netif_carrier_off(priv->net_dev);
  netif_stop_queue(priv->net_dev);
  priv->status |= STATUS_RESET_PENDING;
  if (priv->reset_backoff)
   queue_delayed_work(priv->workqueue, &priv->reset_work,
        priv->reset_backoff * HZ);
  else
   queue_work(priv->workqueue, &priv->reset_work);

  if (priv->reset_backoff < MAX_RESET_BACKOFF)
   priv->reset_backoff++;

  wake_up_interruptible(&priv->wait_command_queue);
 } else
  IPW_DEBUG_INFO("%s: Firmware restart already in progress.\n",
         priv->net_dev->name);

}

#define HOST_COMPLETE_TIMEOUT (2 * HZ)
static int ipw2100_hw_send_command(struct ipw2100_priv *priv,
       struct host_command *cmd)
{
 struct list_head *element;
 struct ipw2100_tx_packet *packet;
 unsigned long flags;
 int err = 0;

 IPW_DEBUG_HC("Sending %s command (#%d), %d bytes\n",
       command_types[cmd->host_command],
  cmd->host_command,
       cmd->host_command_length);
 printk_buf(IPW_DL_HC, (u8 *) cmd->host_command_parameters,
     cmd->host_command_length);

 spin_lock_irqsave(&priv->low_lock, flags);

 if (priv->fatal_error) {
  IPW_DEBUG_INFO
      ("Attempt to send command while hardware in fatal error condition.\n");
  err = -EIO;
  goto fail_unlock;
 }

 if (!(priv->status & STATUS_RUNNING)) {
```

```
 IPW_DEBUG_INFO
    ("Attempt to send command while hardware is not running.\n");
 err = -EIO;
 goto fail_unlock;
}

if (priv->status & STATUS_CMD_ACTIVE) {
 IPW_DEBUG_INFO
    ("Attempt to send command while another command is pending.\n");
 err = -EBUSY;
 goto fail_unlock;
}

if (list_empty(&priv->msg_free_list)) {
 IPW_DEBUG_INFO("no available msg buffers\n");
 goto fail_unlock;
}

priv->status |= STATUS_CMD_ACTIVE;
priv->messages_sent++;

element = priv->msg_free_list.next;

packet = list_entry(element, struct ipw2100_tx_packet, list);
packet->jiffy_start = jiffies;

/*
initialize the firmware command packet */
packet->info.c_struct.cmd->host_command_reg = cmd->host_command;
packet->info.c_struct.cmd->host_command_reg1 = cmd->host_command1;
packet->info.c_struct.cmd->host_command_len_reg =
   cmd->host_command_length;
packet->info.c_struct.cmd->sequence = cmd->host_command_sequence;

memcpy(packet->info.c_struct.cmd->host_command_params_reg,
       cmd->host_command_parameters,
       sizeof(packet->info.c_struct.cmd->host_command_params_reg));

list_del(element);
DEC_STAT(&priv->msg_free_stat);

list_add_tail(element, &priv->msg_pend_list);
INC_STAT(&priv->msg_pend_stat);

ipw2100_tx_send_commands(priv);
ipw2100_tx_send_data(priv);

spin_unlock_irqrestore(&priv->low_lock, flags);
```

```
/*
 * We must wait for this command to complete before another
 * command can be sent...  but if we wait more than 3 seconds
 * then there is a problem.
 */

err =
    wait_event_interruptible_timeout(priv->wait_command_queue,
        !(priv->

      status & STATUS_CMD_ACTIVE),
        HOST_COMPLETE_TIMEOUT);

if (err == 0) {
 IPW_DEBUG_INFO("Command completion failed out after %dms.\n",
        1000 * (HOST_COMPLETE_TIMEOUT / HZ));
 priv->fatal_error = IPW2100_ERR_MSG_TIMEOUT;
 priv->status &= ~STATUS_CMD_ACTIVE;
 schedule_reset(priv);
 return -EIO;
}

if (priv->fatal_error) {
 printk(KERN_WARNING DRV_NAME ": %s: firmware fatal error\n",
        priv->net_dev->name);
 return -EIO;
}

/* !!!!! HACK TEST !!!!!
 * When lots of debug trace statements are enabled, the driver
 * doesn't seem to have as many firmware restart cycles...
 *
 * As a test, we're sticking in a 1/100s delay here */
schedule_timeout_uninterruptible(msecs_to_jiffies(10));

return 0;

    fail_unlock:
spin_unlock_irqrestore(&priv->low_lock, flags);

return err;
}

/*
* Verify the values and data access of the hardware
* No locks needed or used.  No functions called.
*/
```

```
static int ipw2100_verify(struct ipw2100_priv *priv)
{
 u32
 data1, data2;
 u32 address;

 u32 val1 = 0x76543210;
 u32 val2 = 0xFEDCBA98;

 /* Domain 0 check - all values should be DOA_DEBUG */
 for (address = IPW_REG_DOA_DEBUG_AREA_START;
     address < IPW_REG_DOA_DEBUG_AREA_END; address += sizeof(u32)) {
  read_register(priv->net_dev, address, &data1);
  if (data1 != IPW_DATA_DOA_DEBUG_VALUE)
   return -EIO;
 }

 /* Domain 1 check - use arbitrary read/write compare  */
 for (address = 0; address < 5; address++) {
  /* The memory area is not used now */
  write_register(priv->net_dev, IPW_REG_DOMAIN_1_OFFSET + 0x32,
         val1);
  write_register(priv->net_dev, IPW_REG_DOMAIN_1_OFFSET + 0x36,
         val2);
  read_register(priv->net_dev, IPW_REG_DOMAIN_1_OFFSET + 0x32,
        &data1);
  read_register(priv->net_dev, IPW_REG_DOMAIN_1_OFFSET + 0x36,
        &data2);
  if (val1 == data1 && val2 == data2)
   return 0;
 }

 return -EIO;
}

/*
*
* Loop until the CARD_DISABLED bit is the same value as the
* supplied parameter
*
 * TODO: See if it would be more efficient to do a wait/wake
*       cycle and have the completion event trigger the wakeup
*
*/
#define IPW_CARD_DISABLE_COMPLETE_WAIT      100 // 100 milli
static int ipw2100_wait_for_card_state(struct ipw2100_priv *priv, int state)
{
 int i;
```

```
 u32 card_state;
 u32 len = sizeof(card_state);
 int err;

 for (i = 0; i <= IPW_CARD_DISABLE_COMPLETE_WAIT * 1000; i += 50) {
  err = ipw2100_get_ordinal(priv, IPW_ORD_CARD_DISABLED,
      &card_state, &len);
  if (err) {
   IPW_DEBUG_INFO("Query of CARD_DISABLED ordinal "
         "failed.\n");
   return 0;
  }

  /* We'll break out if either the HW state says it is
   * in the state we want, or if HOST_COMPLETE command
   * finishes */
  if ((card_state == state) ||
     ((priv->status & STATUS_ENABLED) ?
      IPW_HW_STATE_ENABLED : IPW_HW_STATE_DISABLED) == state) {
   if (state == IPW_HW_STATE_ENABLED)
    priv->status |= STATUS_ENABLED;
   else
    priv->status &= ~STATUS_ENABLED;

   return 0;
  }

  udelay(50);
 }

 IPW_DEBUG_INFO("ipw2100_wait_for_card_state
 to %s state timed out\n",
        state ? "DISABLED" : "ENABLED");
 return -EIO;
}

/*****************************************************************************
   Procedure   :   sw_reset_and_clock
   Purpose     :   Asserts s/w reset, asserts clock initialization
                   and waits for clock stabilization
*****************************************************************************/
static int sw_reset_and_clock(struct ipw2100_priv *priv)
{
 int i;
 u32 r;

 // assert s/w reset
 write_register(priv->net_dev, IPW_REG_RESET_REG,
```

```
        IPW_AUX_HOST_RESET_REG_SW_RESET);

    // wait for clock stabilization
    for (i = 0; i < 1000; i++) {
     udelay(IPW_WAIT_RESET_ARC_COMPLETE_DELAY);

     // check clock ready bit
     read_register(priv->net_dev, IPW_REG_RESET_REG, &r);
     if (r & IPW_AUX_HOST_RESET_REG_PRINCETON_RESET)
      break;
    }

    if (i == 1000)
     return -EIO; // TODO: better error value

    /* set "initialization complete" bit to move
    adapter to
     * D0 state */
    write_register(priv->net_dev, IPW_REG_GP_CNTRL,
           IPW_AUX_HOST_GP_CNTRL_BIT_INIT_DONE);

    /* wait for clock stabilization */
    for (i = 0; i < 10000; i++) {
     udelay(IPW_WAIT_CLOCK_STABILIZATION_DELAY * 4);

     /* check clock ready bit */
     read_register(priv->net_dev, IPW_REG_GP_CNTRL, &r);
     if (r & IPW_AUX_HOST_GP_CNTRL_BIT_CLOCK_READY)
      break;
    }

    if (i == 10000)
     return -EIO; /* TODO: better error value */

    /* set D0 standby bit */
    read_register(priv->net_dev, IPW_REG_GP_CNTRL, &r);
    write_register(priv->net_dev, IPW_REG_GP_CNTRL,
           r | IPW_AUX_HOST_GP_CNTRL_BIT_HOST_ALLOWS_STANDBY);

    return 0;
    }

    /******************************************************************
      Procedure   :   ipw2100_download_firmware
      Purpose     :   Initiaze adapter after power on.
                      The sequence is:
                      1. assert s/w reset first!
                      2. awake clocks & wait for clock stabilization
```

```
  3. hold ARC (don't ask me why...)
                4. load Dino ucode and reset/clock init again
                5. zero-out shared mem
                6. download f/w
******************************************************************************/
static int ipw2100_download_firmware(struct ipw2100_priv *priv)
{
 u32 address;
 int err;

#ifndef CONFIG_PM
 /* Fetch the firmware and microcode */
 struct ipw2100_fw ipw2100_firmware;
#endif

 if (priv->fatal_error) {
  IPW_DEBUG_ERROR("%s: ipw2100_download_firmware called after "
    "fatal error %d.  Interface must be brought down.\n",
    priv->net_dev->name, priv->fatal_error);
  return -EINVAL;
 }
#ifdef CONFIG_PM
 if (!ipw2100_firmware.version) {
  err = ipw2100_get_firmware(priv, &ipw2100_firmware);
  if (err) {
   IPW_DEBUG_ERROR("%s: ipw2100_get_firmware failed: %d\n",
     priv->net_dev->name, err);
   priv->fatal_error = IPW2100_ERR_FW_LOAD;
   goto fail;
  }
 }
#else
 err = ipw2100_get_firmware(priv, &ipw2100_firmware);
 if (err)
 {
  IPW_DEBUG_ERROR("%s: ipw2100_get_firmware failed: %d\n",
    priv->net_dev->name, err);
  priv->fatal_error = IPW2100_ERR_FW_LOAD;
  goto fail;
 }
#endif
 priv->firmware_version = ipw2100_firmware.version;

 /* s/w reset and clock stabilization */
 err = sw_reset_and_clock(priv);
 if (err) {
  IPW_DEBUG_ERROR("%s: sw_reset_and_clock failed: %d\n",
```

```
   priv->net_dev->name, err);
 goto fail;
 }

 err = ipw2100_verify(priv);
 if (err) {
  IPW_DEBUG_ERROR("%s: ipw2100_verify failed: %d\n",
    priv->net_dev->name, err);
  goto fail;
 }

 /* Hold ARC */
 write_nic_dword(priv->net_dev,
   IPW_INTERNAL_REGISTER_HALT_AND_RESET, 0x80000000);

 /* allow ARC to run */
 write_register(priv->net_dev, IPW_REG_RESET_REG, 0);

 /* load microcode */
 err = ipw2100_ucode_download(priv, &ipw2100_firmware);
 if (err) {
  printk(KERN_ERR DRV_NAME ": %s: Error loading microcode: %d\n",
         priv->net_dev->name, err);
  goto fail;
 }

 /* release ARC */
 write_nic_dword(priv->net_dev,
   IPW_INTERNAL_REGISTER_HALT_AND_RESET,
 0x00000000);

 /* s/w reset and clock stabilization (again!!!) */
 err = sw_reset_and_clock(priv);
 if (err) {
  printk(KERN_ERR DRV_NAME
         ": %s: sw_reset_and_clock failed: %d\n",
         priv->net_dev->name, err);
  goto fail;
 }

 /* load f/w */
 err = ipw2100_fw_download(priv, &ipw2100_firmware);
 if (err) {
  IPW_DEBUG_ERROR("%s: Error loading firmware: %d\n",
    priv->net_dev->name, err);
  goto fail;
 }
#ifndef CONFIG_PM
```

```
 /*
  * When the .resume method of the driver is called, the other
  * part of the system, i.e. the ide driver could still stay in
  * the suspend stage. This prevents us from loading the firmware
  * from the disk.  --YZ
  */

 /* free any storage allocated for firmware image */
 ipw2100_release_firmware(priv, &ipw2100_firmware);
#endif

 /* zero out Domain 1 area indirectly (Si requirement) */
 for (address = IPW_HOST_FW_SHARED_AREA0;
      address < IPW_HOST_FW_SHARED_AREA0_END; address += 4)
  write_nic_dword(priv->net_dev,
 address, 0);
 for (address = IPW_HOST_FW_SHARED_AREA1;
      address < IPW_HOST_FW_SHARED_AREA1_END; address += 4)
  write_nic_dword(priv->net_dev, address, 0);
 for (address = IPW_HOST_FW_SHARED_AREA2;
      address < IPW_HOST_FW_SHARED_AREA2_END; address += 4)
  write_nic_dword(priv->net_dev, address, 0);
 for (address = IPW_HOST_FW_SHARED_AREA3;
      address < IPW_HOST_FW_SHARED_AREA3_END; address += 4)
  write_nic_dword(priv->net_dev, address, 0);
 for (address = IPW_HOST_FW_INTERRUPT_AREA;
      address < IPW_HOST_FW_INTERRUPT_AREA_END; address += 4)
  write_nic_dword(priv->net_dev, address, 0);

 return 0;

     fail:
 ipw2100_release_firmware(priv, &ipw2100_firmware);
 return err;
}

static inline void ipw2100_enable_interrupts(struct ipw2100_priv *priv)
{
 if (priv->status & STATUS_INT_ENABLED)
  return;
 priv->status |= STATUS_INT_ENABLED;
 write_register(priv->net_dev, IPW_REG_INTA_MASK, IPW_INTERRUPT_MASK);
}

static inline void ipw2100_disable_interrupts(struct ipw2100_priv
 *priv)
{
 if (!(priv->status & STATUS_INT_ENABLED))
```

```
 return;
 priv->status &= ~STATUS_INT_ENABLED;
 write_register(priv->net_dev, IPW_REG_INTA_MASK, 0x0);
}

static void ipw2100_initialize_ordinals(struct ipw2100_priv *priv)
{
 struct ipw2100_ordinals *ord = &priv->ordinals;

 IPW_DEBUG_INFO("enter\n");

 read_register(priv->net_dev, IPW_MEM_HOST_SHARED_ORDINALS_TABLE_1,
       &ord->table1_addr);

 read_register(priv->net_dev, IPW_MEM_HOST_SHARED_ORDINALS_TABLE_2,
       &ord->table2_addr);

 read_nic_dword(priv->net_dev, ord->table1_addr, &ord->table1_size);
 read_nic_dword(priv->net_dev, ord->table2_addr, &ord->table2_size);

 ord->table2_size &= 0x0000FFFF;

 IPW_DEBUG_INFO("table 1 size: %d\n", ord->table1_size);
 IPW_DEBUG_INFO("table 2 size: %d\n", ord->table2_size);
 IPW_DEBUG_INFO("exit\n");
}

static inline void ipw2100_hw_set_gpio(struct ipw2100_priv *priv)
{
 u32 reg = 0;
 /*
  * Set GPIO 3 writable by FW; GPIO 1 writable
  * by driver and enable clock
  */
 reg
 = (IPW_BIT_GPIO_GPIO3_MASK | IPW_BIT_GPIO_GPIO1_ENABLE |
       IPW_BIT_GPIO_LED_OFF);
 write_register(priv->net_dev, IPW_REG_GPIO, reg);
}

static int rf_kill_active(struct ipw2100_priv *priv)
{
#define MAX_RF_KILL_CHECKS 5
#define RF_KILL_CHECK_DELAY 40

 unsigned short value = 0;
 u32 reg = 0;
 int i;
```

```
 if (!(priv->hw_features & HW_FEATURE_RFKILL)) {
  priv->status &= ~STATUS_RF_KILL_HW;
  return 0;
 }

 for (i = 0; i < MAX_RF_KILL_CHECKS; i++) {
  udelay(RF_KILL_CHECK_DELAY);
  read_register(priv->net_dev, IPW_REG_GPIO, &reg);
  value = (value << 1) | ((reg & IPW_BIT_GPIO_RF_KILL) ? 0 : 1);
 }

 if (value == 0)
  priv->status |= STATUS_RF_KILL_HW;
 else
  priv->status &= ~STATUS_RF_KILL_HW;

 return (value == 0);
}

static int ipw2100_get_hw_features(struct ipw2100_priv *priv)
{
 u32 addr, len;
 u32 val;

 /*
  * EEPROM_SRAM_DB_START_ADDRESS using ordinal in ordinal table 1
  */
 len = sizeof(addr);
 if (ipw2100_get_ordinal
     (priv, IPW_ORD_EEPROM_SRAM_DB_BLOCK_START_ADDRESS,
 &addr, &len)) {
  IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
         __LINE__);
  return -EIO;
 }

 IPW_DEBUG_INFO("EEPROM address: %08X\n", addr);

 /*
  * EEPROM version is the byte at offset 0xfd in firmware
  * We read 4 bytes, then shift out the byte we actually want */
 read_nic_dword(priv->net_dev, addr + 0xFC, &val);
 priv->eeprom_version = (val >> 24) & 0xFF;
 IPW_DEBUG_INFO("EEPROM version: %d\n", priv->eeprom_version);

 /*
  *  HW RF Kill enable is bit 0 in byte at offset 0x21 in firmware
```

```
 *
 *  notice that the EEPROM bit is reverse polarity, i.e.
 *     bit = 0  signifies HW RF kill switch is supported
 *     bit = 1  signifies HW RF kill switch is NOT supported
 */
read_nic_dword(priv->net_dev, addr + 0x20, &val);
if (!((val >> 24) & 0x01))
 priv->hw_features |= HW_FEATURE_RFKILL;

IPW_DEBUG_INFO("HW RF Kill: %ssupported.\n",
       (priv->hw_features & HW_FEATURE_RFKILL) ? "" : "not ");

return 0;
}

/*
* Start firmware execution after power on
 and intialization
* The sequence is:
*  1. Release ARC
*  2. Wait for f/w initialization completes;
*/
static int ipw2100_start_adapter(struct ipw2100_priv *priv)
{
 int i;
 u32 inta, inta_mask, gpio;

 IPW_DEBUG_INFO("enter\n");

 if (priv->status & STATUS_RUNNING)
  return 0;

 /*
  * Initialize the hw - drive adapter to DO state by setting
  * init_done bit. Wait for clk_ready bit and Download
  * fw & dino ucode
  */
 if (ipw2100_download_firmware(priv)) {
  printk(KERN_ERR DRV_NAME
       ": %s: Failed to power on the adapter.\n",
       priv->net_dev->name);
  return -EIO;
 }

 /* Clear the Tx, Rx and Msg queues and the r/w indexes
  * in the firmware RBD and TBD ring queue */
 ipw2100_queues_initialize(priv);
```

```
ipw2100_hw_set_gpio(priv);

/* TODO -- Look at disabling interrupts here to make sure none
 * get fired during FW initialization */

/* Release ARC - clear reset bit */
write_register(priv->net_dev, IPW_REG_RESET_REG, 0);

/* wait for f/w intialization complete
*/
IPW_DEBUG_FW("Waiting for f/w initialization to complete...\n");
i = 5000;
do {
 schedule_timeout_uninterruptible(msecs_to_jiffies(40));
 /* Todo... wait for sync command ... */

 read_register(priv->net_dev, IPW_REG_INTA, &inta);

 /* check "init done" bit */
 if (inta & IPW2100_INTA_FW_INIT_DONE) {
  /* reset "init done" bit */
  write_register(priv->net_dev, IPW_REG_INTA,
         IPW2100_INTA_FW_INIT_DONE);
  break;
 }

 /* check error conditions : we check these after the firmware
  * check so that if there is an error, the interrupt handler
  * will see it and the adapter will be reset */
 if (inta &
     (IPW2100_INTA_FATAL_ERROR | IPW2100_INTA_PARITY_ERROR)) {
  /* clear error conditions */
  write_register(priv->net_dev, IPW_REG_INTA,
         IPW2100_INTA_FATAL_ERROR |
         IPW2100_INTA_PARITY_ERROR);
 }
} while (i--);

/* Clear out any pending INTAs since we aren't supposed to have
 * interrupts enabled at this point... */
read_register(priv->net_dev,
IPW_REG_INTA, &inta);
read_register(priv->net_dev, IPW_REG_INTA_MASK, &inta_mask);
inta &= IPW_INTERRUPT_MASK;
/* Clear out any pending interrupts */
if (inta & inta_mask)
 write_register(priv->net_dev, IPW_REG_INTA, inta);
```

```
 IPW_DEBUG_FW("f/w initialization complete: %s\n",
      i ? "SUCCESS" : "FAILED");

 if (!i) {
  printk(KERN_WARNING DRV_NAME
         ": %s: Firmware did not initialize.\n",
         priv->net_dev->name);
  return -EIO;
 }

 /* allow firmware to write to GPIO1 & GPIO3 */
 read_register(priv->net_dev, IPW_REG_GPIO, &gpio);

 gpio |= (IPW_BIT_GPIO_GPIO1_MASK | IPW_BIT_GPIO_GPIO3_MASK);

 write_register(priv->net_dev, IPW_REG_GPIO, gpio);

 /* Ready to receive commands */
 priv->status |= STATUS_RUNNING;

 /* The adapter has been reset; we are not associated */
 priv->status &= ~(STATUS_ASSOCIATING | STATUS_ASSOCIATED);

 IPW_DEBUG_INFO("exit\n");

 return 0;
}

static inline void ipw2100_reset_fatalerror(struct ipw2100_priv *priv)
{
 if (!priv->fatal_error)
  return;

 priv->fatal_errors[priv->fatal_index++]
 = priv->fatal_error;
 priv->fatal_index %= IPW2100_ERROR_QUEUE;
 priv->fatal_error = 0;
}

/* NOTE: Our interrupt is disabled when this method is called */
static int ipw2100_power_cycle_adapter(struct ipw2100_priv *priv)
{
 u32 reg;
 int i;

 IPW_DEBUG_INFO("Power cycling the hardware.\n");

 ipw2100_hw_set_gpio(priv);
```

```
/* Step 1. Stop Master Assert */
write_register(priv->net_dev, IPW_REG_RESET_REG,
       IPW_AUX_HOST_RESET_REG_STOP_MASTER);

/* Step 2. Wait for stop Master Assert
 *        (not more then 50us, otherwise ret error */
i = 5;
do {
 udelay(IPW_WAIT_RESET_MASTER_ASSERT_COMPLETE_DELAY);
 read_register(priv->net_dev, IPW_REG_RESET_REG, &reg);

 if (reg & IPW_AUX_HOST_RESET_REG_MASTER_DISABLED)
  break;
} while (i--);

priv->status &= ~STATUS_RESET_PENDING;

if (!i) {
 IPW_DEBUG_INFO
    ("exit - waited too long for master assert stop\n");
 return -EIO;
}

write_register(priv->net_dev,
IPW_REG_RESET_REG,
       IPW_AUX_HOST_RESET_REG_SW_RESET);

/* Reset any fatal_error conditions */
ipw2100_reset_fatalerror(priv);

/* At this point, the adapter is now stopped and disabled */
priv->status &= ~(STATUS_RUNNING | STATUS_ASSOCIATING |
   STATUS_ASSOCIATED | STATUS_ENABLED);

 return 0;
}

/*
* Send the CARD_DISABLE_PHY_OFF comamnd to the card to disable it
*
* After disabling, if the card was associated, a STATUS_ASSN_LOST will be sent.
*
* STATUS_CARD_DISABLE_NOTIFICATION will be sent regardless of
* if STATUS_ASSN_LOST is sent.
*/
static int ipw2100_hw_phy_off(struct ipw2100_priv *priv)
{
```

```
#define HW_PHY_OFF_LOOP_DELAY (HZ / 5000)

 struct host_command cmd = {
  .host_command = CARD_DISABLE_PHY_OFF,
  .host_command_sequence = 0,
  .host_command_length = 0,
 };
 int err, i;
 u32 val1, val2;

 IPW_DEBUG_HC("CARD_DISABLE_PHY_OFF\n");

 /* Turn off the radio */
 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  return err;

 for (i = 0; i < 2500; i++)
 {
  read_nic_dword(priv->net_dev, IPW2100_CONTROL_REG, &val1);
  read_nic_dword(priv->net_dev, IPW2100_COMMAND, &val2);

  if ((val1 & IPW2100_CONTROL_PHY_OFF) &&
      (val2 & IPW2100_COMMAND_PHY_OFF))
   return 0;

  schedule_timeout_uninterruptible(HW_PHY_OFF_LOOP_DELAY);
 }

 return -EIO;
}

static int ipw2100_enable_adapter(struct ipw2100_priv *priv)
{
 struct host_command cmd = {
  .host_command = HOST_COMPLETE,
  .host_command_sequence = 0,
  .host_command_length = 0
 };
 int err = 0;

 IPW_DEBUG_HC("HOST_COMPLETE\n");

 if (priv->status & STATUS_ENABLED)
  return 0;

 mutex_lock(&priv->adapter_mutex);
```

```
 if (rf_kill_active(priv)) {
  IPW_DEBUG_HC("Command aborted due to RF kill active.\n");
  goto fail_up;
 }

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err) {
  IPW_DEBUG_INFO("Failed to send HOST_COMPLETE command\n");
  goto fail_up;
 }

 err = ipw2100_wait_for_card_state(priv, IPW_HW_STATE_ENABLED);
 if (err) {
  IPW_DEBUG_INFO("%s: card not responding to init command.\n",

       priv->net_dev->name);
  goto fail_up;
 }

 if (priv->stop_hang_check) {
  priv->stop_hang_check = 0;
  queue_delayed_work(priv->workqueue, &priv->hang_check, HZ / 2);
 }

     fail_up:
 mutex_unlock(&priv->adapter_mutex);
 return err;
}

static int ipw2100_hw_stop_adapter(struct ipw2100_priv *priv)
{
#define HW_POWER_DOWN_DELAY (msecs_to_jiffies(100))

 struct host_command cmd = {
  .host_command = HOST_PRE_POWER_DOWN,
  .host_command_sequence = 0,
  .host_command_length = 0,
 };
 int err, i;
 u32 reg;

 if (!(priv->status & STATUS_RUNNING))
  return 0;

 priv->status |= STATUS_STOPPING;

 /* We can only shut down the card if the firmware is operational.  So,
```

```
 * if we haven't reset since a fatal_error, then we can not send the
 * shutdown commands. */
if (!priv->fatal_error) {
 /* First, make sure the adapter is enabled so that the PHY_OFF
  * command can shut it down */
 ipw2100_enable_adapter(priv);

 err = ipw2100_hw_phy_off(priv);
 if (err)
  printk(KERN_WARNING
DRV_NAME
        ": Error disabling radio %d\n", err);

 /*
  * If in D0-standby mode going directly to D3 may cause a
  * PCI bus violation.  Therefore we must change out of the D0
  * state.
  *
  * Sending the PREPARE_FOR_POWER_DOWN will restrict the
  * hardware from going into standby mode and will transition
  * out of D0-standby if it is already in that state.
  *
  * STATUS_PREPARE_POWER_DOWN_COMPLETE will be sent by the
  * driver upon completion.  Once received, the driver can
  * proceed to the D3 state.
  *
  * Prepare for power down command to fw.  This command would
  * take HW out of D0-standby and prepare it for D3 state.
  *
  * Currently FW does not support event notification for this
  * event. Therefore, skip waiting for it.  Just wait a fixed
  * 100ms
  */
 IPW_DEBUG_HC("HOST_PRE_POWER_DOWN\n");

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  printk(KERN_WARNING DRV_NAME ": "
        "%s: Power down command failed: Error
%d\n",
        priv->net_dev->name, err);
 else
  schedule_timeout_uninterruptible(HW_POWER_DOWN_DELAY);
}

priv->status &= ~STATUS_ENABLED;

/*
```

```
 * Set GPIO 3 writable by FW; GPIO 1 writable
 * by driver and enable clock
 */
ipw2100_hw_set_gpio(priv);

/*
 * Power down adapter.  Sequence:
 * 1. Stop master assert (RESET_REG[9]=1)
 * 2. Wait for stop master (RESET_REG[8]==1)
 * 3. S/w reset assert (RESET_REG[7] = 1)
 */

/* Stop master assert */
write_register(priv->net_dev, IPW_REG_RESET_REG,
       IPW_AUX_HOST_RESET_REG_STOP_MASTER);

/* wait stop master not more than 50 usec.
 * Otherwise return error. */
for (i = 5; i > 0; i--) {
 udelay(10);

 /* Check master stop bit */
 read_register(priv->net_dev, IPW_REG_RESET_REG, &reg);

 if (reg & IPW_AUX_HOST_RESET_REG_MASTER_DISABLED)
  break;
}

if (i == 0)
 printk(KERN_WARNING DRV_NAME
       ": %s: Could now power down adapter.\n",
       priv->net_dev->name);

/* assert s/w reset */
write_register(priv->net_dev,
IPW_REG_RESET_REG,
       IPW_AUX_HOST_RESET_REG_SW_RESET);

priv->status &= ~(STATUS_RUNNING | STATUS_STOPPING);

return 0;
}

static int ipw2100_disable_adapter(struct ipw2100_priv *priv)
{
 struct host_command cmd = {
  .host_command = CARD_DISABLE,
  .host_command_sequence = 0,
```

```
 .host_command_length = 0
};
int err = 0;

IPW_DEBUG_HC("CARD_DISABLE\n");

if (!(priv->status & STATUS_ENABLED))
 return 0;

/* Make sure we clear the associated state */
priv->status &= ~(STATUS_ASSOCIATED | STATUS_ASSOCIATING);

if (!priv->stop_hang_check) {
 priv->stop_hang_check = 1;
 cancel_delayed_work(&priv->hang_check);
}

mutex_lock(&priv->adapter_mutex);

err = ipw2100_hw_send_command(priv, &cmd);
if (err) {
 printk(KERN_WARNING DRV_NAME
        ": exit - failed to send CARD_DISABLE command\n");
 goto fail_up;
}

err = ipw2100_wait_for_card_state(priv, IPW_HW_STATE_DISABLED);
if (err) {
 printk(KERN_WARNING DRV_NAME
        ": exit - card failed to change
to DISABLED\n");
 goto fail_up;
}

IPW_DEBUG_INFO("TODO: implement scan state machine\n");

    fail_up:
mutex_unlock(&priv->adapter_mutex);
return err;
}

static int ipw2100_set_scan_options(struct ipw2100_priv *priv)
{
 struct host_command cmd = {
  .host_command = SET_SCAN_OPTIONS,
  .host_command_sequence = 0,
  .host_command_length = 8
 };
```

```
 int err;

 IPW_DEBUG_INFO("enter\n");

 IPW_DEBUG_SCAN("setting scan options\n");

 cmd.host_command_parameters[0] = 0;

 if (!(priv->config & CFG_ASSOCIATE))
  cmd.host_command_parameters[0] |= IPW_SCAN_NOASSOCIATE;
 if ((priv->ieee->sec.flags & SEC_ENABLED) && priv->ieee->sec.enabled)
  cmd.host_command_parameters[0] |= IPW_SCAN_MIXED_CELL;
 if (priv->config & CFG_PASSIVE_SCAN)
  cmd.host_command_parameters[0] |= IPW_SCAN_PASSIVE;

 cmd.host_command_parameters[1] = priv->channel_mask;

 err = ipw2100_hw_send_command(priv, &cmd);

 IPW_DEBUG_HC("SET_SCAN_OPTIONS 0x%04X\n",
       cmd.host_command_parameters[0]);

 return err;
}

static
 int ipw2100_start_scan(struct ipw2100_priv *priv)
{
 struct host_command cmd = {
  .host_command = BROADCAST_SCAN,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 IPW_DEBUG_HC("START_SCAN\n");

 cmd.host_command_parameters[0] = 0;

 /* No scanning if in monitor mode */
 if (priv->ieee->iw_mode == IW_MODE_MONITOR)
  return 1;

 if (priv->status & STATUS_SCANNING) {
  IPW_DEBUG_SCAN("Scan requested while already in scan...\n");
  return 0;
 }
```

```
 IPW_DEBUG_INFO("enter\n");

 /* Not clearing here; doing so makes iwlist always return nothing...
  *
  * We should modify the table logic to use aging tables vs. clearing
  * the table on each scan start.
  */
 IPW_DEBUG_SCAN("starting scan\n");

 priv->status |= STATUS_SCANNING;
 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  priv->status &= ~STATUS_SCANNING;

 IPW_DEBUG_INFO("exit\n");

 return err;
}

static const struct ieee80211_geo ipw_geos[] = {
 {   /* Restricted */
  "---",
  .bg_channels = 14,
  .bg = {{2412,
 1}, {2417, 2}, {2422, 3},
  {2427, 4}, {2432, 5}, {2437, 6},
  {2442, 7}, {2447, 8}, {2452, 9},
  {2457, 10}, {2462, 11}, {2467, 12},
  {2472, 13}, {2484, 14}},
  },
};

static int ipw2100_up(struct ipw2100_priv *priv, int deferred)
{
 unsigned long flags;
 int rc = 0;
 u32 lock;
 u32 ord_len = sizeof(lock);

 /* Quite if manually disabled. */
 if (priv->status & STATUS_RF_KILL_SW) {
  IPW_DEBUG_INFO("%s: Radio is disabled by Manual Disable "
         "switch\n", priv->net_dev->name);
  return 0;
 }

 /* the ipw2100 hardware really doesn't want power management delays
  * longer than 175usec
```

```
 */
modify_acceptable_latency("ipw2100", 175);

/* If the interrupt is enabled, turn it off... */
spin_lock_irqsave(&priv->low_lock, flags);
ipw2100_disable_interrupts(priv);

/* Reset any fatal_error conditions */
ipw2100_reset_fatalerror(priv);
spin_unlock_irqrestore(&priv->low_lock, flags);

if (priv->status & STATUS_POWERED ||
   (priv->status & STATUS_RESET_PENDING)) {
 /*
Power cycle the card ... */
 if (ipw2100_power_cycle_adapter(priv)) {
  printk(KERN_WARNING DRV_NAME
         ": %s: Could not cycle adapter.\n",
         priv->net_dev->name);
  rc = 1;
  goto exit;
 }
} else
 priv->status |= STATUS_POWERED;

/* Load the firmware, start the clocks, etc. */
if (ipw2100_start_adapter(priv)) {
 printk(KERN_ERR DRV_NAME
        ": %s: Failed to start the firmware.\n",
        priv->net_dev->name);
 rc = 1;
 goto exit;
}

ipw2100_initialize_ordinals(priv);

/* Determine capabilities of this particular HW configuration */
if (ipw2100_get_hw_features(priv)) {
 printk(KERN_ERR DRV_NAME
        ": %s: Failed to determine HW features.\n",
        priv->net_dev->name);
 rc = 1;
 goto exit;
}

/* Initialize the geo */
if (ieee80211_set_geo(priv->ieee, &ipw_geos[0])) {
 printk(KERN_WARNING DRV_NAME "Could not set geo\n");
```

```
 return 0;
}
priv->ieee->freq_band = IEEE80211_24GHZ_BAND;

lock = LOCK_NONE;
if (ipw2100_set_ordinal(priv, IPW_ORD_PERS_DB_LOCK,
&lock, &ord_len)) {
 printk(KERN_ERR DRV_NAME
        ": %s: Failed to clear ordinal lock.\n",
        priv->net_dev->name);
 rc = 1;
 goto exit;
}

priv->status &= ~STATUS_SCANNING;

if (rf_kill_active(priv)) {
 printk(KERN_INFO "%s: Radio is disabled by RF switch.\n",
        priv->net_dev->name);

 if (priv->stop_rf_kill) {
  priv->stop_rf_kill = 0;
  queue_delayed_work(priv->workqueue, &priv->rf_kill, HZ);
 }

 deferred = 1;
}

/* Turn on the interrupt so that commands can be processed */
ipw2100_enable_interrupts(priv);

/* Send all of the commands that must be sent prior to
 * HOST_COMPLETE */
if (ipw2100_adapter_setup(priv)) {
 printk(KERN_ERR DRV_NAME ": %s: Failed to start the card.\n",
        priv->net_dev->name);
 rc = 1;
 goto exit;
}

if (!deferred) {
 /* Enable the adapter - sends HOST_COMPLETE */
 if (ipw2100_enable_adapter(priv)) {
  printk(KERN_ERR DRV_NAME ": "
         "%s: failed in call to enable adapter.\n",

      priv->net_dev->name);
  ipw2100_hw_stop_adapter(priv);
```

```
  rc = 1;
  goto exit;
 }

 /* Start a scan . . . */
 ipw2100_set_scan_options(priv);
 ipw2100_start_scan(priv);
}

    exit:
return rc;
}

/* Called by register_netdev() */
static int ipw2100_net_init(struct net_device *dev)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 return ipw2100_up(priv, 1);
}

static void ipw2100_down(struct ipw2100_priv *priv)
{
 unsigned long flags;
 union iwreq_data wrqu = {
  .ap_addr = {
      .sa_family = ARPHRD_ETHER}
 };
 int associated = priv->status & STATUS_ASSOCIATED;

 /* Kill the RF switch timer */
 if (!priv->stop_rf_kill) {
  priv->stop_rf_kill = 1;
  cancel_delayed_work(&priv->rf_kill);
 }

 /* Kill the firmare hang check timer */
 if (!priv->stop_hang_check) {
  priv->stop_hang_check = 1;
  cancel_delayed_work(&priv->hang_check);
 }

 /* Kill any pending resets */
 if (priv->status & STATUS_RESET_PENDING)
  cancel_delayed_work(&priv->reset_work);

 /* Make sure the
 interrupt is on so that FW commands will be
  * processed correctly */
```

```
 spin_lock_irqsave(&priv->low_lock, flags);
 ipw2100_enable_interrupts(priv);
 spin_unlock_irqrestore(&priv->low_lock, flags);

 if (ipw2100_hw_stop_adapter(priv))
  printk(KERN_ERR DRV_NAME ": %s: Error stopping adapter.\n",
         priv->net_dev->name);

 /* Do not disable the interrupt until _after_ we disable
  * the adaptor.  Otherwise the CARD_DISABLE command will never
  * be ack'd by the firmware */
 spin_lock_irqsave(&priv->low_lock, flags);
 ipw2100_disable_interrupts(priv);
 spin_unlock_irqrestore(&priv->low_lock, flags);

 modify_acceptable_latency("ipw2100", INFINITE_LATENCY);

#ifdef ACPI_CSTATE_LIMIT_DEFINED
 if (priv->config & CFG_C3_DISABLED) {
  IPW_DEBUG_INFO(": Resetting C3 transitions.\n");
  acpi_set_cstate_limit(priv->cstate_limit);
  priv->config &= ~CFG_C3_DISABLED;
 }
#endif

 /* We have to signal any supplicant if we are disassociating */
 if (associated)
  wireless_send_event(priv->net_dev,
 SIOCGIWAP, &wrqu, NULL);

 priv->status &= ~(STATUS_ASSOCIATED | STATUS_ASSOCIATING);
 netif_carrier_off(priv->net_dev);
 netif_stop_queue(priv->net_dev);
}

static void ipw2100_reset_adapter(struct ipw2100_priv *priv)
{
 unsigned long flags;
 union iwreq_data wrqu = {
  .ap_addr = {
       .sa_family = ARPHRD_ETHER}
 };
 int associated = priv->status & STATUS_ASSOCIATED;

 spin_lock_irqsave(&priv->low_lock, flags);
 IPW_DEBUG_INFO(": %s: Restarting adapter.\n", priv->net_dev->name);
 priv->resets++;
 priv->status &= ~(STATUS_ASSOCIATED | STATUS_ASSOCIATING);
```

```
 priv->status |= STATUS_SECURITY_UPDATED;

 /* Force a power cycle even if interface hasn't been opened
  * yet */
 cancel_delayed_work(&priv->reset_work);
 priv->status |= STATUS_RESET_PENDING;
 spin_unlock_irqrestore(&priv->low_lock, flags);

 mutex_lock(&priv->action_mutex);
 /* stop timed checks so that they don't interfere with reset */
 priv->stop_hang_check = 1;
 cancel_delayed_work(&priv->hang_check);

 /* We have to
 signal any supplicant if we are disassociating */
 if (associated)
  wireless_send_event(priv->net_dev, SIOCGIWAP, &wrqu, NULL);

 ipw2100_up(priv, 0);
 mutex_unlock(&priv->action_mutex);

}

static void isr_indicate_associated(struct ipw2100_priv *priv, u32 status)
{

#define MAC_ASSOCIATION_READ_DELAY (HZ)
 int ret, len, essid_len;
 char essid[IW_ESSID_MAX_SIZE];
 u32 txrate;
 u32 chan;
 char *txratename;
 u8 bssid[ETH_ALEN];

 /*
  * TBD: BSSID is usually 00:00:00:00:00:00 here and not
  *      an actual MAC of the AP. Seems like FW sets this
  *      address too late. Read it later and expose through
  *      /proc or schedule a later task to query and update
  */

 essid_len = IW_ESSID_MAX_SIZE;
 ret = ipw2100_get_ordinal(priv, IPW_ORD_STAT_ASSN_SSID,
     essid, &essid_len);
 if (ret) {
  IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
         __LINE__);
  return;
```

```
}

len = sizeof(u32);
ret = ipw2100_get_ordinal(priv, IPW_ORD_CURRENT_TX_RATE, &txrate, &len);
if (ret)
{
 IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
        __LINE__);
 return;
}

len = sizeof(u32);
ret = ipw2100_get_ordinal(priv, IPW_ORD_OUR_FREQ, &chan, &len);
if (ret) {
 IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
        __LINE__);
 return;
}
len = ETH_ALEN;
ipw2100_get_ordinal(priv, IPW_ORD_STAT_ASSN_AP_BSSID, &bssid, &len);
if (ret) {
 IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
        __LINE__);
 return;
}
memcpy(priv->ieee->bssid, bssid, ETH_ALEN);

switch (txrate) {
case TX_RATE_1_MBIT:
 txratename = "1Mbps";
 break;
case TX_RATE_2_MBIT:
 txratename = "2Mbsp";
 break;
case TX_RATE_5_5_MBIT:
 txratename = "5.5Mbps";
 break;
case TX_RATE_11_MBIT:
 txratename = "11Mbps";
 break;
default:
 IPW_DEBUG_INFO("Unknown rate: %d\n", txrate);
 txratename = "unknown rate";
 break;
}

IPW_DEBUG_INFO("%s: Associated with '%s' at %s, channel %d (BSSID="
       MAC_FMT ")\n",
```

```
        priv->net_dev->name, escape_essid(essid,
 essid_len),
        txratename, chan, MAC_ARG(bssid));

 /* now we copy read ssid into dev */
 if (!(priv->config & CFG_STATIC_ESSID)) {
  priv->essid_len = min((u8) essid_len, (u8) IW_ESSID_MAX_SIZE);
  memcpy(priv->essid, essid, priv->essid_len);
 }
 priv->channel = chan;
 memcpy(priv->bssid, bssid, ETH_ALEN);

 priv->status |= STATUS_ASSOCIATING;
 priv->connect_start = get_seconds();

 queue_delayed_work(priv->workqueue, &priv->wx_event_work, HZ / 10);
}

static int ipw2100_set_essid(struct ipw2100_priv *priv, char *essid,
       int length, int batch_mode)
{
 int ssid_len = min(length, IW_ESSID_MAX_SIZE);
 struct host_command cmd = {
  .host_command = SSID,
  .host_command_sequence = 0,
  .host_command_length = ssid_len
 };
 int err;

 IPW_DEBUG_HC("SSID: '%s'\n", escape_essid(essid, ssid_len));

 if (ssid_len)
  memcpy(cmd.host_command_parameters, essid, ssid_len);

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 /* Bug in FW
 currently doesn't honor bit 0 in SET_SCAN_OPTIONS to
  * disable auto association -- so we cheat by setting a bogus SSID */
 if (!ssid_len && !(priv->config & CFG_ASSOCIATE)) {
  int i;
  u8 *bogus = (u8 *) cmd.host_command_parameters;
  for (i = 0; i < IW_ESSID_MAX_SIZE; i++)
   bogus[i] = 0x18 + i;
```

```
 cmd.host_command_length = IW_ESSID_MAX_SIZE;
}

/* NOTE:  We always send the SSID command even if the provided ESSID is
 * the same as what we currently think is set. */

err = ipw2100_hw_send_command(priv, &cmd);
if (!err) {
 memset(priv->essid + ssid_len, 0, IW_ESSID_MAX_SIZE - ssid_len);
 memcpy(priv->essid, essid, ssid_len);
 priv->essid_len = ssid_len;
}

if (!batch_mode) {
 if (ipw2100_enable_adapter(priv))
  err = -EIO;
}

return err;
}

static void isr_indicate_association_lost(struct ipw2100_priv *priv, u32 status)
{
 IPW_DEBUG(IPW_DL_NOTIF | IPW_DL_STATE | IPW_DL_ASSOC,
    "disassociated: '%s' " MAC_FMT " \n",
    escape_essid(priv->essid, priv->essid_len),

 MAC_ARG(priv->bssid));

 priv->status &= ~(STATUS_ASSOCIATED | STATUS_ASSOCIATING);

 if (priv->status & STATUS_STOPPING) {
 IPW_DEBUG_INFO("Card is stopping itself, discard ASSN_LOST.\n");
 return;
 }

 memset(priv->bssid, 0, ETH_ALEN);
 memset(priv->ieee->bssid, 0, ETH_ALEN);

 netif_carrier_off(priv->net_dev);
 netif_stop_queue(priv->net_dev);

 if (!(priv->status & STATUS_RUNNING))
 return;

 if (priv->status & STATUS_SECURITY_UPDATED)
 queue_work(priv->workqueue, &priv->security_work);
```

```
 queue_work(priv->workqueue, &priv->wx_event_work);
}

static void isr_indicate_rf_kill(struct ipw2100_priv *priv, u32 status)
{
 IPW_DEBUG_INFO("%s: RF Kill state changed to radio OFF.\n",
        priv->net_dev->name);

 /* RF_KILL is now enabled (else we wouldn't be here) */
 priv->status |= STATUS_RF_KILL_HW;

#ifdef ACPI_CSTATE_LIMIT_DEFINED
 if (priv->config & CFG_C3_DISABLED) {
  IPW_DEBUG_INFO(": Resetting C3 transitions.\n");
  acpi_set_cstate_limit(priv->cstate_limit);
  priv->config
 &= ~CFG_C3_DISABLED;
 }
#endif

 /* Make sure the RF Kill check timer is running */
 priv->stop_rf_kill = 0;
 cancel_delayed_work(&priv->rf_kill);
 queue_delayed_work(priv->workqueue, &priv->rf_kill, HZ);
}

static void isr_scan_complete(struct ipw2100_priv *priv, u32 status)
{
 IPW_DEBUG_SCAN("scan complete\n");
 /* Age the scan results... */
 priv->ieee->scans++;
 priv->status &= ~STATUS_SCANNING;
}

#ifdef CONFIG_IPW2100_DEBUG
#define IPW2100_HANDLER(v, f) { v, f, # v }
struct ipw2100_status_indicator {
 int status;
 void (*cb) (struct ipw2100_priv * priv, u32 status);
 char *name;
};
#else
#define IPW2100_HANDLER(v, f) { v, f }
struct ipw2100_status_indicator {
 int status;
 void (*cb) (struct ipw2100_priv * priv, u32 status);
};
#endif    /* CONFIG_IPW2100_DEBUG */
```

```
static void isr_indicate_scanning(struct ipw2100_priv *priv, u32 status)
{
 IPW_DEBUG_SCAN("Scanning...\n");
 priv->status |= STATUS_SCANNING;
}

static const struct ipw2100_status_indicator status_handlers[] = {
 IPW2100_HANDLER(IPW_STATE_INITIALIZED,
 NULL),
 IPW2100_HANDLER(IPW_STATE_COUNTRY_FOUND, NULL),
 IPW2100_HANDLER(IPW_STATE_ASSOCIATED, isr_indicate_associated),
 IPW2100_HANDLER(IPW_STATE_ASSN_LOST, isr_indicate_association_lost),
 IPW2100_HANDLER(IPW_STATE_ASSN_CHANGED, NULL),
 IPW2100_HANDLER(IPW_STATE_SCAN_COMPLETE, isr_scan_complete),
 IPW2100_HANDLER(IPW_STATE_ENTERED_PSP, NULL),
 IPW2100_HANDLER(IPW_STATE_LEFT_PSP, NULL),
 IPW2100_HANDLER(IPW_STATE_RF_KILL, isr_indicate_rf_kill),
 IPW2100_HANDLER(IPW_STATE_DISABLED, NULL),
 IPW2100_HANDLER(IPW_STATE_POWER_DOWN, NULL),
 IPW2100_HANDLER(IPW_STATE_SCANNING, isr_indicate_scanning),
 IPW2100_HANDLER(-1, NULL)
};

static void isr_status_change(struct ipw2100_priv *priv, int status)
{
 int i;

 if (status == IPW_STATE_SCANNING &&
    priv->status & STATUS_ASSOCIATED &&
    !(priv->status & STATUS_SCANNING)) {
  IPW_DEBUG_INFO("Scan detected while associated, with "
        "no scan request.  Restarting firmware.\n");

  /* Wake up any sleeping
 jobs */
  schedule_reset(priv);
 }

 for (i = 0; status_handlers[i].status != -1; i++) {
  if (status == status_handlers[i].status) {
   IPW_DEBUG_NOTIF("Status change: %s\n",
     status_handlers[i].name);
   if (status_handlers[i].cb)
    status_handlers[i].cb(priv, status);
   priv->wstats.status = status;
   return;
  }
```

```
 }

 IPW_DEBUG_NOTIF("unknown status received: %04x\n", status);
}

static void isr_rx_complete_command(struct ipw2100_priv *priv,
       struct ipw2100_cmd_header *cmd)
{
#ifdef CONFIG_IPW2100_DEBUG
 if (cmd->host_command_reg < ARRAY_SIZE(command_types)) {
  IPW_DEBUG_HC("Command completed '%s (%d)'\n",
       command_types[cmd->host_command_reg],
       cmd->host_command_reg);
 }
#endif
 if (cmd->host_command_reg == HOST_COMPLETE)
  priv->status |= STATUS_ENABLED;

 if (cmd->host_command_reg == CARD_DISABLE)
  priv->status &= ~STATUS_ENABLED;

 priv->status &= ~STATUS_CMD_ACTIVE;

 wake_up_interruptible(&priv->wait_command_queue);
}

#ifdef CONFIG_IPW2100_DEBUG
static
 const char *frame_types[] = {
 "COMMAND_STATUS_VAL",
 "STATUS_CHANGE_VAL",
 "P80211_DATA_VAL",
 "P8023_DATA_VAL",
 "HOST_NOTIFICATION_VAL"
};
#endif

static int ipw2100_alloc_skb(struct ipw2100_priv *priv,
       struct ipw2100_rx_packet *packet)
{
 packet->skb = dev_alloc_skb(sizeof(struct ipw2100_rx));
 if (!packet->skb)
  return -ENOMEM;

 packet->rxp = (struct ipw2100_rx *)packet->skb->data;
 packet->dma_addr = pci_map_single(priv->pci_dev, packet->skb->data,
      sizeof(struct ipw2100_rx),
      PCI_DMA_FROMDEVICE);
```

```
 /* NOTE: pci_map_single does not return an error code, and 0 is a valid
  *       dma_addr */

 return 0;
}

#define SEARCH_ERROR   0xffffffff
#define SEARCH_FAIL    0xfffffffe
#define SEARCH_SUCCESS 0xfffffff0
#define SEARCH_DISCARD 0
#define SEARCH_SNAPSHOT 1

#define SNAPSHOT_ADDR(ofs) (priv->snapshot[((ofs) >> 12) & 0xff] + ((ofs) & 0xfff))
static void ipw2100_snapshot_free(struct ipw2100_priv *priv)
{
 int i;
 if (!priv->snapshot[0])
  return;
 for
 (i = 0; i < 0x30; i++)
  kfree(priv->snapshot[i]);
 priv->snapshot[0] = NULL;
}

#ifdef CONFIG_IPW2100_DEBUG_C3
static int ipw2100_snapshot_alloc(struct ipw2100_priv *priv)
{
 int i;
 if (priv->snapshot[0])
  return 1;
 for (i = 0; i < 0x30; i++) {
  priv->snapshot[i] = (u8 *) kmalloc(0x1000, GFP_ATOMIC);
  if (!priv->snapshot[i]) {
   IPW_DEBUG_INFO("%s: Error allocating snapshot "
          "buffer %d\n", priv->net_dev->name, i);
   while (i > 0)
    kfree(priv->snapshot[--i]);
   priv->snapshot[0] = NULL;
   return 0;
  }
 }

 return 1;
}

static u32 ipw2100_match_buf(struct ipw2100_priv *priv, u8 * in_buf,
       size_t len, int mode)
{
```

```
 u32 i, j;
 u32 tmp;
 u8 *s, *d;
 u32 ret;

 s = in_buf;
 if (mode == SEARCH_SNAPSHOT) {
  if (!ipw2100_snapshot_alloc(priv))
   mode = SEARCH_DISCARD;
 }

 for (ret = SEARCH_FAIL, i = 0; i < 0x30000; i += 4) {
  read_nic_dword(priv->net_dev, i, &tmp);
  if (mode == SEARCH_SNAPSHOT)
   *(u32 *) SNAPSHOT_ADDR(i) = tmp;
  if (ret == SEARCH_FAIL)
  {
   d = (u8 *) & tmp;
   for (j = 0; j < 4; j++) {
    if (*s != *d) {
     s = in_buf;
     continue;
    }

    s++;
    d++;

    if ((s - in_buf) == len)
     ret = (i + j) - len + 1;
   }
  } else if (mode == SEARCH_DISCARD)
   return ret;
 }

 return ret;
}
#endif

/*
*
* 0) Disconnect the SKB from the firmware (just unmap)
* 1) Pack the ETH header into the SKB
* 2) Pass the SKB to the network stack
*
* When packet is provided by the firmware, it contains the following:
*
* .  ieee80211_hdr
* .  ieee80211_snap_hdr
```

```
*
* The size of the constructed ethernet
*
*/
#ifdef CONFIG_IPW2100_RX_DEBUG
static u8 packet_data[IPW_RX_NIC_BUFFER_LENGTH];
#endif

static void ipw2100_corruption_detected(struct ipw2100_priv *priv, int i)
{
#ifdef CONFIG_IPW2100_DEBUG_C3
 struct ipw2100_status *status = &priv->status_queue.drv[i];
 u32 match, reg;
 int j;
#endif
#ifdef ACPI_CSTATE_LIMIT_DEFINED
 int limit;
#endif

 IPW_DEBUG_INFO(": PCI latency error detected at 0x%04zX.\n",

  i * sizeof(struct ipw2100_status));

#ifdef ACPI_CSTATE_LIMIT_DEFINED
 IPW_DEBUG_INFO(": Disabling C3 transitions.\n");
 limit = acpi_get_cstate_limit();
 if (limit > 2) {
  priv->cstate_limit = limit;
  acpi_set_cstate_limit(2);
  priv->config |= CFG_C3_DISABLED;
 }
#endif

#ifdef CONFIG_IPW2100_DEBUG_C3
 /* Halt the fimrware so we can get a good image */
 write_register(priv->net_dev, IPW_REG_RESET_REG,
        IPW_AUX_HOST_RESET_REG_STOP_MASTER);
 j = 5;
 do {
  udelay(IPW_WAIT_RESET_MASTER_ASSERT_COMPLETE_DELAY);
  read_register(priv->net_dev, IPW_REG_RESET_REG, &reg);

  if (reg & IPW_AUX_HOST_RESET_REG_MASTER_DISABLED)
   break;
 } while (j--);

 match = ipw2100_match_buf(priv, (u8 *) status,
     sizeof(struct ipw2100_status),
```

```
     SEARCH_SNAPSHOT);
 if (match < SEARCH_SUCCESS)
  IPW_DEBUG_INFO("%s: DMA status match in Firmware at "
         "offset 0x%06X, length %d:\n",
         priv->net_dev->name, match,
         sizeof(struct ipw2100_status));
 else
  IPW_DEBUG_INFO("%s:
 No DMA status match in "
         "Firmware.\n", priv->net_dev->name);

 printk_buf((u8 *) priv->status_queue.drv,
     sizeof(struct ipw2100_status) * RX_QUEUE_LENGTH);
#endif

 priv->fatal_error = IPW2100_ERR_C3_CORRUPTION;
 priv->ieee->stats.rx_errors++;
 schedule_reset(priv);
}

static void isr_rx(struct ipw2100_priv *priv, int i,
     struct ieee80211_rx_stats *stats)
{
 struct ipw2100_status *status = &priv->status_queue.drv[i];
 struct ipw2100_rx_packet *packet = &priv->rx_buffers[i];

 IPW_DEBUG_RX("Handler...\n");

 if (unlikely(status->frame_size > skb_tailroom(packet->skb))) {
  IPW_DEBUG_INFO("%s: frame_size (%u) > skb_tailroom (%u)!"
         "  Dropping.\n",
         priv->net_dev->name,
         status->frame_size, skb_tailroom(packet->skb));
  priv->ieee->stats.rx_errors++;
  return;
 }

 if (unlikely(!netif_running(priv->net_dev))) {
  priv->ieee->stats.rx_errors++;
  priv->wstats.discard.misc++;
  IPW_DEBUG_DROP("Dropping packet while interface is not up.\n");
  return;
 }

 if
 (unlikely(priv->ieee->iw_mode != IW_MODE_MONITOR &&
       !(priv->status & STATUS_ASSOCIATED))) {
  IPW_DEBUG_DROP("Dropping packet while not associated.\n");
```

```
 priv->wstats.discard.misc++;
 return;
 }

 pci_unmap_single(priv->pci_dev,
    packet->dma_addr,
    sizeof(struct ipw2100_rx), PCI_DMA_FROMDEVICE);

 skb_put(packet->skb, status->frame_size);

#ifdef CONFIG_IPW2100_RX_DEBUG
 /* Make a copy of the frame so we can dump it to the logs if
  * ieee80211_rx fails */
 memcpy(packet_data, packet->skb->data,
        min_t(u32, status->frame_size, IPW_RX_NIC_BUFFER_LENGTH));
#endif

 if (!ieee80211_rx(priv->ieee, packet->skb, stats)) {
#ifdef CONFIG_IPW2100_RX_DEBUG
  IPW_DEBUG_DROP("%s: Non consumed packet:\n",
         priv->net_dev->name);
  printk_buf(IPW_DL_DROP, packet_data, status->frame_size);
#endif
  priv->ieee->stats.rx_errors++;

  /* ieee80211_rx failed, so it didn't free the SKB */
  dev_kfree_skb_any(packet->skb);
  packet->skb = NULL;
 }

 /* We need
 to allocate a new SKB and attach it to the RDB. */
 if (unlikely(ipw2100_alloc_skb(priv, packet))) {
  printk(KERN_WARNING DRV_NAME ": "
         "%s: Unable to allocate SKB onto RBD ring - disabling "
         "adapter.\n", priv->net_dev->name);
  /* TODO: schedule adapter shutdown */
  IPW_DEBUG_INFO("TODO: Shutdown adapter...\n");
 }

 /* Update the RDB entry */
 priv->rx_queue.drv[i].host_addr = packet->dma_addr;
}

#ifdef CONFIG_IPW2100_MONITOR

static void isr_rx_monitor(struct ipw2100_priv *priv, int i,
     struct ieee80211_rx_stats *stats)
```

```
{
struct ipw2100_status *status = &priv->status_queue.drv[i];
struct ipw2100_rx_packet *packet = &priv->rx_buffers[i];

/* Magic struct that slots into the radiotap header -- no reason
 * to build this manually element by element, we can write it much
 * more efficiently than we can parse it. ORDER MATTERS HERE */
struct ipw_rt_hdr {
 struct ieee80211_radiotap_header rt_hdr;
 s8 rt_dbmsignal; /* signal in dbM, kluged to signed */
}
*ipw_rt;

IPW_DEBUG_RX("Handler...\n");

if (unlikely(status->frame_size > skb_tailroom(packet->skb) -
   sizeof(struct ipw_rt_hdr))) {
 IPW_DEBUG_INFO("%s: frame_size (%u) > skb_tailroom (%u)!"
        "  Dropping.\n",
        priv->net_dev->name,
        status->frame_size,
        skb_tailroom(packet->skb));
 priv->ieee->stats.rx_errors++;
 return;
}

if (unlikely(!netif_running(priv->net_dev))) {
 priv->ieee->stats.rx_errors++;
 priv->wstats.discard.misc++;
 IPW_DEBUG_DROP("Dropping packet while interface is not up.\n");
 return;
}

if (unlikely(priv->config & CFG_CRC_CHECK &&
     status->flags & IPW_STATUS_FLAG_CRC_ERROR)) {
 IPW_DEBUG_RX("CRC error in packet.  Dropping.\n");
 priv->ieee->stats.rx_errors++;
 return;
}

pci_unmap_single(priv->pci_dev, packet->dma_addr,
   sizeof(struct ipw2100_rx), PCI_DMA_FROMDEVICE);
memmove(packet->skb->data + sizeof(struct ipw_rt_hdr),
 packet->skb->data, status->frame_size);

ipw_rt = (struct ipw_rt_hdr *)
packet->skb->data;
```

```
 ipw_rt->rt_hdr.it_version = PKTHDR_RADIOTAP_VERSION;
 ipw_rt->rt_hdr.it_pad = 0; /* always good to zero */
 ipw_rt->rt_hdr.it_len = sizeof(struct ipw_rt_hdr); /* total hdr+data */

 ipw_rt->rt_hdr.it_present = 1 << IEEE80211_RADIOTAP_DBM_ANTSIGNAL;

 ipw_rt->rt_dbmsignal = status->rssi + IPW2100_RSSI_TO_DBM;

 skb_put(packet->skb, status->frame_size + sizeof(struct ipw_rt_hdr));

 if (!ieee80211_rx(priv->ieee, packet->skb, stats)) {
  priv->ieee->stats.rx_errors++;

  /* ieee80211_rx failed, so it didn't free the SKB */
  dev_kfree_skb_any(packet->skb);
  packet->skb = NULL;
 }

 /* We need to allocate a new SKB and attach it to the RDB. */
 if (unlikely(ipw2100_alloc_skb(priv, packet))) {
  IPW_DEBUG_WARNING(
   "%s: Unable to allocate SKB onto RBD ring - disabling "
   "adapter.\n", priv->net_dev->name);
  /* TODO: schedule adapter shutdown */
  IPW_DEBUG_INFO("TODO: Shutdown adapter...\n");
 }

 /* Update the RDB entry */
 priv->rx_queue.drv[i].host_addr = packet->dma_addr;
}

#endif

static
 int ipw2100_corruption_check(struct ipw2100_priv *priv, int i)
{
 struct ipw2100_status *status = &priv->status_queue.drv[i];
 struct ipw2100_rx *u = priv->rx_buffers[i].rxp;
 u16 frame_type = status->status_fields & STATUS_TYPE_MASK;

 switch (frame_type) {
 case COMMAND_STATUS_VAL:
  return (status->frame_size != sizeof(u->rx_data.command));
 case STATUS_CHANGE_VAL:
  return (status->frame_size != sizeof(u->rx_data.status));
 case HOST_NOTIFICATION_VAL:
  return (status->frame_size < sizeof(u->rx_data.notification));
 case P80211_DATA_VAL:
```

```
 case P8023_DATA_VAL:
#ifdef CONFIG_IPW2100_MONITOR
  return 0;
#else
  switch (WLAN_FC_GET_TYPE(u->rx_data.header.frame_ctl)) {
  case IEEE80211_FTYPE_MGMT:
  case IEEE80211_FTYPE_CTL:
   return 0;
  case IEEE80211_FTYPE_DATA:
   return (status->frame_size >
    IPW_MAX_802_11_PAYLOAD_LENGTH);
  }
#endif
 }

 return 1;
}

/*
* ipw2100 interrupts are disabled at this point, and the ISR
* is the only code that calls
 this method.  So, we do not need
* to play with any locks.
*
* RX Queue works as follows:
*
* Read index - firmware places packet in entry identified by the
*              Read index and advances Read index.  In this manner,
*              Read index will always point to the next packet to
*              be filled--but not yet valid.
*
* Write index - driver fills this entry with an unused RBD entry.
*               This entry has not filled by the firmware yet.
*
* In between the W and R indexes are the RBDs that have been received
* but not yet processed.
*
* The process of handling packets will start at WRITE + 1 and advance
* until it reaches the READ index.
*
* The WRITE index is cached in the variable 'priv->rx_queue.next'.
*
*/
static void __ipw2100_rx_process(struct ipw2100_priv *priv)
{
 struct ipw2100_bd_queue *rxq = &priv->rx_queue;
 struct ipw2100_status_queue *sq = &priv->status_queue;
 struct ipw2100_rx_packet *packet;
```

```
u16 frame_type;
u32 r, w, i, s;
struct
ipw2100_rx *u;
struct ieee80211_rx_stats stats = {
 .mac_time = jiffies,
};

read_register(priv->net_dev, IPW_MEM_HOST_SHARED_RX_READ_INDEX, &r);
read_register(priv->net_dev, IPW_MEM_HOST_SHARED_RX_WRITE_INDEX, &w);

if (r >= rxq->entries) {
 IPW_DEBUG_RX("exit - bad read index\n");
 return;
}

i = (rxq->next + 1) % rxq->entries;
s = i;
while (i != r) {
 /* IPW_DEBUG_RX("r = %d : w = %d : processing = %d\n",
    r, rxq->next, i); */

 packet = &priv->rx_buffers[i];

 /* Sync the DMA for the STATUS buffer so CPU is sure to get
  * the correct values */
 pci_dma_sync_single_for_cpu(priv->pci_dev,
       sq->nic +
       sizeof(struct ipw2100_status) * i,
       sizeof(struct ipw2100_status),
       PCI_DMA_FROMDEVICE);

 /* Sync the DMA for the RX buffer so CPU is sure to get
  * the correct values */
 pci_dma_sync_single_for_cpu(priv->pci_dev, packet->dma_addr,
       sizeof(struct ipw2100_rx),
       PCI_DMA_FROMDEVICE);

 if (unlikely(ipw2100_corruption_check(priv,
i))) {
  ipw2100_corruption_detected(priv, i);
  goto increment;
 }

 u = packet->rxp;
 frame_type = sq->drv[i].status_fields & STATUS_TYPE_MASK;
 stats.rssi = sq->drv[i].rssi + IPW2100_RSSI_TO_DBM;
 stats.len = sq->drv[i].frame_size;
```

```
  stats.mask = 0;
  if (stats.rssi != 0)
   stats.mask |= IEEE80211_STATMASK_RSSI;
  stats.freq = IEEE80211_24GHZ_BAND;

  IPW_DEBUG_RX("%s: '%s' frame type received (%d).\n",
        priv->net_dev->name, frame_types[frame_type],
        stats.len);

  switch (frame_type) {
  case COMMAND_STATUS_VAL:
   /* Reset Rx watchdog */
   isr_rx_complete_command(priv, &u->rx_data.command);
   break;

  case STATUS_CHANGE_VAL:
   isr_status_change(priv, u->rx_data.status);
   break;

  case P80211_DATA_VAL:
  case P8023_DATA_VAL:
#ifdef CONFIG_IPW2100_MONITOR
    if (priv->ieee->iw_mode == IW_MODE_MONITOR) {
     isr_rx_monitor(priv, i, &stats);
     break;
    }
#endif
    if (stats.len < sizeof(u->rx_data.header))
     break;
    switch
   (WLAN_FC_GET_TYPE(u->rx_data.header.frame_ctl)) {
    case IEEE80211_FTYPE_MGMT:
     ieee80211_rx_mgt(priv->ieee,
        &u->rx_data.header, &stats);
     break;

    case IEEE80211_FTYPE_CTL:
     break;

    case IEEE80211_FTYPE_DATA:
     isr_rx(priv, i, &stats);
     break;

    }
    break;
  }
```

```
      increment:
  /* clear status field associated with this RBD */
  rxq->drv[i].status.info.field = 0;

  i = (i + 1) % rxq->entries;
 }

 if (i != s) {
  /* backtrack one entry, wrapping to end if at 0 */
  rxq->next = (i ? i : rxq->entries) - 1;

  write_register(priv->net_dev,
         IPW_MEM_HOST_SHARED_RX_WRITE_INDEX, rxq->next);
 }
}

/*
* __ipw2100_tx_process
*
* This routine will determine whether the next packet on
* the fw_pend_list has been processed by the firmware yet.
*
* If not, then it does nothing and returns.
*
* If so, then it removes the item from the fw_pend_list, frees
* any associated storage, and places the item back on the
* free list of its source (either msg_free_list
 or tx_free_list)
*
* TX Queue works as follows:
*
* Read index - points to the next TBD that the firmware will
*              process.  The firmware will read the data, and once
*              done processing, it will advance the Read index.
*
* Write index - driver fills this entry with an constructed TBD
*               entry.  The Write index is not advanced until the
*               packet has been configured.
*
* In between the W and R indexes are the TBDs that have NOT been
* processed.  Lagging behind the R index are packets that have
* been processed but have not been freed by the driver.
*
* In order to free old storage, an internal index will be maintained
* that points to the next packet to be freed.  When all used
* packets have been freed, the oldest index will be the same as the
* firmware's read index.
*
```

```
* The OLDEST index is cached in the variable 'priv->tx_queue.oldest'
*
* Because the TBD structure can not contain arbitrary data, the
 * driver must keep an internal queue of cached allocations such that
* it can put that data back into the tx_free_list and msg_free_list
* for use by future command and data packets.
*
*/
static int __ipw2100_tx_process(struct ipw2100_priv *priv)
{
 struct ipw2100_bd_queue *txq = &priv->tx_queue;
 struct ipw2100_bd *tbd;
 struct list_head *element;
 struct ipw2100_tx_packet *packet;
 int descriptors_used;
 int e, i;
 u32 r, w, frag_num = 0;

 if (list_empty(&priv->fw_pend_list))
  return 0;

 element = priv->fw_pend_list.next;

 packet = list_entry(element, struct ipw2100_tx_packet, list);
 tbd = &txq->drv[packet->index];

 /* Determine how many TBD entries must be finished... */
 switch (packet->type) {
 case COMMAND:
  /* COMMAND uses only one slot; don't advance */
  descriptors_used = 1;
  e = txq->oldest;
  break;

 case DATA:
  /* DATA uses two slots; advance and loop position. */
  descriptors_used = tbd->num_fragments;
  frag_num = tbd->num_fragments - 1;
  e = txq->oldest
 + frag_num;
  e %= txq->entries;
  break;

 default:
  printk(KERN_WARNING DRV_NAME ": %s: Bad fw_pend_list entry!\n",
         priv->net_dev->name);
  return 0;
 }
```

```
/* if the last TBD is not done by NIC yet, then packet is
 * not ready to be released.
 *
 */
read_register(priv->net_dev, IPW_MEM_HOST_SHARED_TX_QUEUE_READ_INDEX,
      &r);
read_register(priv->net_dev, IPW_MEM_HOST_SHARED_TX_QUEUE_WRITE_INDEX,
      &w);
if (w != txq->next)
 printk(KERN_WARNING DRV_NAME ": %s: write index mismatch\n",
       priv->net_dev->name);

/*
 * txq->next is the index of the last packet written txq->oldest is
 * the index of the r is the index of the next packet to be read by
 * firmware
 */

/*
 * Quick graphic to help you visualize the following
 * if / else statement
 *
 * ===>|                  s---->|===============
 *                           e>|
 * | a | b | c | d | e | f | g | h | i | j | k | l
 *       r---->|
 *              w
 *
 * w - updated
by driver
 * r - updated by firmware
 * s - start of oldest BD entry (txq->oldest)
 * e - end of oldest BD entry
 *
 */
if (!((r <= w && (e < r || e >= w)) || (e < r && e >= w))) {
 IPW_DEBUG_TX("exit - no processed packets ready to release.\n");
 return 0;
}

list_del(element);
DEC_STAT(&priv->fw_pend_stat);

#ifdef CONFIG_IPW2100_DEBUG
 {
 int i = txq->oldest;
 IPW_DEBUG_TX("TX%d V=%p P=%04X T=%04X L=%d\n", i,
```

```
        &txq->drv[i],
        (u32) (txq->nic + i * sizeof(struct ipw2100_bd)),
        txq->drv[i].host_addr, txq->drv[i].buf_length);

  if (packet->type == DATA) {
   i = (i + 1) % txq->entries;

   IPW_DEBUG_TX("TX%d V=%p P=%04X T=%04X L=%d\n", i,
        &txq->drv[i],
        (u32) (txq->nic + i *
        sizeof(struct ipw2100_bd)),
        (u32) txq->drv[i].host_addr,
        txq->drv[i].buf_length);
  }
 }
#endif

 switch (packet->type) {
 case DATA:
  if (txq->drv[txq->oldest].status.info.fields.txType != 0)
   printk(KERN_WARNING DRV_NAME ": %s: Queue
 mismatch.  "
         "Expecting DATA TBD but pulled "
         "something else: ids %d=%d.\n",
         priv->net_dev->name, txq->oldest, packet->index);

  /* DATA packet; we have to unmap and free the SKB */
  for (i = 0; i < frag_num; i++) {
   tbd = &txq->drv[(packet->index + 1 + i) % txq->entries];

   IPW_DEBUG_TX("TX%d P=%08x L=%d\n",
        (packet->index + 1 + i) % txq->entries,
        tbd->host_addr, tbd->buf_length);

   pci_unmap_single(priv->pci_dev,
     tbd->host_addr,
     tbd->buf_length, PCI_DMA_TODEVICE);
  }

  ieee80211_txb_free(packet->info.d_struct.txb);
  packet->info.d_struct.txb = NULL;

  list_add_tail(element, &priv->tx_free_list);
  INC_STAT(&priv->tx_free_stat);

  /* We have a free slot in the Tx queue, so wake up the
   * transmit layer if it is stopped. */
  if (priv->status & STATUS_ASSOCIATED)
```

```
  netif_wake_queue(priv->net_dev);

  /* A packet was processed by the hardware, so update the
   * watchdog */
  priv->net_dev->trans_start = jiffies;

  break;

 case
 COMMAND:
  if (txq->drv[txq->oldest].status.info.fields.txType != 1)
   printk(KERN_WARNING DRV_NAME ": %s: Queue mismatch.  "
          "Expecting COMMAND TBD but pulled "
          "something else: ids %d=%d.\n",
          priv->net_dev->name, txq->oldest, packet->index);

#ifdef CONFIG_IPW2100_DEBUG
  if (packet->info.c_struct.cmd->host_command_reg <
      sizeof(command_types) / sizeof(*command_types))
   IPW_DEBUG_TX("Command '%s (%d)' processed: %d.\n",
          command_types[packet->info.c_struct.cmd->
          host_command_reg],
          packet->info.c_struct.cmd->
          host_command_reg,
          packet->info.c_struct.cmd->cmd_status_reg);
#endif

  list_add_tail(element, &priv->msg_free_list);
  INC_STAT(&priv->msg_free_stat);
  break;
  }

  /* advance oldest used TBD pointer to start of next entry */
  txq->oldest = (e + 1) % txq->entries;
  /* increase available TBDs number */
  txq->available += descriptors_used;
  SET_STAT(&priv->txq_stat, txq->available);

  IPW_DEBUG_TX("packet
  latency (send to process)  %ld jiffies\n",
       jiffies - packet->jiffy_start);

  return (!list_empty(&priv->fw_pend_list));
 }

 static inline void __ipw2100_tx_complete(struct ipw2100_priv *priv)
 {
  int i = 0;
```

```
 while (__ipw2100_tx_process(priv) && i < 200)
  i++;

 if (i == 200) {
  printk(KERN_WARNING DRV_NAME ": "
         "%s: Driver is running slow (%d iters).\n",
         priv->net_dev->name, i);
 }
}

static void ipw2100_tx_send_commands(struct ipw2100_priv *priv)
{
 struct list_head *element;
 struct ipw2100_tx_packet *packet;
 struct ipw2100_bd_queue *txq = &priv->tx_queue;
 struct ipw2100_bd *tbd;
 int next = txq->next;

 while (!list_empty(&priv->msg_pend_list)) {
  /* if there isn't enough space in TBD queue, then
   * don't stuff a new one in.
   * NOTE: 3 are needed as a command will take one,
   *       and there is a minimum of 2 that must be
   *       maintained between the r and w indexes
   */
  if (txq->available <= 3) {
   IPW_DEBUG_TX("no
room in tx_queue\n");
   break;
  }

  element = priv->msg_pend_list.next;
  list_del(element);
  DEC_STAT(&priv->msg_pend_stat);

  packet = list_entry(element, struct ipw2100_tx_packet, list);

  IPW_DEBUG_TX("using TBD at virt=%p, phys=%p\n",
       &txq->drv[txq->next],
       (void *)(txq->nic + txq->next *
         sizeof(struct ipw2100_bd)));

  packet->index = txq->next;

  tbd = &txq->drv[txq->next];

  /* initialize TBD */
```

```
 tbd->host_addr = packet->info.c_struct.cmd_phys;
 tbd->buf_length = sizeof(struct ipw2100_cmd_header);
 /* not marking number of fragments causes problems
  * with f/w debug version */
 tbd->num_fragments = 1;
 tbd->status.info.field =
    IPW_BD_STATUS_TX_FRAME_COMMAND |
    IPW_BD_STATUS_TX_INTERRUPT_ENABLE;

 /* update TBD queue counters */
 txq->next++;
 txq->next %= txq->entries;
 txq->available--;
 DEC_STAT(&priv->txq_stat);

 list_add_tail(element, &priv->fw_pend_list);
 INC_STAT(&priv->fw_pend_stat);
 }

 if (txq->next != next)
 {
  /* kick off the DMA by notifying firmware the
   * write index has moved; make sure TBD stores are sync'd */
  wmb();
  write_register(priv->net_dev,
        IPW_MEM_HOST_SHARED_TX_QUEUE_WRITE_INDEX,
        txq->next);
 }
}

/*
* ipw2100_tx_send_data
*
*/
static void ipw2100_tx_send_data(struct ipw2100_priv *priv)
{
 struct list_head *element;
 struct ipw2100_tx_packet *packet;
 struct ipw2100_bd_queue *txq = &priv->tx_queue;
 struct ipw2100_bd *tbd;
 int next = txq->next;
 int i = 0;
 struct ipw2100_data_header *ipw_hdr;
 struct ieee80211_hdr_3addr *hdr;

 while (!list_empty(&priv->tx_pend_list)) {
  /* if there isn't enough space in TBD queue, then
   * don't stuff a new one in.
```

```
 * NOTE: 4 are needed as a data will take two,
 *       and there is a minimum of 2 that must be
 *       maintained between the r and w indexes
 */
element = priv->tx_pend_list.next;
packet = list_entry(element, struct ipw2100_tx_packet, list);

if (unlikely(1 + packet->info.d_struct.txb->nr_frags
>
     IPW_MAX_BDS)) {
 /* TODO: Support merging buffers if more than
  * IPW_MAX_BDS are used */
 IPW_DEBUG_INFO("%s: Maximum BD theshold exceeded.  "
        "Increase fragmentation level.\n",
        priv->net_dev->name);
}

if (txq->available <= 3 + packet->info.d_struct.txb->nr_frags) {
 IPW_DEBUG_TX("no room in tx_queue\n");
 break;
}

list_del(element);
DEC_STAT(&priv->tx_pend_stat);

tbd = &txq->drv[txq->next];

packet->index = txq->next;

ipw_hdr = packet->info.d_struct.data;
hdr = (struct ieee80211_hdr_3addr *)packet->info.d_struct.txb->
   fragments[0]->data;

if (priv->ieee->iw_mode == IW_MODE_INFRA) {
 /* To DS: Addr1 = BSSID, Addr2 = SA,
    Addr3 = DA */
 memcpy(ipw_hdr->src_addr, hdr->addr2, ETH_ALEN);
 memcpy(ipw_hdr->dst_addr, hdr->addr3, ETH_ALEN);
} else if (priv->ieee->iw_mode == IW_MODE_ADHOC) {
 /* not From/To DS: Addr1 = DA, Addr2 = SA,
    Addr3 = BSSID */
 memcpy(ipw_hdr->src_addr,
hdr->addr2, ETH_ALEN);
 memcpy(ipw_hdr->dst_addr, hdr->addr1, ETH_ALEN);
}

ipw_hdr->host_command_reg = SEND;
ipw_hdr->host_command_reg1 = 0;
```

```
  /* For now we only support host based encryption */
  ipw_hdr->needs_encryption = 0;
  ipw_hdr->encrypted = packet->info.d_struct.txb->encrypted;
  if (packet->info.d_struct.txb->nr_frags > 1)
   ipw_hdr->fragment_size =
       packet->info.d_struct.txb->frag_size -
       IEEE80211_3ADDR_LEN;
  else
   ipw_hdr->fragment_size = 0;

  tbd->host_addr = packet->info.d_struct.data_phys;
  tbd->buf_length = sizeof(struct ipw2100_data_header);
  tbd->num_fragments = 1 + packet->info.d_struct.txb->nr_frags;
  tbd->status.info.field =
      IPW_BD_STATUS_TX_FRAME_802_3 |
      IPW_BD_STATUS_TX_FRAME_NOT_LAST_FRAGMENT;
  txq->next++;
  txq->next %= txq->entries;

  IPW_DEBUG_TX("data header tbd TX%d P=%08x L=%d\n",
         packet->index, tbd->host_addr, tbd->buf_length);
#ifdef CONFIG_IPW2100_DEBUG
  if (packet->info.d_struct.txb->nr_frags > 1)
   IPW_DEBUG_FRAG("fragment
 Tx: %d frames\n",
           packet->info.d_struct.txb->nr_frags);
#endif

  for (i = 0; i < packet->info.d_struct.txb->nr_frags; i++) {
   tbd = &txq->drv[txq->next];
   if (i == packet->info.d_struct.txb->nr_frags - 1)
    tbd->status.info.field =
        IPW_BD_STATUS_TX_FRAME_802_3 |
        IPW_BD_STATUS_TX_INTERRUPT_ENABLE;
   else
    tbd->status.info.field =
        IPW_BD_STATUS_TX_FRAME_802_3 |
        IPW_BD_STATUS_TX_FRAME_NOT_LAST_FRAGMENT;

   tbd->buf_length = packet->info.d_struct.txb->
       fragments[i]->len - IEEE80211_3ADDR_LEN;

   tbd->host_addr = pci_map_single(priv->pci_dev,
       packet->info.d_struct.
       txb->fragments[i]->
       data +
       IEEE80211_3ADDR_LEN,
```

```
      tbd->buf_length,
      PCI_DMA_TODEVICE);

  IPW_DEBUG_TX("data frag tbd TX%d P=%08x L=%d\n",
        txq->next, tbd->host_addr,
        tbd->buf_length);

  pci_dma_sync_single_for_device(priv->pci_dev,
           tbd->host_addr,
           tbd->buf_length,

      PCI_DMA_TODEVICE);

  txq->next++;
  txq->next %= txq->entries;
 }

 txq->available -= 1 + packet->info.d_struct.txb->nr_frags;
 SET_STAT(&priv->txq_stat, txq->available);

 list_add_tail(element, &priv->fw_pend_list);
 INC_STAT(&priv->fw_pend_stat);
}

if (txq->next != next) {
 /* kick off the DMA by notifying firmware the
  * write index has moved; make sure TBD stores are sync'd */
 write_register(priv->net_dev,
        IPW_MEM_HOST_SHARED_TX_QUEUE_WRITE_INDEX,
        txq->next);
}
return;
}

static void ipw2100_irq_tasklet(struct ipw2100_priv *priv)
{
 struct net_device *dev = priv->net_dev;
 unsigned long flags;
 u32 inta, tmp;

 spin_lock_irqsave(&priv->low_lock, flags);
 ipw2100_disable_interrupts(priv);

 read_register(dev, IPW_REG_INTA, &inta);

 IPW_DEBUG_ISR("enter - INTA: 0x%08lX\n",
       (unsigned long)inta & IPW_INTERRUPT_MASK);
```

```
priv->in_isr++;
priv->interrupts++;

/* We do not loop and keep polling for more interrupts as this

* is frowned upon and doesn't play nicely with other potentially
 * chained IRQs */
IPW_DEBUG_ISR("INTA: 0x%08lX\n",
      (unsigned long)inta & IPW_INTERRUPT_MASK);

if (inta & IPW2100_INTA_FATAL_ERROR) {
 printk(KERN_WARNING DRV_NAME
       ": Fatal interrupt. Scheduling firmware restart.\n");
 priv->inta_other++;
 write_register(dev, IPW_REG_INTA, IPW2100_INTA_FATAL_ERROR);

 read_nic_dword(dev, IPW_NIC_FATAL_ERROR, &priv->fatal_error);
 IPW_DEBUG_INFO("%s: Fatal error value: 0x%08X\n",
       priv->net_dev->name, priv->fatal_error);

 read_nic_dword(dev, IPW_ERROR_ADDR(priv->fatal_error), &tmp);
 IPW_DEBUG_INFO("%s: Fatal error address value: 0x%08X\n",
       priv->net_dev->name, tmp);

 /* Wake up any sleeping jobs */
 schedule_reset(priv);
}

if (inta & IPW2100_INTA_PARITY_ERROR) {
 printk(KERN_ERR DRV_NAME
       ": ***** PARITY ERROR INTERRUPT !!!! \n");
 priv->inta_other++;
 write_register(dev, IPW_REG_INTA, IPW2100_INTA_PARITY_ERROR);
}

if
(inta & IPW2100_INTA_RX_TRANSFER) {
 IPW_DEBUG_ISR("RX interrupt\n");

 priv->rx_interrupts++;

 write_register(dev, IPW_REG_INTA, IPW2100_INTA_RX_TRANSFER);

 __ipw2100_rx_process(priv);
 __ipw2100_tx_complete(priv);
}

if (inta & IPW2100_INTA_TX_TRANSFER) {
```

```
 IPW_DEBUG_ISR("TX interrupt\n");

 priv->tx_interrupts++;

 write_register(dev, IPW_REG_INTA, IPW2100_INTA_TX_TRANSFER);

 __ipw2100_tx_complete(priv);
 ipw2100_tx_send_commands(priv);
 ipw2100_tx_send_data(priv);
}

if (inta & IPW2100_INTA_TX_COMPLETE) {
 IPW_DEBUG_ISR("TX complete\n");
 priv->inta_other++;
 write_register(dev, IPW_REG_INTA, IPW2100_INTA_TX_COMPLETE);

 __ipw2100_tx_complete(priv);
}

if (inta & IPW2100_INTA_EVENT_INTERRUPT) {
 /* ipw2100_handle_event(dev); */
 priv->inta_other++;
 write_register(dev, IPW_REG_INTA, IPW2100_INTA_EVENT_INTERRUPT);
}

if (inta & IPW2100_INTA_FW_INIT_DONE) {
 IPW_DEBUG_ISR("FW init done interrupt\n");
 priv->inta_other++;

 read_register(dev, IPW_REG_INTA,
&tmp);
 if (tmp & (IPW2100_INTA_FATAL_ERROR |
    IPW2100_INTA_PARITY_ERROR)) {
  write_register(dev, IPW_REG_INTA,
         IPW2100_INTA_FATAL_ERROR |
         IPW2100_INTA_PARITY_ERROR);
 }

 write_register(dev, IPW_REG_INTA, IPW2100_INTA_FW_INIT_DONE);
}

if (inta & IPW2100_INTA_STATUS_CHANGE) {
 IPW_DEBUG_ISR("Status change interrupt\n");
 priv->inta_other++;
 write_register(dev, IPW_REG_INTA, IPW2100_INTA_STATUS_CHANGE);
}

if (inta & IPW2100_INTA_SLAVE_MODE_HOST_COMMAND_DONE) {
```

```
 IPW_DEBUG_ISR("slave host mode interrupt\n");
 priv->inta_other++;
 write_register(dev, IPW_REG_INTA,
        IPW2100_INTA_SLAVE_MODE_HOST_COMMAND_DONE);
 }

 priv->in_isr--;
 ipw2100_enable_interrupts(priv);

 spin_unlock_irqrestore(&priv->low_lock, flags);

 IPW_DEBUG_ISR("exit\n");
}

static irqreturn_t ipw2100_interrupt(int irq, void *data)
{
 struct ipw2100_priv *priv = data;
 u32 inta, inta_mask;

 if (!data)
  return IRQ_NONE;

 spin_lock(&priv->low_lock);

 /* We check
 to see if we should be ignoring interrupts before
  * we touch the hardware.  During ucode load if we try and handle
  * an interrupt we can cause keyboard problems as well as cause
  * the ucode to fail to initialize */
 if (!(priv->status & STATUS_INT_ENABLED)) {
  /* Shared IRQ */
  goto none;
 }

 read_register(priv->net_dev, IPW_REG_INTA_MASK, &inta_mask);
 read_register(priv->net_dev, IPW_REG_INTA, &inta);

 if (inta == 0xFFFFFFFF) {
  /* Hardware disappeared */
  printk(KERN_WARNING DRV_NAME ": IRQ INTA == 0xFFFFFFFF\n");
  goto none;
 }

 inta &= IPW_INTERRUPT_MASK;

 if (!(inta & inta_mask)) {
  /* Shared interrupt */
  goto none;
```

```
 }

 /* We disable the hardware interrupt here just to prevent unneeded
  * calls to be made.  We disable this again within the actual
  * work tasklet, so if another part of the code re-enables the
  * interrupt, that is fine */
 ipw2100_disable_interrupts(priv);

 tasklet_schedule(&priv->irq_tasklet);
 spin_unlock(&priv->low_lock);

 return IRQ_HANDLED;
     none:
 spin_unlock(&priv->low_lock);
 return IRQ_NONE;
}

static int ipw2100_tx(struct ieee80211_txb *txb, struct net_device *dev,
       int pri)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct list_head *element;
 struct ipw2100_tx_packet *packet;
 unsigned long flags;

 spin_lock_irqsave(&priv->low_lock, flags);

 if (!(priv->status & STATUS_ASSOCIATED)) {
  IPW_DEBUG_INFO("Can not transmit when not connected.\n");
  priv->ieee->stats.tx_carrier_errors++;
  netif_stop_queue(dev);
  goto fail_unlock;
 }

 if (list_empty(&priv->tx_free_list))
  goto fail_unlock;

 element = priv->tx_free_list.next;
 packet = list_entry(element, struct ipw2100_tx_packet, list);

 packet->info.d_struct.txb = txb;

 IPW_DEBUG_TX("Sending fragment (%d bytes):\n", txb->fragments[0]->len);
 printk_buf(IPW_DL_TX, txb->fragments[0]->data, txb->fragments[0]->len);

 packet->jiffy_start = jiffies;

 list_del(element);
```

```
 DEC_STAT(&priv->tx_free_stat);

 list_add_tail(element, &priv->tx_pend_list);
 INC_STAT(&priv->tx_pend_stat);

 ipw2100_tx_send_data(priv);

 spin_unlock_irqrestore(&priv->low_lock,
 flags);
 return 0;

    fail_unlock:
 netif_stop_queue(dev);
 spin_unlock_irqrestore(&priv->low_lock, flags);
 return 1;
}

static int ipw2100_msg_allocate(struct ipw2100_priv *priv)
{
 int i, j, err = -EINVAL;
 void *v;
 dma_addr_t p;

 priv->msg_buffers =
    (struct ipw2100_tx_packet *)kmalloc(IPW_COMMAND_POOL_SIZE *
     sizeof(struct
            ipw2100_tx_packet),
     GFP_KERNEL);
 if (!priv->msg_buffers) {
  printk(KERN_ERR DRV_NAME ": %s: PCI alloc failed for msg "
         "buffers.\n", priv->net_dev->name);
  return -ENOMEM;
 }

 for (i = 0; i < IPW_COMMAND_POOL_SIZE; i++) {
  v = pci_alloc_consistent(priv->pci_dev,
      sizeof(struct ipw2100_cmd_header), &p);
  if (!v) {
   printk(KERN_ERR DRV_NAME ": "
          "%s: PCI alloc failed for msg "
          "buffers.\n", priv->net_dev->name);
   err = -ENOMEM;
   break;
  }

  memset(v, 0, sizeof(struct ipw2100_cmd_header));

  priv->msg_buffers[i].type
```

```
= COMMAND;
 priv->msg_buffers[i].info.c_struct.cmd =
     (struct ipw2100_cmd_header *)v;
 priv->msg_buffers[i].info.c_struct.cmd_phys = p;
 }

 if (i == IPW_COMMAND_POOL_SIZE)
  return 0;

 for (j = 0; j < i; j++) {
  pci_free_consistent(priv->pci_dev,
        sizeof(struct ipw2100_cmd_header),
        priv->msg_buffers[j].info.c_struct.cmd,
        priv->msg_buffers[j].info.c_struct.
        cmd_phys);
 }

 kfree(priv->msg_buffers);
 priv->msg_buffers = NULL;

 return err;
}

static int ipw2100_msg_initialize(struct ipw2100_priv *priv)
{
 int i;

 INIT_LIST_HEAD(&priv->msg_free_list);
 INIT_LIST_HEAD(&priv->msg_pend_list);

 for (i = 0; i < IPW_COMMAND_POOL_SIZE; i++)
  list_add_tail(&priv->msg_buffers[i].list, &priv->msg_free_list);
 SET_STAT(&priv->msg_free_stat, i);

 return 0;
}

static void ipw2100_msg_free(struct ipw2100_priv *priv)
{
 int i;

 if (!priv->msg_buffers)
  return;

 for (i = 0; i < IPW_COMMAND_POOL_SIZE; i++) {
  pci_free_consistent(priv->pci_dev,
```

```
    sizeof(struct ipw2100_cmd_header),
        priv->msg_buffers[i].info.c_struct.cmd,
        priv->msg_buffers[i].info.c_struct.
        cmd_phys);
 }

 kfree(priv->msg_buffers);
 priv->msg_buffers = NULL;
}

static ssize_t show_pci(struct device *d, struct device_attribute *attr,
   char *buf)
{
 struct pci_dev *pci_dev = container_of(d, struct pci_dev, dev);
 char *out = buf;
 int i, j;
 u32 val;

 for (i = 0; i < 16; i++) {
  out += sprintf(out, "[%08X] ", i * 16);
  for (j = 0; j < 16; j += 4) {
   pci_read_config_dword(pci_dev, i * 16 + j, &val);
   out += sprintf(out, "%08X ", val);
  }
  out += sprintf(out, "\n");
 }

 return out - buf;
}

static DEVICE_ATTR(pci, S_IRUGO, show_pci, NULL);

static ssize_t show_cfg(struct device *d, struct device_attribute *attr,
   char *buf)
{
 struct ipw2100_priv *p = d->driver_data;
 return sprintf(buf, "0x%08x\n", (int)p->config);
}

static DEVICE_ATTR(cfg, S_IRUGO, show_cfg, NULL);

static ssize_t show_status(struct
 device *d, struct device_attribute *attr,
      char *buf)
{
 struct ipw2100_priv *p = d->driver_data;
 return sprintf(buf, "0x%08x\n", (int)p->status);
}
```

```
static DEVICE_ATTR(status, S_IRUGO, show_status, NULL);

static ssize_t show_capability(struct device *d, struct device_attribute *attr,
        char *buf)
{
 struct ipw2100_priv *p = d->driver_data;
 return sprintf(buf, "0x%08x\n", (int)p->capability);
}

static DEVICE_ATTR(capability, S_IRUGO, show_capability, NULL);

#define IPW2100_REG(x) { IPW_ ##x, #x }
static const struct {
 u32 addr;
 const char *name;
} hw_data[] = {
IPW2100_REG(REG_GP_CNTRL),
    IPW2100_REG(REG_GPIO),
    IPW2100_REG(REG_INTA),
    IPW2100_REG(REG_INTA_MASK), IPW2100_REG(REG_RESET_REG),};
#define IPW2100_NIC(x, s) { x, #x, s }
static const struct {
 u32 addr;
 const char *name;
 size_t size;
} nic_data[] = {
IPW2100_NIC(IPW2100_CONTROL_REG, 2),
    IPW2100_NIC(0x210014, 1), IPW2100_NIC(0x210000, 1),};
#define IPW2100_ORD(x, d) { IPW_ORD_
 ##x, #x, d }
static const struct {
 u8 index;
 const char *name;
 const char *desc;
} ord_data[] = {
IPW2100_ORD(STAT_TX_HOST_REQUESTS, "requested Host Tx's (MSDU)"),
    IPW2100_ORD(STAT_TX_HOST_COMPLETE,
   "successful Host Tx's (MSDU)"),
    IPW2100_ORD(STAT_TX_DIR_DATA,
   "successful Directed Tx's (MSDU)"),
    IPW2100_ORD(STAT_TX_DIR_DATA1,
   "successful Directed Tx's (MSDU) @ 1MB"),
    IPW2100_ORD(STAT_TX_DIR_DATA2,
   "successful Directed Tx's (MSDU) @ 2MB"),
    IPW2100_ORD(STAT_TX_DIR_DATA5_5,
   "successful Directed Tx's (MSDU) @ 5_5MB"),
    IPW2100_ORD(STAT_TX_DIR_DATA11,
```

```
    "successful Directed Tx's (MSDU) @ 11MB"),
    IPW2100_ORD(STAT_TX_NODIR_DATA1,
    "successful Non_Directed Tx's (MSDU) @ 1MB"),
    IPW2100_ORD(STAT_TX_NODIR_DATA2,
    "successful Non_Directed Tx's (MSDU) @ 2MB"),
    IPW2100_ORD(STAT_TX_NODIR_DATA5_5,
    "successful Non_Directed Tx's (MSDU) @ 5.5MB"),
    IPW2100_ORD(STAT_TX_NODIR_DATA11,
    "successful Non_Directed Tx's (MSDU)
@ 11MB"),
    IPW2100_ORD(STAT_NULL_DATA, "successful NULL data Tx's"),
    IPW2100_ORD(STAT_TX_RTS, "successful Tx RTS"),
    IPW2100_ORD(STAT_TX_CTS, "successful Tx CTS"),
    IPW2100_ORD(STAT_TX_ACK, "successful Tx ACK"),
    IPW2100_ORD(STAT_TX_ASSN, "successful Association Tx's"),
    IPW2100_ORD(STAT_TX_ASSN_RESP,
    "successful Association response Tx's"),
    IPW2100_ORD(STAT_TX_REASSN,
    "successful Reassociation Tx's"),
    IPW2100_ORD(STAT_TX_REASSN_RESP,
    "successful Reassociation response Tx's"),
    IPW2100_ORD(STAT_TX_PROBE,
    "probes successfully transmitted"),
    IPW2100_ORD(STAT_TX_PROBE_RESP,
    "probe responses successfully transmitted"),
    IPW2100_ORD(STAT_TX_BEACON, "tx beacon"),
    IPW2100_ORD(STAT_TX_ATIM, "Tx ATIM"),
    IPW2100_ORD(STAT_TX_DISASSN,
    "successful Disassociation TX"),
    IPW2100_ORD(STAT_TX_AUTH, "successful Authentication Tx"),
    IPW2100_ORD(STAT_TX_DEAUTH,
    "successful Deauthentication TX"),
    IPW2100_ORD(STAT_TX_TOTAL_BYTES,
    "Total
successful Tx data bytes"),
    IPW2100_ORD(STAT_TX_RETRIES, "Tx retries"),
    IPW2100_ORD(STAT_TX_RETRY1, "Tx retries at 1MBPS"),
    IPW2100_ORD(STAT_TX_RETRY2, "Tx retries at 2MBPS"),
    IPW2100_ORD(STAT_TX_RETRY5_5, "Tx retries at 5.5MBPS"),
    IPW2100_ORD(STAT_TX_RETRY11, "Tx retries at 11MBPS"),
    IPW2100_ORD(STAT_TX_FAILURES, "Tx Failures"),
    IPW2100_ORD(STAT_TX_MAX_TRIES_IN_HOP,
    "times max tries in a hop failed"),
    IPW2100_ORD(STAT_TX_DISASSN_FAIL,
    "times disassociation failed"),
    IPW2100_ORD(STAT_TX_ERR_CTS, "missed/bad CTS frames"),
    IPW2100_ORD(STAT_TX_ERR_ACK, "tx err due to acks"),
    IPW2100_ORD(STAT_RX_HOST, "packets passed to host"),
```

```
    IPW2100_ORD(STAT_RX_DIR_DATA, "directed packets"),
    IPW2100_ORD(STAT_RX_DIR_DATA1, "directed packets at 1MB"),
    IPW2100_ORD(STAT_RX_DIR_DATA2, "directed packets at 2MB"),
    IPW2100_ORD(STAT_RX_DIR_DATA5_5,
   "directed packets at 5.5MB"),
    IPW2100_ORD(STAT_RX_DIR_DATA11,
"directed packets at 11MB"),
    IPW2100_ORD(STAT_RX_NODIR_DATA, "nondirected packets"),
    IPW2100_ORD(STAT_RX_NODIR_DATA1,
   "nondirected packets at 1MB"),
    IPW2100_ORD(STAT_RX_NODIR_DATA2,
   "nondirected packets at 2MB"),
    IPW2100_ORD(STAT_RX_NODIR_DATA5_5,
   "nondirected packets at 5.5MB"),
    IPW2100_ORD(STAT_RX_NODIR_DATA11,
   "nondirected packets at 11MB"),
    IPW2100_ORD(STAT_RX_NULL_DATA, "null data rx's"),
    IPW2100_ORD(STAT_RX_RTS, "Rx RTS"), IPW2100_ORD(STAT_RX_CTS,
         "Rx CTS"),
    IPW2100_ORD(STAT_RX_ACK, "Rx ACK"),
    IPW2100_ORD(STAT_RX_CFEND, "Rx CF End"),
    IPW2100_ORD(STAT_RX_CFEND_ACK, "Rx CF End + CF Ack"),
    IPW2100_ORD(STAT_RX_ASSN, "Association Rx's"),
    IPW2100_ORD(STAT_RX_ASSN_RESP, "Association response Rx's"),
    IPW2100_ORD(STAT_RX_REASSN, "Reassociation Rx's"),
    IPW2100_ORD(STAT_RX_REASSN_RESP,
   "Reassociation response Rx's"),
    IPW2100_ORD(STAT_RX_PROBE, "probe Rx's"),

   IPW2100_ORD(STAT_RX_PROBE_RESP, "probe response Rx's"),
    IPW2100_ORD(STAT_RX_BEACON, "Rx beacon"),
    IPW2100_ORD(STAT_RX_ATIM, "Rx ATIM"),
    IPW2100_ORD(STAT_RX_DISASSN, "disassociation Rx"),
    IPW2100_ORD(STAT_RX_AUTH, "authentication Rx"),
    IPW2100_ORD(STAT_RX_DEAUTH, "deauthentication Rx"),
    IPW2100_ORD(STAT_RX_TOTAL_BYTES,
   "Total rx data bytes received"),
    IPW2100_ORD(STAT_RX_ERR_CRC, "packets with Rx CRC error"),
    IPW2100_ORD(STAT_RX_ERR_CRC1, "Rx CRC errors at 1MB"),
    IPW2100_ORD(STAT_RX_ERR_CRC2, "Rx CRC errors at 2MB"),
    IPW2100_ORD(STAT_RX_ERR_CRC5_5, "Rx CRC errors at 5.5MB"),
    IPW2100_ORD(STAT_RX_ERR_CRC11, "Rx CRC errors at 11MB"),
    IPW2100_ORD(STAT_RX_DUPLICATE1,
   "duplicate rx packets at 1MB"),
    IPW2100_ORD(STAT_RX_DUPLICATE2,
   "duplicate rx packets at 2MB"),
    IPW2100_ORD(STAT_RX_DUPLICATE5_5,
   "duplicate rx packets at 5.5MB"),
```

```
    IPW2100_ORD(STAT_RX_DUPLICATE11,
   "duplicate rx packets at 11MB"),

   IPW2100_ORD(STAT_RX_DUPLICATE, "duplicate rx packets"),
    IPW2100_ORD(PERS_DB_LOCK, "locking fw permanent  db"),
    IPW2100_ORD(PERS_DB_SIZE, "size of fw permanent  db"),
    IPW2100_ORD(PERS_DB_ADDR, "address of fw permanent  db"),
    IPW2100_ORD(STAT_RX_INVALID_PROTOCOL,
   "rx frames with invalid protocol"),
    IPW2100_ORD(SYS_BOOT_TIME, "Boot time"),
    IPW2100_ORD(STAT_RX_NO_BUFFER,
   "rx frames rejected due to no buffer"),
    IPW2100_ORD(STAT_RX_MISSING_FRAG,
   "rx frames dropped due to missing fragment"),
    IPW2100_ORD(STAT_RX_ORPHAN_FRAG,
   "rx frames dropped due to non-sequential fragment"),
    IPW2100_ORD(STAT_RX_ORPHAN_FRAME,
   "rx frames dropped due to unmatched 1st frame"),
    IPW2100_ORD(STAT_RX_FRAG_AGEOUT,
   "rx frames dropped due to uncompleted frame"),
    IPW2100_ORD(STAT_RX_ICV_ERRORS,
   "ICV errors during decryption"),
    IPW2100_ORD(STAT_PSP_SUSPENSION, "times adapter suspended"),
    IPW2100_ORD(STAT_PSP_BCN_TIMEOUT, "beacon
timeout"),
    IPW2100_ORD(STAT_PSP_POLL_TIMEOUT,
   "poll response timeouts"),
    IPW2100_ORD(STAT_PSP_NONDIR_TIMEOUT,
   "timeouts waiting for last {broad,multi}cast pkt"),
    IPW2100_ORD(STAT_PSP_RX_DTIMS, "PSP DTIMs received"),
    IPW2100_ORD(STAT_PSP_RX_TIMS, "PSP TIMs received"),
    IPW2100_ORD(STAT_PSP_STATION_ID, "PSP Station ID"),
    IPW2100_ORD(LAST_ASSN_TIME, "RTC time of last association"),
    IPW2100_ORD(STAT_PERCENT_MISSED_BCNS,
   "current calculation of % missed beacons"),
    IPW2100_ORD(STAT_PERCENT_RETRIES,
   "current calculation of % missed tx retries"),
    IPW2100_ORD(ASSOCIATED_AP_PTR,
   "0 if not associated, else pointer to AP table entry"),
    IPW2100_ORD(AVAILABLE_AP_CNT,
   "AP's decsribed in the AP table"),
    IPW2100_ORD(AP_LIST_PTR, "Ptr to list of available APs"),
    IPW2100_ORD(STAT_AP_ASSNS, "associations"),
    IPW2100_ORD(STAT_ASSN_FAIL, "association failures"),
    IPW2100_ORD(STAT_ASSN_RESP_FAIL,
   "failures due to
response fail"),
    IPW2100_ORD(STAT_FULL_SCANS, "full scans"),
```

```
    IPW2100_ORD(CARD_DISABLED, "Card Disabled"),
    IPW2100_ORD(STAT_ROAM_INHIBIT,
   "times roaming was inhibited due to activity"),
    IPW2100_ORD(RSSI_AT_ASSN,
   "RSSI of associated AP at time of association"),
    IPW2100_ORD(STAT_ASSN_CAUSE1,
   "reassociation: no probe response or TX on hop"),
    IPW2100_ORD(STAT_ASSN_CAUSE2,
   "reassociation: poor tx/rx quality"),
    IPW2100_ORD(STAT_ASSN_CAUSE3,
   "reassociation: tx/rx quality (excessive AP load"),
    IPW2100_ORD(STAT_ASSN_CAUSE4,
   "reassociation: AP RSSI level"),
    IPW2100_ORD(STAT_ASSN_CAUSE5,
   "reassociations due to load leveling"),
    IPW2100_ORD(STAT_AUTH_FAIL, "times authentication failed"),
    IPW2100_ORD(STAT_AUTH_RESP_FAIL,
   "times authentication response failed"),
    IPW2100_ORD(STATION_TABLE_CNT,
   "entries in association table"),
    IPW2100_ORD(RSSI_AVG_CURR, "Current avg RSSI"),
    IPW2100_ORD(POWER_MGMT_MODE,
"Power mode - 0=CAM, 1=PSP"),
    IPW2100_ORD(COUNTRY_CODE,
   "IEEE country code as recv'd from beacon"),
    IPW2100_ORD(COUNTRY_CHANNELS,
   "channels suported by country"),
    IPW2100_ORD(RESET_CNT, "adapter resets (warm)"),
    IPW2100_ORD(BEACON_INTERVAL, "Beacon interval"),
    IPW2100_ORD(ANTENNA_DIVERSITY,
   "TRUE if antenna diversity is disabled"),
    IPW2100_ORD(DTIM_PERIOD, "beacon intervals between DTIMs"),
    IPW2100_ORD(OUR_FREQ,
   "current radio freq lower digits - channel ID"),
    IPW2100_ORD(RTC_TIME, "current RTC time"),
    IPW2100_ORD(PORT_TYPE, "operating mode"),
    IPW2100_ORD(CURRENT_TX_RATE, "current tx rate"),
    IPW2100_ORD(SUPPORTED_RATES, "supported tx rates"),
    IPW2100_ORD(ATIM_WINDOW, "current ATIM Window"),
    IPW2100_ORD(BASIC_RATES, "basic tx rates"),
    IPW2100_ORD(NIC_HIGHEST_RATE, "NIC highest tx rate"),
    IPW2100_ORD(AP_HIGHEST_RATE, "AP highest tx rate"),
    IPW2100_ORD(CAPABILITIES,
   "Management frame
capability field"),
    IPW2100_ORD(AUTH_TYPE, "Type of authentication"),
    IPW2100_ORD(RADIO_TYPE, "Adapter card platform type"),
    IPW2100_ORD(RTS_THRESHOLD,
```

```
    "Min packet length for RTS handshaking"),
     IPW2100_ORD(INT_MODE, "International mode"),
     IPW2100_ORD(FRAGMENTATION_THRESHOLD,
    "protocol frag threshold"),
     IPW2100_ORD(EEPROM_SRAM_DB_BLOCK_START_ADDRESS,
    "EEPROM offset in SRAM"),
     IPW2100_ORD(EEPROM_SRAM_DB_BLOCK_SIZE,
    "EEPROM size in SRAM"),
     IPW2100_ORD(EEPROM_SKU_CAPABILITY, "EEPROM SKU Capability"),
     IPW2100_ORD(EEPROM_IBSS_11B_CHANNELS,
    "EEPROM IBSS 11b channel set"),
     IPW2100_ORD(MAC_VERSION, "MAC Version"),
     IPW2100_ORD(MAC_REVISION, "MAC Revision"),
     IPW2100_ORD(RADIO_VERSION, "Radio Version"),
     IPW2100_ORD(NIC_MANF_DATE_TIME, "MANF Date/Time STAMP"),
     IPW2100_ORD(UCODE_VERSION, "Ucode Version"),};

static ssize_t show_registers(struct device *d, struct device_attribute *attr,
        char *buf)
{
 int
 i;
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 struct net_device *dev = priv->net_dev;
 char *out = buf;
 u32 val = 0;

 out += sprintf(out, "%30s [Address ] : Hex\n", "Register");

 for (i = 0; i < (sizeof(hw_data) / sizeof(*hw_data)); i++) {
  read_register(dev, hw_data[i].addr, &val);
  out += sprintf(out, "%30s [%08X] : %08X\n",
         hw_data[i].name, hw_data[i].addr, val);
 }

 return out - buf;
}

static DEVICE_ATTR(registers, S_IRUGO, show_registers, NULL);

static ssize_t show_hardware(struct device *d, struct device_attribute *attr,
       char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 struct net_device *dev = priv->net_dev;
 char *out = buf;
 int i;
```

```
 out += sprintf(out, "%30s [Address ] : Hex\n", "NIC entry");

 for (i = 0; i < (sizeof(nic_data) / sizeof(*nic_data)); i++) {
  u8 tmp8;
  u16 tmp16;
  u32 tmp32;

  switch (nic_data[i].size) {
  case 1:
   read_nic_byte(dev, nic_data[i].addr, &tmp8);
   out += sprintf(out, "%30s [%08X] : %02X\n",

       nic_data[i].name, nic_data[i].addr,
           tmp8);
   break;
  case 2:
   read_nic_word(dev, nic_data[i].addr, &tmp16);
   out += sprintf(out, "%30s [%08X] : %04X\n",
           nic_data[i].name, nic_data[i].addr,
           tmp16);
   break;
  case 4:
   read_nic_dword(dev, nic_data[i].addr, &tmp32);
   out += sprintf(out, "%30s [%08X] : %08X\n",
           nic_data[i].name, nic_data[i].addr,
           tmp32);
   break;
  }
 }
 return out - buf;
}

static DEVICE_ATTR(hardware, S_IRUGO, show_hardware, NULL);

static ssize_t show_memory(struct device *d, struct device_attribute *attr,
      char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 struct net_device *dev = priv->net_dev;
 static unsigned long loop = 0;
 int len = 0;
 u32 buffer[4];
 int i;
 char line[81];

 if (loop >= 0x30000)
  loop = 0;
```

```
/* sysfs provides us PAGE_SIZE buffer */
while (len < PAGE_SIZE - 128 && loop < 0x30000) {

 if (priv->snapshot[0])
  for (i = 0; i < 4; i++)
   buffer[i] =

   *(u32 *) SNAPSHOT_ADDR(loop + i * 4);
 else
  for (i = 0; i < 4; i++)
   read_nic_dword(dev, loop + i * 4, &buffer[i]);

 if (priv->dump_raw)
  len += sprintf(buf + len,
          "%c%c%c%c"
          "%c%c%c%c"
          "%c%c%c%c"
          "%c%c%c%c",
          ((u8 *) buffer)[0x0],
          ((u8 *) buffer)[0x1],
          ((u8 *) buffer)[0x2],
          ((u8 *) buffer)[0x3],
          ((u8 *) buffer)[0x4],
          ((u8 *) buffer)[0x5],
          ((u8 *) buffer)[0x6],
          ((u8 *) buffer)[0x7],
          ((u8 *) buffer)[0x8],
          ((u8 *) buffer)[0x9],
          ((u8 *) buffer)[0xa],
          ((u8 *) buffer)[0xb],
          ((u8 *) buffer)[0xc],
          ((u8 *) buffer)[0xd],
          ((u8 *) buffer)[0xe],
          ((u8 *) buffer)[0xf]);
 else
  len += sprintf(buf + len, "%s\n",
          snprint_line(line, sizeof(line),
         (u8 *) buffer, 16, loop));
 loop += 16;
 }

 return len;
}

static ssize_t store_memory(struct device *d,
 struct device_attribute *attr,
       const char *buf, size_t count)
{
```

```
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 struct net_device *dev = priv->net_dev;
 const char *p = buf;

 (void)dev;  /* kill unused-var warning for debug-only code */

 if (count < 1)
  return count;

 if (p[0] == '1' ||
     (count >= 2 && tolower(p[0]) == 'o' && tolower(p[1]) == 'n')) {
  IPW_DEBUG_INFO("%s: Setting memory dump to RAW mode.\n",
         dev->name);
  priv->dump_raw = 1;

 } else if (p[0] == '0' || (count >= 2 && tolower(p[0]) == 'o' &&
       tolower(p[1]) == 'f')) {
  IPW_DEBUG_INFO("%s: Setting memory dump to HEX mode.\n",
         dev->name);
  priv->dump_raw = 0;

 } else if (tolower(p[0]) == 'r') {
  IPW_DEBUG_INFO("%s: Resetting firmware snapshot.\n", dev->name);
  ipw2100_snapshot_free(priv);

 } else
  IPW_DEBUG_INFO("%s: Usage: 0|on = HEX, 1|off = RAW, "
         "reset = clear memory snapshot\n", dev->name);

 return count;
}

static DEVICE_ATTR(memory, S_IWUSR | S_IRUGO,
 show_memory, store_memory);

static ssize_t show_ordinals(struct device *d, struct device_attribute *attr,
        char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 u32 val = 0;
 int len = 0;
 u32 val_len;
 static int loop = 0;

 if (priv->status & STATUS_RF_KILL_MASK)
  return 0;

 if (loop >= sizeof(ord_data) / sizeof(*ord_data))
```

```
 loop = 0;

 /* sysfs provides us PAGE_SIZE buffer */
 while (len < PAGE_SIZE - 128 &&
        loop < (sizeof(ord_data) / sizeof(*ord_data))) {

 val_len = sizeof(u32);

 if (ipw2100_get_ordinal(priv, ord_data[loop].index, &val,
    &val_len))
  len += sprintf(buf + len, "[0x%02X] = ERROR    %s\n",
          ord_data[loop].index,
          ord_data[loop].desc);
 else
  len += sprintf(buf + len, "[0x%02X] = 0x%08X %s\n",
          ord_data[loop].index, val,
          ord_data[loop].desc);
 loop++;
 }

 return len;
}

static DEVICE_ATTR(ordinals, S_IRUGO, show_ordinals, NULL);

static ssize_t show_stats(struct device *d, struct device_attribute
 *attr,
    char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 char *out = buf;

 out += sprintf(out, "interrupts: %d {tx: %d, rx: %d, other: %d}\n",
         priv->interrupts, priv->tx_interrupts,
         priv->rx_interrupts, priv->inta_other);
 out += sprintf(out, "firmware resets: %d\n", priv->resets);
 out += sprintf(out, "firmware hangs: %d\n", priv->hangs);
#ifdef CONFIG_IPW2100_DEBUG
 out += sprintf(out, "packet mismatch image: %s\n",
         priv->snapshot[0] ? "YES" : "NO");
#endif

 return out - buf;
}

static DEVICE_ATTR(stats, S_IRUGO, show_stats, NULL);

static int ipw2100_switch_mode(struct ipw2100_priv *priv, u32 mode)
```

```
{
 int err;

 if (mode == priv->ieee->iw_mode)
  return 0;

 err = ipw2100_disable_adapter(priv);
 if (err) {
  printk(KERN_ERR DRV_NAME ": %s: Could not disable adapter %d\n",
         priv->net_dev->name, err);
  return err;
 }

 switch (mode) {
 case IW_MODE_INFRA:
  priv->net_dev->type = ARPHRD_ETHER;
  break;
 case IW_MODE_ADHOC:
  priv->net_dev->type
 = ARPHRD_ETHER;
  break;
#ifdef CONFIG_IPW2100_MONITOR
 case IW_MODE_MONITOR:
  priv->last_mode = priv->ieee->iw_mode;
  priv->net_dev->type = ARPHRD_IEEE80211_RADIOTAP;
  break;
#endif    /* CONFIG_IPW2100_MONITOR */
 }

 priv->ieee->iw_mode = mode;

#ifdef CONFIG_PM
 /* Indicate ipw2100_download_firmware download firmware
  * from disk instead of memory. */
 ipw2100_firmware.version = 0;
#endif

 printk(KERN_INFO "%s: Reseting on mode change.\n", priv->net_dev->name);
 priv->reset_backoff = 0;
 schedule_reset(priv);

 return 0;
}

static ssize_t show_internals(struct device *d, struct device_attribute *attr,
         char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
```

```
 int len = 0;

#define DUMP_VAR(x,y) len += sprintf(buf + len, # x ": %" y "\n", priv-> x)

 if (priv->status & STATUS_ASSOCIATED)
  len += sprintf(buf + len, "connected: %lu\n",
         get_seconds() - priv->connect_start);
 else
  len += sprintf(buf + len, "not connected\n");

 DUMP_VAR(ieee->crypt[priv->ieee->tx_keyidx],
 "p");
 DUMP_VAR(status, "08lx");
 DUMP_VAR(config, "08lx");
 DUMP_VAR(capability, "08lx");

 len +=
     sprintf(buf + len, "last_rtc: %lu\n",
      (unsigned long)priv->last_rtc);

 DUMP_VAR(fatal_error, "d");
 DUMP_VAR(stop_hang_check, "d");
 DUMP_VAR(stop_rf_kill, "d");
 DUMP_VAR(messages_sent, "d");

 DUMP_VAR(tx_pend_stat.value, "d");
 DUMP_VAR(tx_pend_stat.hi, "d");

 DUMP_VAR(tx_free_stat.value, "d");
 DUMP_VAR(tx_free_stat.lo, "d");

 DUMP_VAR(msg_free_stat.value, "d");
 DUMP_VAR(msg_free_stat.lo, "d");

 DUMP_VAR(msg_pend_stat.value, "d");
 DUMP_VAR(msg_pend_stat.hi, "d");

 DUMP_VAR(fw_pend_stat.value, "d");
 DUMP_VAR(fw_pend_stat.hi, "d");

 DUMP_VAR(txq_stat.value, "d");
 DUMP_VAR(txq_stat.lo, "d");

 DUMP_VAR(ieee->scans, "d");
 DUMP_VAR(reset_backoff, "d");

 return len;
}
```

```
static DEVICE_ATTR(internals, S_IRUGO, show_internals, NULL);

static ssize_t show_bssinfo(struct device *d, struct device_attribute *attr,
      char *buf)
{
 struct
 ipw2100_priv *priv = dev_get_drvdata(d);
 char essid[IW_ESSID_MAX_SIZE + 1];
 u8 bssid[ETH_ALEN];
 u32 chan = 0;
 char *out = buf;
 int length;
 int ret;

 if (priv->status & STATUS_RF_KILL_MASK)
  return 0;

 memset(essid, 0, sizeof(essid));
 memset(bssid, 0, sizeof(bssid));

 length = IW_ESSID_MAX_SIZE;
 ret = ipw2100_get_ordinal(priv, IPW_ORD_STAT_ASSN_SSID, essid, &length);
 if (ret)
  IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
         __LINE__);

 length = sizeof(bssid);
 ret = ipw2100_get_ordinal(priv, IPW_ORD_STAT_ASSN_AP_BSSID,
     bssid, &length);
 if (ret)
  IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
         __LINE__);

 length = sizeof(u32);
 ret = ipw2100_get_ordinal(priv, IPW_ORD_OUR_FREQ, &chan, &length);
 if (ret)
  IPW_DEBUG_INFO("failed querying ordinals at line %d\n",
         __LINE__);

 out += sprintf(out, "ESSID: %s\n", essid);
 out += sprintf(out, "BSSID:   %02x:%02x:%02x:%02x:%02x:%02x\n",
        bssid[0], bssid[1], bssid[2],

       bssid[3], bssid[4], bssid[5]);
 out += sprintf(out, "Channel: %d\n", chan);

 return out - buf;
```

```
}

static DEVICE_ATTR(bssinfo, S_IRUGO, show_bssinfo, NULL);

#ifdef CONFIG_IPW2100_DEBUG
static ssize_t show_debug_level(struct device_driver *d, char *buf)
{
 return sprintf(buf, "0x%08X\n", ipw2100_debug_level);
}

static ssize_t store_debug_level(struct device_driver *d,
     const char *buf, size_t count)
{
 char *p = (char *)buf;
 u32 val;

 if (p[1] == 'x' || p[1] == 'X' || p[0] == 'x' || p[0] == 'X') {
  p++;
  if (p[0] == 'x' || p[0] == 'X')
   p++;
  val = simple_strtoul(p, &p, 16);
 } else
  val = simple_strtoul(p, &p, 10);
 if (p == buf)
  IPW_DEBUG_INFO(": %s is not in hex or decimal form.\n", buf);
 else
  ipw2100_debug_level = val;

 return strnlen(buf, count);
}

static DRIVER_ATTR(debug_level, S_IWUSR | S_IRUGO, show_debug_level,
     store_debug_level);
#endif    /* CONFIG_IPW2100_DEBUG */

static ssize_t show_fatal_error(struct device *d,
    struct device_attribute
 *attr, char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 char *out = buf;
 int i;

 if (priv->fatal_error)
  out += sprintf(out, "0x%08X\n", priv->fatal_error);
 else
  out += sprintf(out, "0\n");
```

```
 for (i = 1; i <= IPW2100_ERROR_QUEUE; i++) {
  if (!priv->fatal_errors[(priv->fatal_index - i) %
     IPW2100_ERROR_QUEUE])
   continue;

  out += sprintf(out, "%d. 0x%08X\n", i,
         priv->fatal_errors[(priv->fatal_index - i) %
        IPW2100_ERROR_QUEUE]);
 }

 return out - buf;
}

static ssize_t store_fatal_error(struct device *d,
     struct device_attribute *attr, const char *buf,
     size_t count)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 schedule_reset(priv);
 return count;
}

static DEVICE_ATTR(fatal_error, S_IWUSR | S_IRUGO, show_fatal_error,
     store_fatal_error);

static ssize_t show_scan_age(struct device *d, struct device_attribute *attr,
        char *buf)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 return sprintf(buf, "%d\n",
 priv->ieee->scan_age);
}

static ssize_t store_scan_age(struct device *d, struct device_attribute *attr,
         const char *buf, size_t count)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 struct net_device *dev = priv->net_dev;
 char buffer[] = "00000000";
 unsigned long len =
     (sizeof(buffer) - 1) > count ? count : sizeof(buffer) - 1;
 unsigned long val;
 char *p = buffer;

 (void)dev;  /* kill unused-var warning for debug-only code */

 IPW_DEBUG_INFO("enter\n");
```

```
 strncpy(buffer, buf, len);
 buffer[len] = 0;

 if (p[1] == 'x' || p[1] == 'X' || p[0] == 'x' || p[0] == 'X') {
  p++;
  if (p[0] == 'x' || p[0] == 'X')
   p++;
  val = simple_strtoul(p, &p, 16);
 } else
  val = simple_strtoul(p, &p, 10);
 if (p == buffer) {
  IPW_DEBUG_INFO("%s: user supplied invalid value.\n", dev->name);
 } else {
  priv->ieee->scan_age = val;
  IPW_DEBUG_INFO("set scan_age = %u\n", priv->ieee->scan_age);
 }

 IPW_DEBUG_INFO("exit\n");
 return len;
}

static DEVICE_ATTR(scan_age, S_IWUSR |
 S_IRUGO, show_scan_age, store_scan_age);

static ssize_t show_rf_kill(struct device *d, struct device_attribute *attr,
       char *buf)
{
 /* 0 - RF kill not enabled
    1 - SW based RF kill active (sysfs)
    2 - HW based RF kill active
    3 - Both HW and SW baed RF kill active */
 struct ipw2100_priv *priv = (struct ipw2100_priv *)d->driver_data;
 int val = ((priv->status & STATUS_RF_KILL_SW) ? 0x1 : 0x0) |
     (rf_kill_active(priv) ? 0x2 : 0x0);
 return sprintf(buf, "%i\n", val);
}

static int ipw_radio_kill_sw(struct ipw2100_priv *priv, int disable_radio)
{
 if ((disable_radio ? 1 : 0) ==
     (priv->status & STATUS_RF_KILL_SW ? 1 : 0))
  return 0;

 IPW_DEBUG_RF_KILL("Manual SW RF Kill set to: RADIO  %s\n",
     disable_radio ? "OFF" : "ON");

 mutex_lock(&priv->action_mutex);
```

```
 if (disable_radio) {
  priv->status |= STATUS_RF_KILL_SW;
  ipw2100_down(priv);
 } else {
  priv->status &= ~STATUS_RF_KILL_SW;
  if (rf_kill_active(priv)) {
   IPW_DEBUG_RF_KILL("Can not turn radio back
 on - "
       "disabled by HW switch\n");
   /* Make sure the RF_KILL check timer is running */
   priv->stop_rf_kill = 0;
   cancel_delayed_work(&priv->rf_kill);
   queue_delayed_work(priv->workqueue, &priv->rf_kill, HZ);
  } else
   schedule_reset(priv);
 }

 mutex_unlock(&priv->action_mutex);
 return 1;
}

static ssize_t store_rf_kill(struct device *d, struct device_attribute *attr,
        const char *buf, size_t count)
{
 struct ipw2100_priv *priv = dev_get_drvdata(d);
 ipw_radio_kill_sw(priv, buf[0] == '1');
 return count;
}

static DEVICE_ATTR(rf_kill, S_IWUSR | S_IRUGO, show_rf_kill, store_rf_kill);

static struct attribute *ipw2100_sysfs_entries[] = {
 &dev_attr_hardware.attr,
 &dev_attr_registers.attr,
 &dev_attr_ordinals.attr,
 &dev_attr_pci.attr,
 &dev_attr_stats.attr,
 &dev_attr_internals.attr,
 &dev_attr_bssinfo.attr,
 &dev_attr_memory.attr,
 &dev_attr_scan_age.attr,
 &dev_attr_fatal_error.attr,
 &dev_attr_rf_kill.attr,
 &dev_attr_cfg.attr,
 &dev_attr_status.attr,
 &dev_attr_capability.attr,
 NULL,
};
```

```
static
 struct attribute_group ipw2100_attribute_group = {
 .attrs = ipw2100_sysfs_entries,
};

static int status_queue_allocate(struct ipw2100_priv *priv, int entries)
{
 struct ipw2100_status_queue *q = &priv->status_queue;

 IPW_DEBUG_INFO("enter\n");

 q->size = entries * sizeof(struct ipw2100_status);
 q->drv =
     (struct ipw2100_status *)pci_alloc_consistent(priv->pci_dev,
         q->size, &q->nic);
 if (!q->drv) {
  IPW_DEBUG_WARNING("Can not allocate status queue.\n");
  return -ENOMEM;
 }

 memset(q->drv, 0, q->size);

 IPW_DEBUG_INFO("exit\n");

 return 0;
}

static void status_queue_free(struct ipw2100_priv *priv)
{
 IPW_DEBUG_INFO("enter\n");

 if (priv->status_queue.drv) {
  pci_free_consistent(priv->pci_dev, priv->status_queue.size,
        priv->status_queue.drv,
        priv->status_queue.nic);
  priv->status_queue.drv = NULL;
 }

 IPW_DEBUG_INFO("exit\n");
}

static int bd_queue_allocate(struct ipw2100_priv *priv,
        struct ipw2100_bd_queue
 *q, int entries)
{
 IPW_DEBUG_INFO("enter\n");
```

```
 memset(q, 0, sizeof(struct ipw2100_bd_queue));

 q->entries = entries;
 q->size = entries * sizeof(struct ipw2100_bd);
 q->drv = pci_alloc_consistent(priv->pci_dev, q->size, &q->nic);
 if (!q->drv) {
  IPW_DEBUG_INFO
      ("can't allocate shared memory for buffer descriptors\n");
  return -ENOMEM;
 }
 memset(q->drv, 0, q->size);

 IPW_DEBUG_INFO("exit\n");

 return 0;
}

static void bd_queue_free(struct ipw2100_priv *priv, struct ipw2100_bd_queue *q)
{
 IPW_DEBUG_INFO("enter\n");

 if (!q)
  return;

 if (q->drv) {
  pci_free_consistent(priv->pci_dev, q->size, q->drv, q->nic);
  q->drv = NULL;
 }

 IPW_DEBUG_INFO("exit\n");
}

static void bd_queue_initialize(struct ipw2100_priv *priv,
    struct ipw2100_bd_queue *q, u32 base, u32 size,
    u32 r, u32 w)
{
 IPW_DEBUG_INFO("enter\n");

 IPW_DEBUG_INFO("initializing bd queue at virt=%p, phys=%08x\n", q->drv,
        (u32) q->nic);

 write_register(priv->net_dev, base, q->nic);
 write_register(priv->net_dev,
 size, q->entries);
 write_register(priv->net_dev, r, q->oldest);
 write_register(priv->net_dev, w, q->next);

 IPW_DEBUG_INFO("exit\n");
```

```
}

static void ipw2100_kill_workqueue(struct ipw2100_priv *priv)
{
 if (priv->workqueue) {
  priv->stop_rf_kill = 1;
  priv->stop_hang_check = 1;
  cancel_delayed_work(&priv->reset_work);
  cancel_delayed_work(&priv->security_work);
  cancel_delayed_work(&priv->wx_event_work);
  cancel_delayed_work(&priv->hang_check);
  cancel_delayed_work(&priv->rf_kill);
  destroy_workqueue(priv->workqueue);
  priv->workqueue = NULL;
 }
}

static int ipw2100_tx_allocate(struct ipw2100_priv *priv)
{
 int i, j, err = -EINVAL;
 void *v;
 dma_addr_t p;

 IPW_DEBUG_INFO("enter\n");

 err = bd_queue_allocate(priv, &priv->tx_queue, TX_QUEUE_LENGTH);
 if (err) {
  IPW_DEBUG_ERROR("%s: failed bd_queue_allocate\n",
    priv->net_dev->name);
  return err;
 }

 priv->tx_buffers =
     (struct ipw2100_tx_packet *)kmalloc(TX_PENDED_QUEUE_LENGTH *
      sizeof(struct

       ipw2100_tx_packet),
      GFP_ATOMIC);
 if (!priv->tx_buffers) {
  printk(KERN_ERR DRV_NAME
         ": %s: alloc failed form tx buffers.\n",
         priv->net_dev->name);
  bd_queue_free(priv, &priv->tx_queue);
  return -ENOMEM;
 }

 for (i = 0; i < TX_PENDED_QUEUE_LENGTH; i++) {
  v = pci_alloc_consistent(priv->pci_dev,
```

```
    sizeof(struct ipw2100_data_header),
    &p);
 if (!v) {
  printk(KERN_ERR DRV_NAME
        ": %s: PCI alloc failed for tx " "buffers.\n",
        priv->net_dev->name);
  err = -ENOMEM;
  break;
 }

 priv->tx_buffers[i].type = DATA;
 priv->tx_buffers[i].info.d_struct.data =
    (struct ipw2100_data_header *)v;
 priv->tx_buffers[i].info.d_struct.data_phys = p;
 priv->tx_buffers[i].info.d_struct.txb = NULL;
}

if (i == TX_PENDED_QUEUE_LENGTH)
 return 0;

for (j = 0; j < i; j++) {
 pci_free_consistent(priv->pci_dev,
       sizeof(struct ipw2100_data_header),
       priv->tx_buffers[j].info.d_struct.data,

   priv->tx_buffers[j].info.d_struct.
       data_phys);
}

kfree(priv->tx_buffers);
priv->tx_buffers = NULL;

return err;
}

static void ipw2100_tx_initialize(struct ipw2100_priv *priv)
{
 int i;

 IPW_DEBUG_INFO("enter\n");

 /*
  * reinitialize packet info lists
  */
 INIT_LIST_HEAD(&priv->fw_pend_list);
 INIT_STAT(&priv->fw_pend_stat);

 /*
```

```
 * reinitialize lists
 */
INIT_LIST_HEAD(&priv->tx_pend_list);
INIT_LIST_HEAD(&priv->tx_free_list);
INIT_STAT(&priv->tx_pend_stat);
INIT_STAT(&priv->tx_free_stat);

for (i = 0; i < TX_PENDED_QUEUE_LENGTH; i++) {
 /* We simply drop any SKBs that have been queued for
  * transmit */
 if (priv->tx_buffers[i].info.d_struct.txb) {
  ieee80211_txb_free(priv->tx_buffers[i].info.d_struct.
      txb);
  priv->tx_buffers[i].info.d_struct.txb = NULL;
 }

 list_add_tail(&priv->tx_buffers[i].list, &priv->tx_free_list);
}

SET_STAT(&priv->tx_free_stat, i);

priv->tx_queue.oldest = 0;
priv->tx_queue.available = priv->tx_queue.entries;
priv->tx_queue.next
= 0;
INIT_STAT(&priv->txq_stat);
SET_STAT(&priv->txq_stat, priv->tx_queue.available);

bd_queue_initialize(priv, &priv->tx_queue,
      IPW_MEM_HOST_SHARED_TX_QUEUE_BD_BASE,
      IPW_MEM_HOST_SHARED_TX_QUEUE_BD_SIZE,
      IPW_MEM_HOST_SHARED_TX_QUEUE_READ_INDEX,
      IPW_MEM_HOST_SHARED_TX_QUEUE_WRITE_INDEX);

IPW_DEBUG_INFO("exit\n");

}

static void ipw2100_tx_free(struct ipw2100_priv *priv)
{
 int i;

 IPW_DEBUG_INFO("enter\n");

 bd_queue_free(priv, &priv->tx_queue);

 if (!priv->tx_buffers)
  return;
```

```
 for (i = 0; i < TX_PENDED_QUEUE_LENGTH; i++) {
  if (priv->tx_buffers[i].info.d_struct.txb) {
   ieee80211_txb_free(priv->tx_buffers[i].info.d_struct.
        txb);
   priv->tx_buffers[i].info.d_struct.txb = NULL;
  }
  if (priv->tx_buffers[i].info.d_struct.data)
   pci_free_consistent(priv->pci_dev,
         sizeof(struct ipw2100_data_header),
         priv->tx_buffers[i].info.d_struct.
         data,
         priv->tx_buffers[i].info.d_struct.

    data_phys);
 }

 kfree(priv->tx_buffers);
 priv->tx_buffers = NULL;

 IPW_DEBUG_INFO("exit\n");
}

static int ipw2100_rx_allocate(struct ipw2100_priv *priv)
{
 int i, j, err = -EINVAL;

 IPW_DEBUG_INFO("enter\n");

 err = bd_queue_allocate(priv, &priv->rx_queue, RX_QUEUE_LENGTH);
 if (err) {
  IPW_DEBUG_INFO("failed bd_queue_allocate\n");
  return err;
 }

 err = status_queue_allocate(priv, RX_QUEUE_LENGTH);
 if (err) {
  IPW_DEBUG_INFO("failed status_queue_allocate\n");
  bd_queue_free(priv, &priv->rx_queue);
  return err;
 }

 /*
  * allocate packets
  */
 priv->rx_buffers = (struct ipw2100_rx_packet *)
     kmalloc(RX_QUEUE_LENGTH * sizeof(struct ipw2100_rx_packet),
      GFP_KERNEL);
```

```
if (!priv->rx_buffers) {
 IPW_DEBUG_INFO("can't allocate rx packet buffer table\n");

 bd_queue_free(priv, &priv->rx_queue);

 status_queue_free(priv);

 return -ENOMEM;
}

for (i = 0; i < RX_QUEUE_LENGTH; i++) {
 struct ipw2100_rx_packet *packet = &priv->rx_buffers[i];

 err = ipw2100_alloc_skb(priv,
packet);
 if (unlikely(err)) {
  err = -ENOMEM;
  break;
 }

 /* The BD holds the cache aligned address */
 priv->rx_queue.drv[i].host_addr = packet->dma_addr;
 priv->rx_queue.drv[i].buf_length = IPW_RX_NIC_BUFFER_LENGTH;
 priv->status_queue.drv[i].status_fields = 0;
}

if (i == RX_QUEUE_LENGTH)
 return 0;

for (j = 0; j < i; j++) {
 pci_unmap_single(priv->pci_dev, priv->rx_buffers[j].dma_addr,
    sizeof(struct ipw2100_rx_packet),
    PCI_DMA_FROMDEVICE);
 dev_kfree_skb(priv->rx_buffers[j].skb);
}

kfree(priv->rx_buffers);
priv->rx_buffers = NULL;

bd_queue_free(priv, &priv->rx_queue);

status_queue_free(priv);

return err;
}

static void ipw2100_rx_initialize(struct ipw2100_priv *priv)
{
```

```
 IPW_DEBUG_INFO("enter\n");

 priv->rx_queue.oldest = 0;
 priv->rx_queue.available = priv->rx_queue.entries - 1;
 priv->rx_queue.next = priv->rx_queue.entries - 1;

 INIT_STAT(&priv->rxq_stat);
 SET_STAT(&priv->rxq_stat, priv->rx_queue.available);

 bd_queue_initialize(priv,
 &priv->rx_queue,
      IPW_MEM_HOST_SHARED_RX_BD_BASE,
      IPW_MEM_HOST_SHARED_RX_BD_SIZE,
      IPW_MEM_HOST_SHARED_RX_READ_INDEX,
      IPW_MEM_HOST_SHARED_RX_WRITE_INDEX);

 /* set up the status queue */
 write_register(priv->net_dev, IPW_MEM_HOST_SHARED_RX_STATUS_BASE,
        priv->status_queue.nic);

 IPW_DEBUG_INFO("exit\n");
}

static void ipw2100_rx_free(struct ipw2100_priv *priv)
{
 int i;

 IPW_DEBUG_INFO("enter\n");

 bd_queue_free(priv, &priv->rx_queue);
 status_queue_free(priv);

 if (!priv->rx_buffers)
  return;

 for (i = 0; i < RX_QUEUE_LENGTH; i++) {
  if (priv->rx_buffers[i].rxp) {
   pci_unmap_single(priv->pci_dev,
      priv->rx_buffers[i].dma_addr,
      sizeof(struct ipw2100_rx),
      PCI_DMA_FROMDEVICE);
   dev_kfree_skb(priv->rx_buffers[i].skb);
  }
 }

 kfree(priv->rx_buffers);
 priv->rx_buffers = NULL;
```

```
 IPW_DEBUG_INFO("exit\n");
}

static int ipw2100_read_mac_address(struct ipw2100_priv *priv)
{
 u32 length = ETH_ALEN;
 u8 mac[ETH_ALEN];

 int
 err;

 err = ipw2100_get_ordinal(priv, IPW_ORD_STAT_ADAPTER_MAC, mac, &length);
 if (err) {
  IPW_DEBUG_INFO("MAC address read failed\n");
  return -EIO;
 }
 IPW_DEBUG_INFO("card MAC is %02X:%02X:%02X:%02X:%02X:%02X\n",
        mac[0], mac[1], mac[2], mac[3], mac[4], mac[5]);

 memcpy(priv->net_dev->dev_addr, mac, ETH_ALEN);

 return 0;
}

/********************************************************************
*
* Firmware Commands
*
********************************************************************/

static int ipw2100_set_mac_address(struct ipw2100_priv *priv, int batch_mode)
{
 struct host_command cmd = {
  .host_command = ADAPTER_ADDRESS,
  .host_command_sequence = 0,
  .host_command_length = ETH_ALEN
 };
 int err;

 IPW_DEBUG_HC("SET_MAC_ADDRESS\n");

 IPW_DEBUG_INFO("enter\n");

 if (priv->config & CFG_CUSTOM_MAC) {
  memcpy(cmd.host_command_parameters, priv->mac_addr, ETH_ALEN);
  memcpy(priv->net_dev->dev_addr, priv->mac_addr, ETH_ALEN);
 } else
  memcpy(cmd.host_command_parameters,
```

```
 priv->net_dev->dev_addr,
        ETH_ALEN);

 err = ipw2100_hw_send_command(priv, &cmd);

 IPW_DEBUG_INFO("exit\n");
 return err;
}

static int ipw2100_set_port_type(struct ipw2100_priv *priv, u32 port_type,
     int batch_mode)
{
 struct host_command cmd = {
  .host_command = PORT_TYPE,
  .host_command_sequence = 0,
  .host_command_length = sizeof(u32)
 };
 int err;

 switch (port_type) {
 case IW_MODE_INFRA:
  cmd.host_command_parameters[0] = IPW_BSS;
  break;
 case IW_MODE_ADHOC:
  cmd.host_command_parameters[0] = IPW_IBSS;
  break;
 }

 IPW_DEBUG_HC("PORT_TYPE: %s\n",
       port_type == IPW_IBSS ? "Ad-Hoc" : "Managed");

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err) {
   printk(KERN_ERR DRV_NAME
          ": %s: Could not disable adapter %d\n",
          priv->net_dev->name, err);
   return err;
  }
 }

 /* send cmd to firmware */
 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode)
  ipw2100_enable_adapter(priv);

 return err;
```

```
}

static
 int ipw2100_set_channel(struct ipw2100_priv *priv, u32 channel,
         int batch_mode)
{
 struct host_command cmd = {
  .host_command = CHANNEL,
  .host_command_sequence = 0,
  .host_command_length = sizeof(u32)
 };
 int err;

 cmd.host_command_parameters[0] = channel;

 IPW_DEBUG_HC("CHANNEL: %d\n", channel);

 /* If BSS then we don't support channel selection */
 if (priv->ieee->iw_mode == IW_MODE_INFRA)
  return 0;

 if ((channel != 0) &&
     ((channel < REG_MIN_CHANNEL) || (channel > REG_MAX_CHANNEL)))
  return -EINVAL;

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err) {
  IPW_DEBUG_INFO("Failed to set channel to %d", channel);
  return err;
 }

 if (channel)
  priv->config |= CFG_STATIC_CHANNEL;
 else
  priv->config &= ~CFG_STATIC_CHANNEL;

 priv->channel = channel;

 if (!batch_mode) {
  err = ipw2100_enable_adapter(priv);
  if (err)
   return err;
```

```
 }

 return 0;
}

static int
 ipw2100_system_config(struct ipw2100_priv *priv, int batch_mode)
{
 struct host_command cmd = {
  .host_command = SYSTEM_CONFIG,
  .host_command_sequence = 0,
  .host_command_length = 12,
 };
 u32 ibss_mask, len = sizeof(u32);
 int err;

 /* Set system configuration */

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 if (priv->ieee->iw_mode == IW_MODE_ADHOC)
  cmd.host_command_parameters[0] |= IPW_CFG_IBSS_AUTO_START;

 cmd.host_command_parameters[0] |= IPW_CFG_IBSS_MASK |
     IPW_CFG_BSS_MASK | IPW_CFG_802_1x_ENABLE;

 if (!(priv->config & CFG_LONG_PREAMBLE))
  cmd.host_command_parameters[0] |= IPW_CFG_PREAMBLE_AUTO;

 err = ipw2100_get_ordinal(priv,
      IPW_ORD_EEPROM_IBSS_11B_CHANNELS,
      &ibss_mask, &len);
 if (err)
  ibss_mask = IPW_IBSS_11B_DEFAULT_MASK;

 cmd.host_command_parameters[1] = REG_CHANNEL_MASK;
 cmd.host_command_parameters[2] = REG_CHANNEL_MASK & ibss_mask;

 /* 11b only */
 /*cmd.host_command_parameters[0] |= DIVERSITY_ANTENNA_A;
 */

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
```

```
 return err;

/* If IPv6 is configured in the kernel then we don't want to filter out all
* of the multicast packets as IPv6 needs some. */
#if !defined(CONFIG_IPV6) && !defined(CONFIG_IPV6_MODULE)
 cmd.host_command = ADD_MULTICAST;
 cmd.host_command_sequence = 0;
 cmd.host_command_length = 0;

 ipw2100_hw_send_command(priv, &cmd);
#endif
 if (!batch_mode) {
  err = ipw2100_enable_adapter(priv);
  if (err)
   return err;
 }

 return 0;
}

static int ipw2100_set_tx_rates(struct ipw2100_priv *priv, u32 rate,
    int batch_mode)
{
 struct host_command cmd = {
  .host_command = BASIC_TX_RATES,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 cmd.host_command_parameters[0] = rate & TX_RATE_MASK;

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 /* Set BASIC TX Rate first */
 ipw2100_hw_send_command(priv, &cmd);

 /* Set TX Rate */
 cmd.host_command = TX_RATES;
 ipw2100_hw_send_command(priv,
 &cmd);

 /* Set MSDU TX Rate */
 cmd.host_command = MSDU_TX_RATES;
```

```
 ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode) {
  err = ipw2100_enable_adapter(priv);
  if (err)
   return err;
 }

 priv->tx_rates = rate;

 return 0;
}

static int ipw2100_set_power_mode(struct ipw2100_priv *priv, int power_level)
{
 struct host_command cmd = {
  .host_command = POWER_MODE,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 cmd.host_command_parameters[0] = power_level;

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  return err;

 if (power_level == IPW_POWER_MODE_CAM)
  priv->power_mode = IPW_POWER_LEVEL(priv->power_mode);
 else
  priv->power_mode = IPW_POWER_ENABLED | power_level;

#ifdef CONFIG_IPW2100_TX_POWER
 if (priv->port_type == IBSS && priv->adhoc_power != DFTL_IBSS_TX_POWER) {
  /* Set beacon interval */
  cmd.host_command = TX_POWER_INDEX;
  cmd.host_command_parameters[0] = (u32) priv->adhoc_power;

  err = ipw2100_hw_send_command(priv,
 &cmd);
  if (err)
   return err;
 }
#endif

 return 0;
}
```

```
static int ipw2100_set_rts_threshold(struct ipw2100_priv *priv, u32 threshold)
{
 struct host_command cmd = {
  .host_command = RTS_THRESHOLD,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 if (threshold & RTS_DISABLED)
  cmd.host_command_parameters[0] = MAX_RTS_THRESHOLD;
 else
  cmd.host_command_parameters[0] = threshold & ~RTS_DISABLED;

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  return err;

 priv->rts_threshold = threshold;

 return 0;
}

#if 0
int ipw2100_set_fragmentation_threshold(struct ipw2100_priv *priv,
     u32 threshold, int batch_mode)
{
 struct host_command cmd = {
  .host_command = FRAG_THRESHOLD,
  .host_command_sequence = 0,
  .host_command_length = 4,
  .host_command_parameters[0] = 0,
 };
 int err;

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 if (threshold == 0)
  threshold = DEFAULT_FRAG_THRESHOLD;
 else
 {
  threshold = max(threshold, MIN_FRAG_THRESHOLD);
  threshold = min(threshold, MAX_FRAG_THRESHOLD);
```

```
 }

 cmd.host_command_parameters[0] = threshold;

 IPW_DEBUG_HC("FRAG_THRESHOLD: %u\n", threshold);

 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode)
  ipw2100_enable_adapter(priv);

 if (!err)
  priv->frag_threshold = threshold;

 return err;
}
#endif

static int ipw2100_set_short_retry(struct ipw2100_priv *priv, u32 retry)
{
 struct host_command cmd = {
  .host_command = SHORT_RETRY_LIMIT,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 cmd.host_command_parameters[0] = retry;

 err = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  return err;

 priv->short_retry_limit = retry;

 return 0;
}

static int ipw2100_set_long_retry(struct ipw2100_priv *priv, u32 retry)
{
 struct host_command cmd = {
  .host_command = LONG_RETRY_LIMIT,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 cmd.host_command_parameters[0] = retry;
```

```
 err
 = ipw2100_hw_send_command(priv, &cmd);
 if (err)
  return err;

 priv->long_retry_limit = retry;

 return 0;
}

static int ipw2100_set_mandatory_bssid(struct ipw2100_priv *priv, u8 * bssid,
           int batch_mode)
{
 struct host_command cmd = {
  .host_command = MANDATORY_BSSID,
  .host_command_sequence = 0,
  .host_command_length = (bssid == NULL) ? 0 : ETH_ALEN
 };
 int err;

#ifdef CONFIG_IPW2100_DEBUG
 if (bssid != NULL)
  IPW_DEBUG_HC("MANDATORY_BSSID: %02X:%02X:%02X:%02X:%02X:%02X\n",
        bssid[0], bssid[1], bssid[2], bssid[3], bssid[4],
        bssid[5]);
 else
  IPW_DEBUG_HC("MANDATORY_BSSID: <clear>\n");
#endif
 /* if BSSID is empty then we disable mandatory bssid mode */
 if (bssid != NULL)
  memcpy(cmd.host_command_parameters, bssid, ETH_ALEN);

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode)
  ipw2100_enable_adapter(priv);

 return err;
}

static int ipw2100_disassociate_bssid(struct
```

```
 ipw2100_priv *priv)
{
 struct host_command cmd = {
  .host_command = DISASSOCIATION_BSSID,
  .host_command_sequence = 0,
  .host_command_length = ETH_ALEN
 };
 int err;
 int len;

 IPW_DEBUG_HC("DISASSOCIATION_BSSID\n");

 len = ETH_ALEN;
 /* The Firmware currently ignores the BSSID and just disassociates from
  * the currently associated AP -- but in the off chance that a future
  * firmware does use the BSSID provided here, we go ahead and try and
  * set it to the currently associated AP's BSSID */
 memcpy(cmd.host_command_parameters, priv->bssid, ETH_ALEN);

 err = ipw2100_hw_send_command(priv, &cmd);

 return err;
}

static int ipw2100_set_wpa_ie(struct ipw2100_priv *,
        struct ipw2100_wpa_assoc_frame *, int)
   __attribute__ ((unused));

static int ipw2100_set_wpa_ie(struct ipw2100_priv *priv,
        struct ipw2100_wpa_assoc_frame *wpa_frame,
        int batch_mode)
{
 struct host_command cmd = {
  .host_command = SET_WPA_IE,
  .host_command_sequence
 = 0,
  .host_command_length = sizeof(struct ipw2100_wpa_assoc_frame),
 };
 int err;

 IPW_DEBUG_HC("SET_WPA_IE\n");

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }
```

```
 memcpy(cmd.host_command_parameters, wpa_frame,
        sizeof(struct ipw2100_wpa_assoc_frame));

 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode) {
  if (ipw2100_enable_adapter(priv))
   err = -EIO;
 }

 return err;
}

struct security_info_params {
 u32 allowed_ciphers;
 u16 version;
 u8 auth_mode;
 u8 replay_counters_number;
 u8 unicast_using_group;
} __attribute__ ((packed));

static int ipw2100_set_security_information(struct ipw2100_priv *priv,
         int auth_mode,
         int security_level,
         int unicast_using_group,
         int batch_mode)
{
 struct host_command cmd = {
  .host_command = SET_SECURITY_INFORMATION,
  .host_command_sequence = 0,
  .host_command_length = sizeof(struct security_info_params)
 };
 struct security_info_params *security
 =
     (struct security_info_params *)&cmd.host_command_parameters;
 int err;
 memset(security, 0, sizeof(*security));

 /* If shared key AP authentication is turned on, then we need to
  * configure the firmware to try and use it.
  *
  * Actual data encryption/decryption is handled by the host. */
 security->auth_mode = auth_mode;
 security->unicast_using_group = unicast_using_group;

 switch (security_level) {
 default:
 case SEC_LEVEL_0:
```

```
 security->allowed_ciphers = IPW_NONE_CIPHER;
 break;
case SEC_LEVEL_1:
 security->allowed_ciphers = IPW_WEP40_CIPHER |
    IPW_WEP104_CIPHER;
 break;
case SEC_LEVEL_2:
 security->allowed_ciphers = IPW_WEP40_CIPHER |
    IPW_WEP104_CIPHER | IPW_TKIP_CIPHER;
 break;
case SEC_LEVEL_2_CKIP:
 security->allowed_ciphers = IPW_WEP40_CIPHER |
    IPW_WEP104_CIPHER | IPW_CKIP_CIPHER;
 break;
case SEC_LEVEL_3:
 security->allowed_ciphers = IPW_WEP40_CIPHER |
    IPW_WEP104_CIPHER | IPW_TKIP_CIPHER | IPW_CCMP_CIPHER;
 break;
}

IPW_DEBUG_HC

  ("SET_SECURITY_INFORMATION: auth:%d cipher:0x%02X (level %d)\n",
    security->auth_mode, security->allowed_ciphers, security_level);

security->replay_counters_number = 0;

if (!batch_mode) {
 err = ipw2100_disable_adapter(priv);
 if (err)
  return err;
}

err = ipw2100_hw_send_command(priv, &cmd);

if (!batch_mode)
 ipw2100_enable_adapter(priv);

return err;
}

static int ipw2100_set_tx_power(struct ipw2100_priv *priv, u32 tx_power)
{
 struct host_command cmd = {
  .host_command = TX_POWER_INDEX,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
```

```
 int err = 0;
 u32 tmp = tx_power;

 if (tx_power != IPW_TX_POWER_DEFAULT)
  tmp = (tx_power - IPW_TX_POWER_MIN_DBM) * 16 /
        (IPW_TX_POWER_MAX_DBM - IPW_TX_POWER_MIN_DBM);

 cmd.host_command_parameters[0] = tmp;

 if (priv->ieee->iw_mode == IW_MODE_ADHOC)
  err = ipw2100_hw_send_command(priv, &cmd);
 if (!err)
  priv->tx_power = tx_power;

 return 0;
}

static int ipw2100_set_ibss_beacon_interval(struct
 ipw2100_priv *priv,
         u32 interval, int batch_mode)
{
 struct host_command cmd = {
  .host_command = BEACON_INTERVAL,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 cmd.host_command_parameters[0] = interval;

 IPW_DEBUG_INFO("enter\n");

 if (priv->ieee->iw_mode == IW_MODE_ADHOC) {
  if (!batch_mode) {
   err = ipw2100_disable_adapter(priv);
   if (err)
    return err;
  }

  ipw2100_hw_send_command(priv, &cmd);

  if (!batch_mode) {
   err = ipw2100_enable_adapter(priv);
   if (err)
    return err;
  }
 }
```

```
 IPW_DEBUG_INFO("exit\n");

 return 0;
}

void ipw2100_queues_initialize(struct ipw2100_priv *priv)
{
 ipw2100_tx_initialize(priv);
 ipw2100_rx_initialize(priv);
 ipw2100_msg_initialize(priv);
}

void ipw2100_queues_free(struct ipw2100_priv *priv)
{
 ipw2100_tx_free(priv);
 ipw2100_rx_free(priv);
 ipw2100_msg_free(priv);
}

int ipw2100_queues_allocate(struct ipw2100_priv *priv)
{
 if (ipw2100_tx_allocate(priv) ||
     ipw2100_rx_allocate(priv) ||
 ipw2100_msg_allocate(priv))
  goto fail;

 return 0;

     fail:
 ipw2100_tx_free(priv);
 ipw2100_rx_free(priv);
 ipw2100_msg_free(priv);
 return -ENOMEM;
}

#define IPW_PRIVACY_CAPABLE 0x0008

static int ipw2100_set_wep_flags(struct ipw2100_priv *priv, u32 flags,
     int batch_mode)
{
 struct host_command cmd = {
  .host_command = WEP_FLAGS,
  .host_command_sequence = 0,
  .host_command_length = 4
 };
 int err;

 cmd.host_command_parameters[0] = flags;
```

```
 IPW_DEBUG_HC("WEP_FLAGS: flags = 0x%08X\n", flags);

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err) {
   printk(KERN_ERR DRV_NAME
          ": %s: Could not disable adapter %d\n",
          priv->net_dev->name, err);
   return err;
  }
 }

 /* send cmd to firmware */
 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode)
  ipw2100_enable_adapter(priv);

 return err;
}

struct ipw2100_wep_key {
 u8 idx;
 u8 len;
 u8 key[13];
};

/* Macros to ease up priting WEP keys */
#define WEP_FMT_64  "%02X%02X%02X%02X-%02X"
#define
 WEP_FMT_128 "%02X%02X%02X%02X-%02X%02X%02X%02X-%02X%02X%02X"
#define WEP_STR_64(x) x[0],x[1],x[2],x[3],x[4]
#define WEP_STR_128(x) x[0],x[1],x[2],x[3],x[4],x[5],x[6],x[7],x[8],x[9],x[10]

/**
* Set a the wep key
*
* @priv: struct to work on
* @idx: index of the key we want to set
* @key: ptr to the key data to set
* @len: length of the buffer at @key
* @batch_mode: FIXME perform the operation in batch mode, not
*              disabling the device.
*
* @returns 0 if OK, < 0 errno code on error.
*
* Fill out a command structure with the new wep key, length an
```

```
* index and send it down the wire.
*/
static int ipw2100_set_key(struct ipw2100_priv *priv,
      int idx, char *key, int len, int batch_mode)
{
 int keylen = len ? (len <= 5 ? 5 : 13) : 0;
 struct host_command cmd = {
  .host_command = WEP_KEY_INFO,
  .host_command_sequence = 0,
  .host_command_length = sizeof(struct ipw2100_wep_key),
 };
 struct ipw2100_wep_key *wep_key = (void *)cmd.host_command_parameters;
 int
 err;

 IPW_DEBUG_HC("WEP_KEY_INFO: index = %d, len = %d/%d\n",
       idx, keylen, len);

 /* NOTE: We don't check cached values in case the firmware was reset
  * or some other problem is occurring.  If the user is setting the key,
  * then we push the change */

 wep_key->idx = idx;
 wep_key->len = keylen;

 if (keylen) {
  memcpy(wep_key->key, key, len);
  memset(wep_key->key + len, 0, keylen - len);
 }

 /* Will be optimized out on debug not being configured in */
 if (keylen == 0)
  IPW_DEBUG_WEP("%s: Clearing key %d\n",
        priv->net_dev->name, wep_key->idx);
 else if (keylen == 5)
  IPW_DEBUG_WEP("%s: idx: %d, len: %d key: " WEP_FMT_64 "\n",
        priv->net_dev->name, wep_key->idx, wep_key->len,
        WEP_STR_64(wep_key->key));
 else
  IPW_DEBUG_WEP("%s: idx: %d, len: %d key: " WEP_FMT_128
        "\n",
        priv->net_dev->name, wep_key->idx, wep_key->len,
        WEP_STR_128(wep_key->key));

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  /*
```

```
 FIXME: IPG: shouldn't this prink be in _disable_adapter()? */
  if (err) {
   printk(KERN_ERR DRV_NAME
          ": %s: Could not disable adapter %d\n",
          priv->net_dev->name, err);
   return err;
  }
 }

 /* send cmd to firmware */
 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode) {
  int err2 = ipw2100_enable_adapter(priv);
  if (err == 0)
   err = err2;
 }
 return err;
}

static int ipw2100_set_key_index(struct ipw2100_priv *priv,
     int idx, int batch_mode)
{
 struct host_command cmd = {
  .host_command = WEP_KEY_INDEX,
  .host_command_sequence = 0,
  .host_command_length = 4,
  .host_command_parameters = {idx},
 };
 int err;

 IPW_DEBUG_HC("WEP_KEY_INDEX: index = %d\n", idx);

 if (idx < 0 || idx > 3)
  return -EINVAL;

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err) {
   printk(KERN_ERR DRV_NAME
          ": %s: Could not disable adapter %d\n",
          priv->net_dev->name, err);
   return err;
  }
 }

 /* send cmd to firmware
```

```
 */
 err = ipw2100_hw_send_command(priv, &cmd);

 if (!batch_mode)
  ipw2100_enable_adapter(priv);

 return err;
}

static int ipw2100_configure_security(struct ipw2100_priv *priv, int batch_mode)
{
 int i, err, auth_mode, sec_level, use_group;

 if (!(priv->status & STATUS_RUNNING))
  return 0;

 if (!batch_mode) {
  err = ipw2100_disable_adapter(priv);
  if (err)
   return err;
 }

 if (!priv->ieee->sec.enabled) {
  err =
      ipw2100_set_security_information(priv, IPW_AUTH_OPEN,
          SEC_LEVEL_0, 0, 1);
 } else {
  auth_mode = IPW_AUTH_OPEN;
  if (priv->ieee->sec.flags & SEC_AUTH_MODE) {
   if (priv->ieee->sec.auth_mode == WLAN_AUTH_SHARED_KEY)
    auth_mode = IPW_AUTH_SHARED;
   else if (priv->ieee->sec.auth_mode == WLAN_AUTH_LEAP)
    auth_mode = IPW_AUTH_LEAP_CISCO_ID;
  }

  sec_level = SEC_LEVEL_0;
  if (priv->ieee->sec.flags & SEC_LEVEL)
   sec_level = priv->ieee->sec.level;

  use_group = 0;
  if (priv->ieee->sec.flags & SEC_UNICAST_GROUP)
   use_group = priv->ieee->sec.unicast_uses_group;

  err
 =
      ipw2100_set_security_information(priv, auth_mode, sec_level,
          use_group, 1);
 }
```

```
 if (err)
  goto exit;

 if (priv->ieee->sec.enabled) {
  for (i = 0; i < 4; i++) {
   if (!(priv->ieee->sec.flags & (1 << i))) {
    memset(priv->ieee->sec.keys[i], 0, WEP_KEY_LEN);
    priv->ieee->sec.key_sizes[i] = 0;
   } else {
    err = ipw2100_set_key(priv, i,
          priv->ieee->sec.keys[i],
          priv->ieee->sec.
          key_sizes[i], 1);
    if (err)
     goto exit;
   }
  }

  ipw2100_set_key_index(priv, priv->ieee->tx_keyidx, 1);
 }

 /* Always enable privacy so the Host can filter WEP packets if
  * encrypted data is sent up */
 err =
     ipw2100_set_wep_flags(priv,
      priv->ieee->sec.
      enabled ? IPW_PRIVACY_CAPABLE : 0, 1);
 if (err)
  goto exit;

 priv->status &= ~STATUS_SECURITY_UPDATED;

     exit:
 if (!batch_mode)
  ipw2100_enable_adapter(priv);

 return err;
}

static void ipw2100_security_work(struct ipw2100_priv
 *priv)
{
 /* If we happen to have reconnected before we get a chance to
  * process this, then update the security settings--which causes
  * a disassociation to occur */
 if (!(priv->status & STATUS_ASSOCIATED) &&
     priv->status & STATUS_SECURITY_UPDATED)
```

```
 ipw2100_configure_security(priv, 0);
}

static void shim__set_security(struct net_device *dev,
        struct ieee80211_security *sec)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int i, force_update = 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED))
  goto done;

 for (i = 0; i < 4; i++) {
  if (sec->flags & (1 << i)) {
   priv->ieee->sec.key_sizes[i] = sec->key_sizes[i];
   if (sec->key_sizes[i] == 0)
    priv->ieee->sec.flags &= ~(1 << i);
   else
    memcpy(priv->ieee->sec.keys[i], sec->keys[i],
          sec->key_sizes[i]);
   if (sec->level == SEC_LEVEL_1) {
    priv->ieee->sec.flags |= (1 << i);
    priv->status |= STATUS_SECURITY_UPDATED;
   } else
    priv->ieee->sec.flags &=
~(1 << i);
  }
 }

 if ((sec->flags & SEC_ACTIVE_KEY) &&
     priv->ieee->sec.active_key != sec->active_key) {
  if (sec->active_key <= 3) {
   priv->ieee->sec.active_key = sec->active_key;
   priv->ieee->sec.flags |= SEC_ACTIVE_KEY;
  } else
   priv->ieee->sec.flags &= ~SEC_ACTIVE_KEY;

  priv->status |= STATUS_SECURITY_UPDATED;
 }

 if ((sec->flags & SEC_AUTH_MODE) &&
     (priv->ieee->sec.auth_mode != sec->auth_mode)) {
  priv->ieee->sec.auth_mode = sec->auth_mode;
  priv->ieee->sec.flags |= SEC_AUTH_MODE;
  priv->status |= STATUS_SECURITY_UPDATED;
 }
```

```
 if (sec->flags & SEC_ENABLED && priv->ieee->sec.enabled != sec->enabled) {
  priv->ieee->sec.flags |= SEC_ENABLED;
  priv->ieee->sec.enabled = sec->enabled;
  priv->status |= STATUS_SECURITY_UPDATED;
  force_update = 1;
 }

 if (sec->flags & SEC_ENCRYPT)
  priv->ieee->sec.encrypt = sec->encrypt;

 if (sec->flags & SEC_LEVEL && priv->ieee->sec.level != sec->level) {
  priv->ieee->sec.level = sec->level;
  priv->ieee->sec.flags |= SEC_LEVEL;
  priv->status
 |= STATUS_SECURITY_UPDATED;
 }

 IPW_DEBUG_WEP("Security flags: %c %c%c%c%c %c%c%c%c\n",
        priv->ieee->sec.flags & (1 << 8) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 7) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 6) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 5) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 4) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 3) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 2) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 1) ? '1' : '0',
        priv->ieee->sec.flags & (1 << 0) ? '1' : '0');

/* As a temporary work around to enable WPA until we figure out why
* wpa_supplicant toggles the security capability of the driver, which
* forces a disassocation with force_update...
*
* if (force_update || !(priv->status & STATUS_ASSOCIATED))*/
 if (!(priv->status & (STATUS_ASSOCIATED | STATUS_ASSOCIATING)))
  ipw2100_configure_security(priv, 0);
     done:
 mutex_unlock(&priv->action_mutex);
}

static
 int ipw2100_adapter_setup(struct ipw2100_priv *priv)
{
 int err;
 int batch_mode = 1;
 u8 *bssid;

 IPW_DEBUG_INFO("enter\n");
```

```
 err = ipw2100_disable_adapter(priv);
 if (err)
  return err;
#ifdef CONFIG_IPW2100_MONITOR
 if (priv->ieee->iw_mode == IW_MODE_MONITOR) {
  err = ipw2100_set_channel(priv, priv->channel, batch_mode);
  if (err)
   return err;

  IPW_DEBUG_INFO("exit\n");

  return 0;
 }
#endif    /* CONFIG_IPW2100_MONITOR */

 err = ipw2100_read_mac_address(priv);
 if (err)
  return -EIO;

 err = ipw2100_set_mac_address(priv, batch_mode);
 if (err)
  return err;

 err = ipw2100_set_port_type(priv, priv->ieee->iw_mode, batch_mode);
 if (err)
  return err;

 if (priv->ieee->iw_mode == IW_MODE_ADHOC) {
  err = ipw2100_set_channel(priv, priv->channel, batch_mode);
  if (err)
   return err;
 }

 err = ipw2100_system_config(priv, batch_mode);
 if (err)
  return err;

 err = ipw2100_set_tx_rates(priv, priv->tx_rates, batch_mode);
 if (err)
  return err;

 /* Default to power
 mode OFF */
 err = ipw2100_set_power_mode(priv, IPW_POWER_MODE_CAM);
 if (err)
  return err;

 err = ipw2100_set_rts_threshold(priv, priv->rts_threshold);
```

```
if (err)
 return err;

if (priv->config & CFG_STATIC_BSSID)
 bssid = priv->bssid;
else
 bssid = NULL;
err = ipw2100_set_mandatory_bssid(priv, bssid, batch_mode);
if (err)
 return err;

if (priv->config & CFG_STATIC_ESSID)
 err = ipw2100_set_essid(priv, priv->essid, priv->essid_len,
    batch_mode);
else
 err = ipw2100_set_essid(priv, NULL, 0, batch_mode);
if (err)
 return err;

err = ipw2100_configure_security(priv, batch_mode);
if (err)
 return err;

if (priv->ieee->iw_mode == IW_MODE_ADHOC) {
 err =
     ipw2100_set_ibss_beacon_interval(priv,
         priv->beacon_interval,
         batch_mode);
 if (err)
  return err;

 err = ipw2100_set_tx_power(priv, priv->tx_power);
 if (err)
  return err;
}

/*
   err = ipw2100_set_fragmentation_threshold(
   priv, priv->frag_threshold, batch_mode);

if (err)
   return err;
 */

IPW_DEBUG_INFO("exit\n");

return 0;
}
```

```
/*********************************************************************
*
* EXTERNALLY CALLED METHODS
*
*********************************************************************/

/* This method is called by the network layer -- not to be confused with
* ipw2100_set_mac_address() declared above called by this driver (and this
* method as well) to talk to the firmware */
static int ipw2100_set_address(struct net_device *dev, void *p)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct sockaddr *addr = p;
 int err = 0;

 if (!is_valid_ether_addr(addr->sa_data))
  return -EADDRNOTAVAIL;

 mutex_lock(&priv->action_mutex);

 priv->config |= CFG_CUSTOM_MAC;
 memcpy(priv->mac_addr, addr->sa_data, ETH_ALEN);

 err = ipw2100_set_mac_address(priv, 0);
 if (err)
  goto done;

 priv->reset_backoff = 0;
 mutex_unlock(&priv->action_mutex);
 ipw2100_reset_adapter(priv);
 return 0;

    done:
 mutex_unlock(&priv->action_mutex);
 return
 err;
}

static int ipw2100_open(struct net_device *dev)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 unsigned long flags;
 IPW_DEBUG_INFO("dev->open\n");

 spin_lock_irqsave(&priv->low_lock, flags);
 if (priv->status & STATUS_ASSOCIATED) {
  netif_carrier_on(dev);
```

```
 netif_start_queue(dev);
 }
 spin_unlock_irqrestore(&priv->low_lock, flags);

 return 0;
}

static int ipw2100_close(struct net_device *dev)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 unsigned long flags;
 struct list_head *element;
 struct ipw2100_tx_packet *packet;

 IPW_DEBUG_INFO("enter\n");

 spin_lock_irqsave(&priv->low_lock, flags);

 if (priv->status & STATUS_ASSOCIATED)
  netif_carrier_off(dev);
 netif_stop_queue(dev);

 /* Flush the TX queue ... */
 while (!list_empty(&priv->tx_pend_list)) {
  element = priv->tx_pend_list.next;
  packet = list_entry(element, struct ipw2100_tx_packet, list);

  list_del(element);
  DEC_STAT(&priv->tx_pend_stat);

  ieee80211_txb_free(packet->info.d_struct.txb);
  packet->info.d_struct.txb
 = NULL;

  list_add_tail(element, &priv->tx_free_list);
  INC_STAT(&priv->tx_free_stat);
 }
 spin_unlock_irqrestore(&priv->low_lock, flags);

 IPW_DEBUG_INFO("exit\n");

 return 0;
}

/*
* TODO:  Fix this function... its just wrong
*/
static void ipw2100_tx_timeout(struct net_device *dev)
```

```
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);

 priv->ieee->stats.tx_errors++;

#ifdef CONFIG_IPW2100_MONITOR
 if (priv->ieee->iw_mode == IW_MODE_MONITOR)
  return;
#endif

 IPW_DEBUG_INFO("%s: TX timed out.  Scheduling firmware restart.\n",
        dev->name);
 schedule_reset(priv);
}

/*
* TODO: reimplement it so that it reads statistics
*       from the adapter using ordinal tables
*       instead of/in addition to collecting them
*       in the driver
*/
static struct net_device_stats *ipw2100_stats(struct net_device *dev)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);

 return &priv->ieee->stats;
}

static int ipw2100_wpa_enable(struct
 ipw2100_priv *priv, int value)
{
 /* This is called when wpa_supplicant loads and closes the driver
  * interface. */
 priv->ieee->wpa_enabled = value;
 return 0;
}

static int ipw2100_wpa_set_auth_algs(struct ipw2100_priv *priv, int value)
{

 struct ieee80211_device *ieee = priv->ieee;
 struct ieee80211_security sec = {
  .flags = SEC_AUTH_MODE,
 };
 int ret = 0;

 if (value & IW_AUTH_ALG_SHARED_KEY) {
  sec.auth_mode = WLAN_AUTH_SHARED_KEY;
```

```
 ieee->open_wep = 0;
 } else if (value & IW_AUTH_ALG_OPEN_SYSTEM) {
 sec.auth_mode = WLAN_AUTH_OPEN;
 ieee->open_wep = 1;
 } else if (value & IW_AUTH_ALG_LEAP) {
 sec.auth_mode = WLAN_AUTH_LEAP;
 ieee->open_wep = 1;
 } else
 return -EINVAL;

 if (ieee->set_security)
 ieee->set_security(ieee->dev, &sec);
 else
 ret = -EOPNOTSUPP;

 return ret;
}

static void ipw2100_wpa_assoc_frame(struct ipw2100_priv *priv,
       char *wpa_ie, int wpa_ie_len)
{

 struct ipw2100_wpa_assoc_frame frame;

 frame.fixed_ie_mask = 0;

 /* copy WPA IE
 */
 memcpy(frame.var_ie, wpa_ie, wpa_ie_len);
 frame.var_ie_len = wpa_ie_len;

 /* make sure WPA is enabled */
 ipw2100_wpa_enable(priv, 1);
 ipw2100_set_wpa_ie(priv, &frame, 0);
}

static void ipw_ethtool_get_drvinfo(struct net_device *dev,
       struct ethtool_drvinfo *info)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 char fw_ver[64], ucode_ver[64];

 strcpy(info->driver, DRV_NAME);
 strcpy(info->version, DRV_VERSION);

 ipw2100_get_fwversion(priv, fw_ver, sizeof(fw_ver));
 ipw2100_get_ucodeversion(priv, ucode_ver, sizeof(ucode_ver));
```

```
 snprintf(info->fw_version, sizeof(info->fw_version), "%s:%d:%s",
   fw_ver, priv->eeprom_version, ucode_ver);

 strcpy(info->bus_info, pci_name(priv->pci_dev));
}

static u32 ipw2100_ethtool_get_link(struct net_device *dev)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 return (priv->status & STATUS_ASSOCIATED) ? 1 : 0;
}

static const struct ethtool_ops ipw2100_ethtool_ops = {
 .get_link = ipw2100_ethtool_get_link,
 .get_drvinfo = ipw_ethtool_get_drvinfo,
};

static
 void ipw2100_hang_check(void *adapter)
{
 struct ipw2100_priv *priv = adapter;
 unsigned long flags;
 u32 rtc = 0xa5a5a5a5;
 u32 len = sizeof(rtc);
 int restart = 0;

 spin_lock_irqsave(&priv->low_lock, flags);

 if (priv->fatal_error != 0) {
  /* If fatal_error is set then we need to restart */
  IPW_DEBUG_INFO("%s: Hardware fatal error detected.\n",
         priv->net_dev->name);

  restart = 1;
 } else if (ipw2100_get_ordinal(priv, IPW_ORD_RTC_TIME, &rtc, &len) ||
     (rtc == priv->last_rtc)) {
  /* Check if firmware is hung */
  IPW_DEBUG_INFO("%s: Firmware RTC stalled.\n",
         priv->net_dev->name);

  restart = 1;
 }

 if (restart) {
  /* Kill timer */
  priv->stop_hang_check = 1;
  priv->hangs++;
```

```
 /* Restart the NIC */
 schedule_reset(priv);
 }

 priv->last_rtc = rtc;

 if (!priv->stop_hang_check)
  queue_delayed_work(priv->workqueue, &priv->hang_check, HZ / 2);

 spin_unlock_irqrestore(&priv->low_lock, flags);
}

static void ipw2100_rf_kill(void
 *adapter)
{
 struct ipw2100_priv *priv = adapter;
 unsigned long flags;

 spin_lock_irqsave(&priv->low_lock, flags);

 if (rf_kill_active(priv)) {
  IPW_DEBUG_RF_KILL("RF Kill active, rescheduling GPIO check\n");
  if (!priv->stop_rf_kill)
   queue_delayed_work(priv->workqueue, &priv->rf_kill, HZ);
  goto exit_unlock;
 }

 /* RF Kill is now disabled, so bring the device back up */

 if (!(priv->status & STATUS_RF_KILL_MASK)) {
  IPW_DEBUG_RF_KILL("HW RF Kill no longer active, restarting "
      "device\n");
  schedule_reset(priv);
 } else
  IPW_DEBUG_RF_KILL("HW RF Kill deactivated.  SW RF Kill still "
      "enabled\n");

     exit_unlock:
 spin_unlock_irqrestore(&priv->low_lock, flags);
}

static void ipw2100_irq_tasklet(struct ipw2100_priv *priv);

/* Look into using netdev destructor to shutdown ieee80211? */

static struct net_device *ipw2100_alloc_device(struct pci_dev *pci_dev,
            void __iomem * base_addr,
            unsigned long mem_start,
```

```
        unsigned
long mem_len)
{
struct ipw2100_priv *priv;
struct net_device *dev;

dev = alloc_ieee80211(sizeof(struct ipw2100_priv));
if (!dev)
 return NULL;
priv = ieee80211_priv(dev);
priv->ieee = netdev_priv(dev);
priv->pci_dev = pci_dev;
priv->net_dev = dev;

priv->ieee->hard_start_xmit = ipw2100_tx;
priv->ieee->set_security = shim__set_security;

priv->ieee->perfect_rssi = -20;
priv->ieee->worst_rssi = -85;

dev->open = ipw2100_open;
dev->stop = ipw2100_close;
dev->init = ipw2100_net_init;
dev->get_stats = ipw2100_stats;
dev->ethtool_ops = &ipw2100_ethtool_ops;
dev->tx_timeout = ipw2100_tx_timeout;
dev->wireless_handlers = &ipw2100_wx_handler_def;
priv->wireless_data.ieee80211 = priv->ieee;
dev->wireless_data = &priv->wireless_data;
dev->set_mac_address = ipw2100_set_address;
dev->watchdog_timeo = 3 * HZ;
dev->irq = 0;

dev->base_addr = (unsigned long)base_addr;
dev->mem_start = mem_start;
dev->mem_end = dev->mem_start + mem_len - 1;

/* NOTE: We don't use the
wireless_handlers hook
 * in dev as the system will start throwing WX requests
 * to us before we're actually initialized and it just
 * ends up causing problems.  So, we just handle
 * the WX extensions through the ipw2100_ioctl interface */

/* memset() puts everything to 0, so we only have explicitely set
 * those values that need to be something else */

/* If power management is turned on, default to AUTO mode */
```

```
 priv->power_mode = IPW_POWER_AUTO;

#ifdef CONFIG_IPW2100_MONITOR
 priv->config |= CFG_CRC_CHECK;
#endif
 priv->ieee->wpa_enabled = 0;
 priv->ieee->drop_unencrypted = 0;
 priv->ieee->privacy_invoked = 0;
 priv->ieee->ieee802_1x = 1;

 /* Set module parameters */
 switch (mode) {
 case 1:
  priv->ieee->iw_mode = IW_MODE_ADHOC;
  break;
#ifdef CONFIG_IPW2100_MONITOR
 case 2:
  priv->ieee->iw_mode = IW_MODE_MONITOR;
  break;
#endif
 default:
 case 0:
  priv->ieee->iw_mode = IW_MODE_INFRA;
  break;
 }

 if (disable == 1)
  priv->status |= STATUS_RF_KILL_SW;

 if (channel
 != 0 &&
     ((channel >= REG_MIN_CHANNEL) && (channel <= REG_MAX_CHANNEL))) {
  priv->config |= CFG_STATIC_CHANNEL;
  priv->channel = channel;
 }

 if (associate)
  priv->config |= CFG_ASSOCIATE;

 priv->beacon_interval = DEFAULT_BEACON_INTERVAL;
 priv->short_retry_limit = DEFAULT_SHORT_RETRY_LIMIT;
 priv->long_retry_limit = DEFAULT_LONG_RETRY_LIMIT;
 priv->rts_threshold = DEFAULT_RTS_THRESHOLD | RTS_DISABLED;
 priv->frag_threshold = DEFAULT_FTS | FRAG_DISABLED;
 priv->tx_power = IPW_TX_POWER_DEFAULT;
 priv->tx_rates = DEFAULT_TX_RATES;

 strcpy(priv->nick, "ipw2100");
```

```
 spin_lock_init(&priv->low_lock);
 mutex_init(&priv->action_mutex);
 mutex_init(&priv->adapter_mutex);

 init_waitqueue_head(&priv->wait_command_queue);

 netif_carrier_off(dev);

 INIT_LIST_HEAD(&priv->msg_free_list);
 INIT_LIST_HEAD(&priv->msg_pend_list);
 INIT_STAT(&priv->msg_free_stat);
 INIT_STAT(&priv->msg_pend_stat);

 INIT_LIST_HEAD(&priv->tx_free_list);
 INIT_LIST_HEAD(&priv->tx_pend_list);
 INIT_STAT(&priv->tx_free_stat);
 INIT_STAT(&priv->tx_pend_stat);

 INIT_LIST_HEAD(&priv->fw_pend_list);
 INIT_STAT(&priv->fw_pend_stat);

 priv->workqueue
 = create_workqueue(DRV_NAME);

 INIT_WORK(&priv->reset_work,
   (void (*)(void *))ipw2100_reset_adapter, priv);
 INIT_WORK(&priv->security_work,
   (void (*)(void *))ipw2100_security_work, priv);
 INIT_WORK(&priv->wx_event_work,
   (void (*)(void *))ipw2100_wx_event_work, priv);
 INIT_WORK(&priv->hang_check, ipw2100_hang_check, priv);
 INIT_WORK(&priv->rf_kill, ipw2100_rf_kill, priv);

 tasklet_init(&priv->irq_tasklet, (void (*)(unsigned long))
      ipw2100_irq_tasklet, (unsigned long)priv);

 /* NOTE:  We do not start the deferred work for status checks yet */
 priv->stop_rf_kill = 1;
 priv->stop_hang_check = 1;

 return dev;
}

static int ipw2100_pci_init_one(struct pci_dev *pci_dev,
   const struct pci_device_id *ent)
{
 unsigned long mem_start, mem_len, mem_flags;
```

```
void __iomem *base_addr = NULL;
struct net_device *dev = NULL;
struct
ipw2100_priv *priv = NULL;
int err = 0;
int registered = 0;
u32 val;

IPW_DEBUG_INFO("enter\n");

mem_start = pci_resource_start(pci_dev, 0);
mem_len = pci_resource_len(pci_dev, 0);
mem_flags = pci_resource_flags(pci_dev, 0);

if ((mem_flags & IORESOURCE_MEM) != IORESOURCE_MEM) {
 IPW_DEBUG_INFO("weird - resource type is not memory\n");
 err = -ENODEV;
 goto fail;
}

base_addr = ioremap_nocache(mem_start, mem_len);
if (!base_addr) {
 printk(KERN_WARNING DRV_NAME
        "Error calling ioremap_nocache.\n");
 err = -EIO;
 goto fail;
}

/* allocate and initialize our net_device */
dev = ipw2100_alloc_device(pci_dev, base_addr, mem_start, mem_len);
if (!dev) {
 printk(KERN_WARNING DRV_NAME
        "Error calling ipw2100_alloc_device.\n");
 err = -ENOMEM;
 goto fail;
}

/* set up PCI mappings for device */
err = pci_enable_device(pci_dev);
if (err) {
 printk(KERN_WARNING DRV_NAME
        "Error calling pci_enable_device.\n");
 return err;
}

priv = ieee80211_priv(dev);

pci_set_master(pci_dev);
```

```
pci_set_drvdata(pci_dev,
priv);

err = pci_set_dma_mask(pci_dev, DMA_32BIT_MASK);
if (err) {
 printk(KERN_WARNING DRV_NAME
        "Error calling pci_set_dma_mask.\n");
 pci_disable_device(pci_dev);
 return err;
}

err = pci_request_regions(pci_dev, DRV_NAME);
if (err) {
 printk(KERN_WARNING DRV_NAME
        "Error calling pci_request_regions.\n");
 pci_disable_device(pci_dev);
 return err;
}

/* We disable the RETRY_TIMEOUT register (0x41) to keep
 * PCI Tx retries from interfering with C3 CPU state */
pci_read_config_dword(pci_dev, 0x40, &val);
if ((val & 0x0000ff00) != 0)
 pci_write_config_dword(pci_dev, 0x40, val & 0xffff00ff);

pci_set_power_state(pci_dev, PCI_D0);

if (!ipw2100_hw_is_adapter_in_system(dev)) {
 printk(KERN_WARNING DRV_NAME
        "Device not found via register read.\n");
 err = -ENODEV;
 goto fail;
}

SET_NETDEV_DEV(dev, &pci_dev->dev);

/* Force interrupts to be shut off on the device */
priv->status
|= STATUS_INT_ENABLED;
ipw2100_disable_interrupts(priv);

/* Allocate and initialize the Tx/Rx queues and lists */
if (ipw2100_queues_allocate(priv)) {
 printk(KERN_WARNING DRV_NAME
        "Error calilng ipw2100_queues_allocate.\n");
 err = -ENOMEM;
 goto fail;
}
```

```
ipw2100_queues_initialize(priv);

err = request_irq(pci_dev->irq,
   ipw2100_interrupt, IRQF_SHARED, dev->name, priv);
if (err) {
 printk(KERN_WARNING DRV_NAME
       "Error calling request_irq: %d.\n", pci_dev->irq);
 goto fail;
}
dev->irq = pci_dev->irq;

IPW_DEBUG_INFO("Attempting to register device...\n");

SET_MODULE_OWNER(dev);

printk(KERN_INFO DRV_NAME
      ": Detected Intel PRO/Wireless 2100 Network Connection\n");

/* Bring up the interface.  Pre 0.46, after we registered the
 * network device we would call ipw2100_up.  This introduced a race
 * condition with newer hotplug configurations (network was coming
 * up and making calls before the device was initialized).
 *
 * If
we called ipw2100_up before we registered the device, then the
 * device name wasn't registered.  So, we instead use the net_dev->init
 * member to call a function that then just turns and calls ipw2100_up.
 * net_dev->init is called after name allocation but before the
 * notifier chain is called */
err = register_netdev(dev);
if (err) {
 printk(KERN_WARNING DRV_NAME
       "Error calling register_netdev.\n");
 goto fail;
}

mutex_lock(&priv->action_mutex);
registered = 1;

IPW_DEBUG_INFO("%s: Bound to %s\n", dev->name, pci_name(pci_dev));

/* perform this after register_netdev so that dev->name is set */
err = sysfs_create_group(&pci_dev->dev.kobj, &ipw2100_attribute_group);
if (err)
 goto fail_unlock;

/* If the RF Kill switch is disabled, go ahead and complete the
 * startup sequence */
```

```
if (!(priv->status & STATUS_RF_KILL_MASK)) {
 /* Enable the adapter - sends HOST_COMPLETE */
 if (ipw2100_enable_adapter(priv)) {
  printk(KERN_WARNING DRV_NAME
         ":
%s: failed in call to enable adapter.\n",
         priv->net_dev->name);
  ipw2100_hw_stop_adapter(priv);
  err = -EIO;
  goto fail_unlock;
 }

 /* Start a scan . . . */
 ipw2100_set_scan_options(priv);
 ipw2100_start_scan(priv);
}

IPW_DEBUG_INFO("exit\n");

priv->status |= STATUS_INITIALIZED;

mutex_unlock(&priv->action_mutex);

return 0;

    fail_unlock:
mutex_unlock(&priv->action_mutex);

    fail:
if (dev) {
 if (registered)
  unregister_netdev(dev);

 ipw2100_hw_stop_adapter(priv);

 ipw2100_disable_interrupts(priv);

 if (dev->irq)
  free_irq(dev->irq, priv);

 ipw2100_kill_workqueue(priv);

 /* These are safe to call even if they weren't allocated */
 ipw2100_queues_free(priv);
 sysfs_remove_group(&pci_dev->dev.kobj,
      &ipw2100_attribute_group);

 free_ieee80211(dev);
```

```
 pci_set_drvdata(pci_dev, NULL);
 }

 if (base_addr)
  iounmap(base_addr);

 pci_release_regions(pci_dev);
 pci_disable_device(pci_dev);

 return err;
}

static void __devexit
 ipw2100_pci_remove_one(struct pci_dev *pci_dev)
{
 struct ipw2100_priv *priv = pci_get_drvdata(pci_dev);
 struct net_device *dev;

 if (priv) {
  mutex_lock(&priv->action_mutex);

  priv->status &= ~STATUS_INITIALIZED;

  dev = priv->net_dev;
  sysfs_remove_group(&pci_dev->dev.kobj,
      &ipw2100_attribute_group);

#ifdef CONFIG_PM
  if (ipw2100_firmware.version)
   ipw2100_release_firmware(priv, &ipw2100_firmware);
#endif
  /* Take down the hardware */
  ipw2100_down(priv);

  /* Release the mutex so that the network subsystem can
   * complete any needed calls into the driver... */
  mutex_unlock(&priv->action_mutex);

  /* Unregister the device first - this results in close()
   * being called if the device is open.  If we free storage
   * first, then close() will crash. */
  unregister_netdev(dev);

  /* ipw2100_down will ensure that there is no more pending work
   * in the workqueue's, so we can safely remove them now. */
  ipw2100_kill_workqueue(priv);

  ipw2100_queues_free(priv);
```

```
 /*
Free potential debugging firmware snapshot */
 ipw2100_snapshot_free(priv);

 if (dev->irq)
  free_irq(dev->irq, priv);

 if (dev->base_addr)
  iounmap((void __iomem *)dev->base_addr);

 free_ieee80211(dev);
}

 pci_release_regions(pci_dev);
 pci_disable_device(pci_dev);

 IPW_DEBUG_INFO("exit\n");
}

#ifdef CONFIG_PM
static int ipw2100_suspend(struct pci_dev *pci_dev, pm_message_t state)
{
 struct ipw2100_priv *priv = pci_get_drvdata(pci_dev);
 struct net_device *dev = priv->net_dev;

 IPW_DEBUG_INFO("%s: Going into suspend...\n", dev->name);

 mutex_lock(&priv->action_mutex);
 if (priv->status & STATUS_INITIALIZED) {
  /* Take down the device; powers it off, etc. */
  ipw2100_down(priv);
 }

 /* Remove the PRESENT state of the device */
 netif_device_detach(dev);

 pci_save_state(pci_dev);
 pci_disable_device(pci_dev);
 pci_set_power_state(pci_dev, PCI_D3hot);

 mutex_unlock(&priv->action_mutex);

 return 0;
}

static int ipw2100_resume(struct pci_dev *pci_dev)
{
```

```
 struct
 ipw2100_priv *priv = pci_get_drvdata(pci_dev);
 struct net_device *dev = priv->net_dev;
 u32 val;

 if (IPW2100_PM_DISABLED)
  return 0;

 mutex_lock(&priv->action_mutex);

 IPW_DEBUG_INFO("%s: Coming out of suspend...\n", dev->name);

 pci_set_power_state(pci_dev, PCI_D0);
 pci_enable_device(pci_dev);
 pci_restore_state(pci_dev);

 /*
  * Suspend/Resume resets the PCI configuration space, so we have to
  * re-disable the RETRY_TIMEOUT register (0x41) to keep PCI Tx retries
  * from interfering with C3 CPU state. pci_restore_state won't help
  * here since it only restores the first 64 bytes pci config header.
  */
 pci_read_config_dword(pci_dev, 0x40, &val);
 if ((val & 0x0000ff00) != 0)
  pci_write_config_dword(pci_dev, 0x40, val & 0xffff00ff);

 /* Set the device back into the PRESENT state; this will also wake
  * the queue of needed */
 netif_device_attach(dev);

 /* Bring the device back up */
 if (!(priv->status & STATUS_RF_KILL_SW))
  ipw2100_up(priv, 0);

 mutex_unlock(&priv->action_mutex);

 return
 0;
}
#endif

#define IPW2100_DEV_ID(x) { PCI_VENDOR_ID_INTEL, 0x1043, 0x8086, x }

static struct pci_device_id ipw2100_pci_id_table[] __devinitdata = {
 IPW2100_DEV_ID(0x2520), /* IN 2100A mPCI 3A */
 IPW2100_DEV_ID(0x2521), /* IN 2100A mPCI 3B */
 IPW2100_DEV_ID(0x2524), /* IN 2100A mPCI 3B */
 IPW2100_DEV_ID(0x2525), /* IN 2100A mPCI 3B */
```

```
IPW2100_DEV_ID(0x2526), /* IN 2100A mPCI Gen A3 */
IPW2100_DEV_ID(0x2522), /* IN 2100 mPCI 3B */
IPW2100_DEV_ID(0x2523), /* IN 2100 mPCI 3A */
IPW2100_DEV_ID(0x2527), /* IN 2100 mPCI 3B */
IPW2100_DEV_ID(0x2528), /* IN 2100 mPCI 3B */
IPW2100_DEV_ID(0x2529), /* IN 2100 mPCI 3B */
IPW2100_DEV_ID(0x252B), /* IN 2100 mPCI 3A */
IPW2100_DEV_ID(0x252C), /* IN 2100 mPCI 3A */
IPW2100_DEV_ID(0x252D), /* IN 2100 mPCI 3A */

IPW2100_DEV_ID(0x2550), /* IB 2100A mPCI 3B */
IPW2100_DEV_ID(0x2551), /* IB 2100 mPCI 3B */
IPW2100_DEV_ID(0x2553), /* IB 2100 mPCI 3B */
IPW2100_DEV_ID(0x2554), /* IB 2100 mPCI 3B */
IPW2100_DEV_ID(0x2555), /*
IB 2100 mPCI 3B */

IPW2100_DEV_ID(0x2560), /* DE 2100A mPCI 3A */
IPW2100_DEV_ID(0x2562), /* DE 2100A mPCI 3A */
IPW2100_DEV_ID(0x2563), /* DE 2100A mPCI 3A */
IPW2100_DEV_ID(0x2561), /* DE 2100 mPCI 3A */
IPW2100_DEV_ID(0x2565), /* DE 2100 mPCI 3A */
IPW2100_DEV_ID(0x2566), /* DE 2100 mPCI 3A */
IPW2100_DEV_ID(0x2567), /* DE 2100 mPCI 3A */

IPW2100_DEV_ID(0x2570), /* GA 2100 mPCI 3B */

IPW2100_DEV_ID(0x2580), /* TO 2100A mPCI 3B */
IPW2100_DEV_ID(0x2582), /* TO 2100A mPCI 3B */
IPW2100_DEV_ID(0x2583), /* TO 2100A mPCI 3B */
IPW2100_DEV_ID(0x2581), /* TO 2100 mPCI 3B */
IPW2100_DEV_ID(0x2585), /* TO 2100 mPCI 3B */
IPW2100_DEV_ID(0x2586), /* TO 2100 mPCI 3B */
IPW2100_DEV_ID(0x2587), /* TO 2100 mPCI 3B */

IPW2100_DEV_ID(0x2590), /* SO 2100A mPCI 3B */
IPW2100_DEV_ID(0x2592), /* SO 2100A mPCI 3B */
IPW2100_DEV_ID(0x2591), /* SO 2100 mPCI 3B */
IPW2100_DEV_ID(0x2593), /* SO 2100 mPCI 3B */
IPW2100_DEV_ID(0x2596), /* SO 2100 mPCI 3B */
IPW2100_DEV_ID(0x2598), /*
SO 2100 mPCI 3B */

IPW2100_DEV_ID(0x25A0), /* HP 2100 mPCI 3B */
{0,},
};

MODULE_DEVICE_TABLE(pci, ipw2100_pci_id_table);
```

```
static struct pci_driver ipw2100_pci_driver = {
 .name = DRV_NAME,
 .id_table = ipw2100_pci_id_table,
 .probe = ipw2100_pci_init_one,
 .remove = __devexit_p(ipw2100_pci_remove_one),
#ifdef CONFIG_PM
 .suspend = ipw2100_suspend,
 .resume = ipw2100_resume,
#endif
};

/**
* Initialize the ipw2100 driver/module
*
* @returns 0 if ok, < 0 errno node con error.
*
* Note: we cannot init the /proc stuff until the PCI driver is there,
* or we risk an unlikely race condition on someone accessing
* uninitialized data in the PCI dev struct through /proc.
*/
static int __init ipw2100_init(void)
{
 int ret;

 printk(KERN_INFO DRV_NAME ": %s, %s\n", DRV_DESCRIPTION, DRV_VERSION);
 printk(KERN_INFO DRV_NAME ": %s\n", DRV_COPYRIGHT);

 ret = pci_register_driver(&ipw2100_pci_driver);
 if (ret)
  goto out;

 set_acceptable_latency("ipw2100", INFINITE_LATENCY);
#ifdef
 CONFIG_IPW2100_DEBUG
 ipw2100_debug_level = debug;
 ret = driver_create_file(&ipw2100_pci_driver.driver,
    &driver_attr_debug_level);
#endif

out:
 return ret;
}

/**
* Cleanup ipw2100 driver registration
*/
static void __exit ipw2100_exit(void)
```

```
{
 /* FIXME: IPG: check that we have no instances of the devices open */
#ifdef CONFIG_IPW2100_DEBUG
 driver_remove_file(&ipw2100_pci_driver.driver,
     &driver_attr_debug_level);
#endif
 pci_unregister_driver(&ipw2100_pci_driver);
 remove_acceptable_latency("ipw2100");
}

module_init(ipw2100_init);
module_exit(ipw2100_exit);

#define WEXT_USECHANNELS 1

static const long ipw2100_frequencies[] = {
 2412, 2417, 2422, 2427,
 2432, 2437, 2442, 2447,
 2452, 2457, 2462, 2467,
 2472, 2484
};

#define FREQ_COUNT (sizeof(ipw2100_frequencies) / \
                sizeof(ipw2100_frequencies[0]))

static const long ipw2100_rates_11b[] = {
 1000000,
 2000000,
 5500000,
 11000000
};

#define RATE_COUNT (sizeof(ipw2100_rates_11b) /
 sizeof(ipw2100_rates_11b[0]))

static int ipw2100_wx_get_name(struct net_device *dev,
         struct iw_request_info *info,
         union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 if (!(priv->status & STATUS_ASSOCIATED))
  strcpy(wrqu->name, "unassociated");
 else
  snprintf(wrqu->name, IFNAMSIZ, "IEEE 802.11b");
```

```
 IPW_DEBUG_WX("Name: %s\n", wrqu->name);
 return 0;
}

static int ipw2100_wx_set_freq(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct iw_freq *fwrq = &wrqu->freq;
 int err = 0;

 if (priv->ieee->iw_mode == IW_MODE_INFRA)
  return -EOPNOTSUPP;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 /* if setting by freq convert to channel */
 if (fwrq->e == 1) {
  if ((fwrq->m >= (int)2.412e8
 && fwrq->m <= (int)2.487e8)) {
   int f = fwrq->m / 100000;
   int c = 0;

   while ((c < REG_MAX_CHANNEL) &&
        (f != ipw2100_frequencies[c]))
    c++;

   /* hack to fall through */
   fwrq->e = 0;
   fwrq->m = c + 1;
  }
 }

 if (fwrq->e > 0 || fwrq->m > 1000) {
  err = -EOPNOTSUPP;
  goto done;
 } else {  /* Set the channel */
  IPW_DEBUG_WX("SET Freq/Channel -> %d \n", fwrq->m);
  err = ipw2100_set_channel(priv, fwrq->m, 0);
 }

    done:
```

```
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_freq(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 wrqu->freq.e = 0;

 /* If we are associated, trying to associate, or have a statically
  * configured CHANNEL then return that; otherwise return ANY */
 if (priv->config & CFG_STATIC_CHANNEL ||
    priv->status & STATUS_ASSOCIATED)
  wrqu->freq.m
 = priv->channel;
 else
  wrqu->freq.m = 0;

 IPW_DEBUG_WX("GET Freq/Channel -> %d \n", priv->channel);
 return 0;

}

static int ipw2100_wx_set_mode(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0;

 IPW_DEBUG_WX("SET Mode -> %d \n", wrqu->mode);

 if (wrqu->mode == priv->ieee->iw_mode)
  return 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }
```

```
 switch (wrqu->mode) {
#ifdef CONFIG_IPW2100_MONITOR
 case IW_MODE_MONITOR:
  err = ipw2100_switch_mode(priv, IW_MODE_MONITOR);
  break;
#endif    /* CONFIG_IPW2100_MONITOR */
 case IW_MODE_ADHOC:
  err = ipw2100_switch_mode(priv, IW_MODE_ADHOC);
  break;
 case IW_MODE_INFRA:
 case IW_MODE_AUTO:
 default:
  err = ipw2100_switch_mode(priv, IW_MODE_INFRA);
  break;
 }

    done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static
 int ipw2100_wx_get_mode(struct net_device *dev,
         struct iw_request_info *info,
         union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 wrqu->mode = priv->ieee->iw_mode;
 IPW_DEBUG_WX("GET Mode -> %d\n", wrqu->mode);

 return 0;
}

#define POWER_MODES 5

/* Values are in microsecond */
static const s32 timeout_duration[POWER_MODES] = {
 350000,
 250000,
 75000,
 37000,
 25000,
};
```

```
static const s32 period_duration[POWER_MODES] = {
 400000,
 700000,
 1000000,
 1000000,
 1000000
};

static int ipw2100_wx_get_range(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct iw_range *range = (struct iw_range *)extra;
 u16 val;
 int i, level;

 wrqu->data.length = sizeof(*range);
 memset(range,
 0, sizeof(*range));

 /* Let's try to keep this struct in the same order as in
  * linux/include/wireless.h
  */

 /* TODO: See what values we can set, and remove the ones we can't
  * set, or fill them with some default data.
  */

 /* ~5 Mb/s real (802.11b) */
 range->throughput = 5 * 1000 * 1000;

//      range->sensitivity;     /* signal level threshold range */

 range->max_qual.qual = 100;
 /* TODO: Find real max RSSI and stick here */
 range->max_qual.level = 0;
 range->max_qual.noise = 0;
 range->max_qual.updated = 7; /* Updated all three */

 range->avg_qual.qual = 70; /* > 8% missed beacons is 'bad' */
 /* TODO: Find real 'good' to 'bad' threshol value for RSSI */
 range->avg_qual.level = 20 + IPW2100_RSSI_TO_DBM;
```

```
range->avg_qual.noise = 0;
range->avg_qual.updated = 7; /* Updated all three */

range->num_bitrates = RATE_COUNT;

for (i = 0; i < RATE_COUNT && i < IW_MAX_BITRATES; i++) {
 range->bitrate[i] = ipw2100_rates_11b[i];
}

range->min_rts = MIN_RTS_THRESHOLD;
range->max_rts
= MAX_RTS_THRESHOLD;
range->min_frag = MIN_FRAG_THRESHOLD;
range->max_frag = MAX_FRAG_THRESHOLD;

range->min_pmp = period_duration[0]; /* Minimal PM period */
range->max_pmp = period_duration[POWER_MODES - 1]; /* Maximal PM period */
range->min_pmt = timeout_duration[POWER_MODES - 1]; /* Minimal PM timeout */
range->max_pmt = timeout_duration[0]; /* Maximal PM timeout */

/* How to decode max/min PM period */
range->pmp_flags = IW_POWER_PERIOD;
/* How to decode max/min PM period */
range->pmt_flags = IW_POWER_TIMEOUT;
/* What PM options are supported */
range->pm_capa = IW_POWER_TIMEOUT | IW_POWER_PERIOD;

range->encoding_size[0] = 5;
range->encoding_size[1] = 13; /* Different token sizes */
range->num_encoding_sizes = 2; /* Number of entry in the list */
range->max_encoding_tokens = WEP_KEYS; /* Max number of tokens */
//      range->encoding_login_index;            /* token index for login token */

if (priv->ieee->iw_mode == IW_MODE_ADHOC) {
 range->txpower_capa
= IW_TXPOW_DBM;
 range->num_txpower = IW_MAX_TXPOWER;
 for (i = 0, level = (IPW_TX_POWER_MAX_DBM * 16);
      i < IW_MAX_TXPOWER;
      i++, level -=
      ((IPW_TX_POWER_MAX_DBM -
        IPW_TX_POWER_MIN_DBM) * 16) / (IW_MAX_TXPOWER - 1))
  range->txpower[i] = level / 16;
} else {
 range->txpower_capa = 0;
 range->num_txpower = 0;
}
```

```
 /* Set the Wireless Extension versions */
 range->we_version_compiled = WIRELESS_EXT;
 range->we_version_source = 18;

//      range->retry_capa;      /* What retry options are supported */
//      range->retry_flags;     /* How to decode max/min retry limit */
//      range->r_time_flags;    /* How to decode max/min retry life */
//      range->min_retry;       /* Minimal number of retries */
//      range->max_retry;       /* Maximal number of retries */
//      range->min_r_time;      /* Minimal retry lifetime */
//      range->max_r_time;      /* Maximal retry lifetime */

 range->num_channels = FREQ_COUNT;

 val = 0;
 for
 (i = 0; i < FREQ_COUNT; i++) {
  // TODO: Include only legal frequencies for some countries
//              if (local->channel_mask & (1 << i)) {
  range->freq[val].i = i + 1;
  range->freq[val].m = ipw2100_frequencies[i] * 100000;
  range->freq[val].e = 1;
  val++;
//              }
  if (val == IW_MAX_FREQUENCIES)
   break;
 }
 range->num_frequency = val;

 /* Event capability (kernel + driver) */
 range->event_capa[0] = (IW_EVENT_CAPA_K_0 |
    IW_EVENT_CAPA_MASK(SIOCGIWAP));
 range->event_capa[1] = IW_EVENT_CAPA_K_1;

 range->enc_capa = IW_ENC_CAPA_WPA | IW_ENC_CAPA_WPA2 |
  IW_ENC_CAPA_CIPHER_TKIP | IW_ENC_CAPA_CIPHER_CCMP;

 IPW_DEBUG_WX("GET Range\n");

 return 0;
}

static int ipw2100_wx_set_wap(struct net_device *dev,
         struct iw_request_info *info,
         union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0;
```

```
static const unsigned char any[] = {
 0xff, 0xff, 0xff, 0xff, 0xff, 0xff
};
static const unsigned char off[]
= {
 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

// sanity checks
if (wrqu->ap_addr.sa_family != ARPHRD_ETHER)
 return -EINVAL;

mutex_lock(&priv->action_mutex);
if (!(priv->status & STATUS_INITIALIZED)) {
 err = -EIO;
 goto done;
}

if (!memcmp(any, wrqu->ap_addr.sa_data, ETH_ALEN) ||
    !memcmp(off, wrqu->ap_addr.sa_data, ETH_ALEN)) {
 /* we disable mandatory BSSID association */
 IPW_DEBUG_WX("exit - disable mandatory BSSID\n");
 priv->config &= ~CFG_STATIC_BSSID;
 err = ipw2100_set_mandatory_bssid(priv, NULL, 0);
 goto done;
}

priv->config |= CFG_STATIC_BSSID;
memcpy(priv->mandatory_bssid_mac, wrqu->ap_addr.sa_data, ETH_ALEN);

err = ipw2100_set_mandatory_bssid(priv, wrqu->ap_addr.sa_data, 0);

IPW_DEBUG_WX("SET BSSID -> %02X:%02X:%02X:%02X:%02X:%02X\n",
      wrqu->ap_addr.sa_data[0] & 0xff,
      wrqu->ap_addr.sa_data[1] & 0xff,
      wrqu->ap_addr.sa_data[2] & 0xff,
      wrqu->ap_addr.sa_data[3] & 0xff,
      wrqu->ap_addr.sa_data[4] & 0xff,
      wrqu->ap_addr.sa_data[5]
& 0xff);

    done:
mutex_unlock(&priv->action_mutex);
return err;
}

static int ipw2100_wx_get_wap(struct net_device *dev,
```

```
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 /* If we are associated, trying to associate, or have a statically
  * configured BSSID then return that; otherwise return ANY */
 if (priv->config & CFG_STATIC_BSSID || priv->status & STATUS_ASSOCIATED) {
  wrqu->ap_addr.sa_family = ARPHRD_ETHER;
  memcpy(wrqu->ap_addr.sa_data, priv->bssid, ETH_ALEN);
 } else
  memset(wrqu->ap_addr.sa_data, 0, ETH_ALEN);

 IPW_DEBUG_WX("Getting WAP BSSID: " MAC_FMT "\n",
      MAC_ARG(wrqu->ap_addr.sa_data));
 return 0;
}

static int ipw2100_wx_set_essid(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 char
 *essid = ""; /* ANY */
 int length = 0;
 int err = 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 if (wrqu->essid.flags && wrqu->essid.length) {
  length = wrqu->essid.length;
  essid = extra;
 }

 if (length == 0) {
  IPW_DEBUG_WX("Setting ESSID to ANY\n");
  priv->config &= ~CFG_STATIC_ESSID;
  err = ipw2100_set_essid(priv, NULL, 0, 0);
  goto done;
```

```
 }

 length = min(length, IW_ESSID_MAX_SIZE);

 priv->config |= CFG_STATIC_ESSID;

 if (priv->essid_len == length && !memcmp(priv->essid, extra, length)) {
  IPW_DEBUG_WX("ESSID set to current ESSID.\n");
  err = 0;
  goto done;
 }

 IPW_DEBUG_WX("Setting ESSID: '%s' (%d)\n", escape_essid(essid, length),
       length);

 priv->essid_len = length;
 memcpy(priv->essid, essid, priv->essid_len);

 err = ipw2100_set_essid(priv, essid, length, 0);

     done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_essid(struct net_device *dev,
    struct
 iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 /* If we are associated, trying to associate, or have a statically
  * configured ESSID then return that; otherwise return ANY */
 if (priv->config & CFG_STATIC_ESSID || priv->status & STATUS_ASSOCIATED) {
  IPW_DEBUG_WX("Getting essid: '%s'\n",
       escape_essid(priv->essid, priv->essid_len));
  memcpy(extra, priv->essid, priv->essid_len);
  wrqu->essid.length = priv->essid_len;
  wrqu->essid.flags = 1; /* active */
 } else {
  IPW_DEBUG_WX("Getting essid: ANY\n");
  wrqu->essid.length = 0;
  wrqu->essid.flags = 0; /* active */
```

```
 }

 return 0;
}

static int ipw2100_wx_set_nick(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv
 = ieee80211_priv(dev);

 if (wrqu->data.length > IW_ESSID_MAX_SIZE)
  return -E2BIG;

 wrqu->data.length = min((size_t) wrqu->data.length, sizeof(priv->nick));
 memset(priv->nick, 0, sizeof(priv->nick));
 memcpy(priv->nick, extra, wrqu->data.length);

 IPW_DEBUG_WX("SET Nickname -> %s \n", priv->nick);

 return 0;
}

static int ipw2100_wx_get_nick(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 wrqu->data.length = strlen(priv->nick);
 memcpy(extra, priv->nick, wrqu->data.length);
 wrqu->data.flags = 1; /* active */

 IPW_DEBUG_WX("GET Nickname -> %s \n", extra);

 return 0;
}

static int ipw2100_wx_set_rate(struct net_device *dev,
```

```
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 u32 target_rate = wrqu->bitrate.value;
 u32
 rate;
 int err = 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 rate = 0;

 if (target_rate == 1000000 ||
    (!wrqu->bitrate.fixed && target_rate > 1000000))
  rate |= TX_RATE_1_MBIT;
 if (target_rate == 2000000 ||
    (!wrqu->bitrate.fixed && target_rate > 2000000))
  rate |= TX_RATE_2_MBIT;
 if (target_rate == 5500000 ||
    (!wrqu->bitrate.fixed && target_rate > 5500000))
  rate |= TX_RATE_5_5_MBIT;
 if (target_rate == 11000000 ||
    (!wrqu->bitrate.fixed && target_rate > 11000000))
  rate |= TX_RATE_11_MBIT;
 if (rate == 0)
  rate = DEFAULT_TX_RATES;

 err = ipw2100_set_tx_rates(priv, rate, 0);

 IPW_DEBUG_WX("SET Rate -> %04X \n", rate);
    done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_rate(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int val;
 int
```

```
len = sizeof(val);
int err = 0;

if (!(priv->status & STATUS_ENABLED) ||
    priv->status & STATUS_RF_KILL_MASK ||
    !(priv->status & STATUS_ASSOCIATED)) {
 wrqu->bitrate.value = 0;
 return 0;
}

mutex_lock(&priv->action_mutex);
if (!(priv->status & STATUS_INITIALIZED)) {
 err = -EIO;
 goto done;
}

err = ipw2100_get_ordinal(priv, IPW_ORD_CURRENT_TX_RATE, &val, &len);
if (err) {
 IPW_DEBUG_WX("failed querying ordinals.\n");
 return err;
}

switch (val & TX_RATE_MASK) {
case TX_RATE_1_MBIT:
 wrqu->bitrate.value = 1000000;
 break;
case TX_RATE_2_MBIT:
 wrqu->bitrate.value = 2000000;
 break;
case TX_RATE_5_5_MBIT:
 wrqu->bitrate.value = 5500000;
 break;
case TX_RATE_11_MBIT:
 wrqu->bitrate.value = 11000000;
 break;
default:
 wrqu->bitrate.value = 0;
}

IPW_DEBUG_WX("GET Rate -> %d \n", wrqu->bitrate.value);

   done:
mutex_unlock(&priv->action_mutex);
return err;
}

static int ipw2100_wx_set_rts(struct net_device *dev,
        struct iw_request_info
```

```
 *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int value, err;

 /* Auto RTS not yet supported */
 if (wrqu->rts.fixed == 0)
  return -EINVAL;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 if (wrqu->rts.disabled)
  value = priv->rts_threshold | RTS_DISABLED;
 else {
  if (wrqu->rts.value < 1 || wrqu->rts.value > 2304) {
   err = -EINVAL;
   goto done;
  }
  value = wrqu->rts.value;
 }

 err = ipw2100_set_rts_threshold(priv, value);

 IPW_DEBUG_WX("SET RTS Threshold -> 0x%08X \n", value);
    done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_rts(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 wrqu->rts.value = priv->rts_threshold & ~RTS_DISABLED;
 wrqu->rts.fixed
 = 1; /* no auto select */
```

```
 /* If RTS is set to the default value, then it is disabled */
 wrqu->rts.disabled = (priv->rts_threshold & RTS_DISABLED) ? 1 : 0;

 IPW_DEBUG_WX("GET RTS Threshold -> 0x%08X \n", wrqu->rts.value);

 return 0;
}

static int ipw2100_wx_set_txpow(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0, value;

 if (ipw_radio_kill_sw(priv, wrqu->txpower.disabled))
  return -EINPROGRESS;

 if (priv->ieee->iw_mode != IW_MODE_ADHOC)
  return 0;

 if ((wrqu->txpower.flags & IW_TXPOW_TYPE) != IW_TXPOW_DBM)
  return -EINVAL;

 if (wrqu->txpower.fixed == 0)
  value = IPW_TX_POWER_DEFAULT;
 else {
  if (wrqu->txpower.value < IPW_TX_POWER_MIN_DBM ||
      wrqu->txpower.value > IPW_TX_POWER_MAX_DBM)
   return -EINVAL;

  value = wrqu->txpower.value;
 }

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err
 = -EIO;
  goto done;
 }

 err = ipw2100_set_tx_power(priv, value);

 IPW_DEBUG_WX("SET TX Power -> %d \n", value);

     done:
 mutex_unlock(&priv->action_mutex);
 return err;
```

```
}

static int ipw2100_wx_get_txpow(struct net_device *dev,
   struct iw_request_info *info,
   union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 wrqu->txpower.disabled = (priv->status & STATUS_RF_KILL_MASK) ? 1 : 0;

 if (priv->tx_power == IPW_TX_POWER_DEFAULT) {
  wrqu->txpower.fixed = 0;
  wrqu->txpower.value = IPW_TX_POWER_MAX_DBM;
 } else {
  wrqu->txpower.fixed = 1;
  wrqu->txpower.value = priv->tx_power;
 }

 wrqu->txpower.flags = IW_TXPOW_DBM;

 IPW_DEBUG_WX("GET TX Power -> %d \n", wrqu->txpower.value);

 return 0;
}

static int ipw2100_wx_set_frag(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any
 time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 if (!wrqu->frag.fixed)
  return -EINVAL;

 if (wrqu->frag.disabled) {
  priv->frag_threshold |= FRAG_DISABLED;
  priv->ieee->fts = DEFAULT_FTS;
 } else {
  if (wrqu->frag.value < MIN_FRAG_THRESHOLD ||
```

```
     wrqu->frag.value > MAX_FRAG_THRESHOLD)
  return -EINVAL;

 priv->ieee->fts = wrqu->frag.value & ~0x1;
 priv->frag_threshold = priv->ieee->fts;
 }

 IPW_DEBUG_WX("SET Frag Threshold -> %d \n", priv->ieee->fts);

 return 0;
}

static int ipw2100_wx_get_frag(struct net_device *dev,
         struct iw_request_info *info,
         union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 wrqu->frag.value = priv->frag_threshold & ~FRAG_DISABLED;
 wrqu->frag.fixed = 0; /* no auto select */
 wrqu->frag.disabled = (priv->frag_threshold & FRAG_DISABLED) ? 1 : 0;

 IPW_DEBUG_WX("GET Frag Threshold
 -> %d \n", wrqu->frag.value);

 return 0;
}

static int ipw2100_wx_set_retry(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0;

 if (wrqu->retry.flags & IW_RETRY_LIFETIME || wrqu->retry.disabled)
  return -EINVAL;

 if (!(wrqu->retry.flags & IW_RETRY_LIMIT))
  return 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
```

```
 }

 if (wrqu->retry.flags & IW_RETRY_SHORT) {
  err = ipw2100_set_short_retry(priv, wrqu->retry.value);
  IPW_DEBUG_WX("SET Short Retry Limit -> %d \n",
       wrqu->retry.value);
  goto done;
 }

 if (wrqu->retry.flags & IW_RETRY_LONG) {
  err = ipw2100_set_long_retry(priv, wrqu->retry.value);
  IPW_DEBUG_WX("SET Long Retry Limit -> %d \n",
       wrqu->retry.value);
  goto done;
 }

 err = ipw2100_set_short_retry(priv, wrqu->retry.value);
 if (!err)
  err = ipw2100_set_long_retry(priv, wrqu->retry.value);

 IPW_DEBUG_WX("SET
 Both Retry Limits -> %d \n", wrqu->retry.value);

     done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_retry(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 wrqu->retry.disabled = 0; /* can't be disabled */

 if ((wrqu->retry.flags & IW_RETRY_TYPE) == IW_RETRY_LIFETIME)
  return -EINVAL;

 if (wrqu->retry.flags & IW_RETRY_LONG) {
  wrqu->retry.flags = IW_RETRY_LIMIT | IW_RETRY_LONG;
  wrqu->retry.value = priv->long_retry_limit;
 } else {
  wrqu->retry.flags =
```

```
      (priv->short_retry_limit !=
       priv->long_retry_limit) ?
      IW_RETRY_LIMIT | IW_RETRY_SHORT : IW_RETRY_LIMIT;

  wrqu->retry.value = priv->short_retry_limit;
 }

 IPW_DEBUG_WX("GET Retry -> %d \n", wrqu->retry.value);

 return 0;
}

static int ipw2100_wx_set_scan(struct net_device
 *dev,
         struct iw_request_info *info,
         union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 IPW_DEBUG_WX("Initiating scan...\n");
 if (ipw2100_set_scan_options(priv) || ipw2100_start_scan(priv)) {
  IPW_DEBUG_WX("Start scan failed.\n");

  /* TODO: Mark a scan as pending so when hardware initialized
   *       a scan starts */
 }

    done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_scan(struct net_device *dev,
         struct iw_request_info *info,
         union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
```

```
 return ieee80211_wx_get_scan(priv->ieee, info, wrqu, extra);
}

/*
* Implementation based on code in hostap-driver v0.1.3 hostap_ioctl.c
*/
static int ipw2100_wx_set_encode(struct
 net_device *dev,
     struct iw_request_info *info,
     union iwreq_data *wrqu, char *key)
{
 /*
  * No check of STATUS_INITIALIZED required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 return ieee80211_wx_set_encode(priv->ieee, info, wrqu, key);
}

static int ipw2100_wx_get_encode(struct net_device *dev,
     struct iw_request_info *info,
     union iwreq_data *wrqu, char *key)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 return ieee80211_wx_get_encode(priv->ieee, info, wrqu, key);
}

static int ipw2100_wx_set_power(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 if (wrqu->power.disabled) {
  priv->power_mode = IPW_POWER_LEVEL(priv->power_mode);
  err
```

```
= ipw2100_set_power_mode(priv, IPW_POWER_MODE_CAM);
 IPW_DEBUG_WX("SET Power Management Mode -> off\n");
 goto done;
}

switch (wrqu->power.flags & IW_POWER_MODE) {
case IW_POWER_ON: /* If not specified */
case IW_POWER_MODE: /* If set all mask */
case IW_POWER_ALL_R: /* If explicitely state all */
 break;
default:  /* Otherwise we don't support it */
 IPW_DEBUG_WX("SET PM Mode: %X not supported.\n",
       wrqu->power.flags);
 err = -EOPNOTSUPP;
 goto done;
}

/* If the user hasn't specified a power management mode yet, default
 * to BATTERY */
priv->power_mode = IPW_POWER_ENABLED | priv->power_mode;
err = ipw2100_set_power_mode(priv, IPW_POWER_LEVEL(priv->power_mode));

IPW_DEBUG_WX("SET Power Management Mode -> 0x%02X\n", priv->power_mode);

    done:
mutex_unlock(&priv->action_mutex);
return err;

}

static int ipw2100_wx_get_power(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
/*
 * This can be
called at any time.  No action lock required
 */

struct ipw2100_priv *priv = ieee80211_priv(dev);

if (!(priv->power_mode & IPW_POWER_ENABLED))
 wrqu->power.disabled = 1;
else {
 wrqu->power.disabled = 0;
 wrqu->power.flags = 0;
}
```

```
 IPW_DEBUG_WX("GET Power Management Mode -> %02X\n", priv->power_mode);

 return 0;
}

/*
* WE-18 WPA support
*/

/* SIOCSIWGENIE */
static int ipw2100_wx_set_genie(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct ieee80211_device *ieee = priv->ieee;
 u8 *buf;

 if (!ieee->wpa_enabled)
  return -EOPNOTSUPP;

 if (wrqu->data.length > MAX_WPA_IE_LEN ||
     (wrqu->data.length && extra == NULL))
  return -EINVAL;

 if (wrqu->data.length) {
  buf = kmalloc(wrqu->data.length, GFP_KERNEL);
  if (buf == NULL)
   return -ENOMEM;

  memcpy(buf, extra, wrqu->data.length);
  kfree(ieee->wpa_ie);
  ieee->wpa_ie = buf;
  ieee->wpa_ie_len = wrqu->data.length;
 }
 else {
  kfree(ieee->wpa_ie);
  ieee->wpa_ie = NULL;
  ieee->wpa_ie_len = 0;
 }

 ipw2100_wpa_assoc_frame(priv, ieee->wpa_ie, ieee->wpa_ie_len);

 return 0;
}

/* SIOCGIWGENIE */
```

```
static int ipw2100_wx_get_genie(struct net_device *dev,
    struct iw_request_info *info,
    union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct ieee80211_device *ieee = priv->ieee;

 if (ieee->wpa_ie_len == 0 || ieee->wpa_ie == NULL) {
  wrqu->data.length = 0;
  return 0;
 }

 if (wrqu->data.length < ieee->wpa_ie_len)
  return -E2BIG;

 wrqu->data.length = ieee->wpa_ie_len;
 memcpy(extra, ieee->wpa_ie, ieee->wpa_ie_len);

 return 0;
}

/* SIOCSIWAUTH */
static int ipw2100_wx_set_auth(struct net_device *dev,
          struct iw_request_info *info,
          union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct ieee80211_device *ieee = priv->ieee;
 struct iw_param *param = &wrqu->param;
 struct ieee80211_crypt_data
 *crypt;
 unsigned long flags;
 int ret = 0;

 switch (param->flags & IW_AUTH_INDEX) {
 case IW_AUTH_WPA_VERSION:
 case IW_AUTH_CIPHER_PAIRWISE:
 case IW_AUTH_CIPHER_GROUP:
 case IW_AUTH_KEY_MGMT:
  /*
   * ipw2200 does not use these parameters
   */
  break;

 case IW_AUTH_TKIP_COUNTERMEASURES:
  crypt = priv->ieee->crypt[priv->ieee->tx_keyidx];
  if (!crypt || !crypt->ops->set_flags || !crypt->ops->get_flags)
   break;
```

```
 flags = crypt->ops->get_flags(crypt->priv);

 if (param->value)
  flags |= IEEE80211_CRYPTO_TKIP_COUNTERMEASURES;
 else
  flags &= ~IEEE80211_CRYPTO_TKIP_COUNTERMEASURES;

 crypt->ops->set_flags(flags, crypt->priv);

 break;

case IW_AUTH_DROP_UNENCRYPTED:{
  /* HACK:
   *
   * wpa_supplicant calls set_wpa_enabled when the driver
   * is loaded and unloaded, regardless of if WPA is being
   * used.  No other calls are made which can be used to
   * determine if encryption will be used or not prior to
   * association being expected.  If encryption is
not being
   * used, drop_unencrypted is set to false, else true -- we
   * can use this to determine if the CAP_PRIVACY_ON bit should
   * be set.
   */
  struct ieee80211_security sec = {
   .flags = SEC_ENABLED,
   .enabled = param->value,
  };
  priv->ieee->drop_unencrypted = param->value;
  /* We only change SEC_LEVEL for open mode. Others
   * are set by ipw_wpa_set_encryption.
   */
  if (!param->value) {
   sec.flags |= SEC_LEVEL;
   sec.level = SEC_LEVEL_0;
  } else {
   sec.flags |= SEC_LEVEL;
   sec.level = SEC_LEVEL_1;
  }
  if (priv->ieee->set_security)
   priv->ieee->set_security(priv->ieee->dev, &sec);
  break;
 }

case IW_AUTH_80211_AUTH_ALG:
 ret = ipw2100_wpa_set_auth_algs(priv, param->value);
 break;
```

```
 case IW_AUTH_WPA_ENABLED:
  ret = ipw2100_wpa_enable(priv, param->value);
  break;

 case IW_AUTH_RX_UNENCRYPTED_EAPOL:
  ieee->ieee802_1x = param->value;
  break;

  //case IW_AUTH_ROAMING_CONTROL:
 case IW_AUTH_PRIVACY_INVOKED:
  ieee->privacy_invoked
 = param->value;
  break;

 default:
  return -EOPNOTSUPP;
 }
 return ret;
}

/* SIOCGIWAUTH */
static int ipw2100_wx_get_auth(struct net_device *dev,
          struct iw_request_info *info,
          union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct ieee80211_device *ieee = priv->ieee;
 struct ieee80211_crypt_data *crypt;
 struct iw_param *param = &wrqu->param;
 int ret = 0;

 switch (param->flags & IW_AUTH_INDEX) {
 case IW_AUTH_WPA_VERSION:
 case IW_AUTH_CIPHER_PAIRWISE:
 case IW_AUTH_CIPHER_GROUP:
 case IW_AUTH_KEY_MGMT:
  /*
   * wpa_supplicant will control these internally
   */
  ret = -EOPNOTSUPP;
  break;

 case IW_AUTH_TKIP_COUNTERMEASURES:
  crypt = priv->ieee->crypt[priv->ieee->tx_keyidx];
  if (!crypt || !crypt->ops->get_flags) {
   IPW_DEBUG_WARNING("Can't get TKIP countermeasures: "
       "crypt not set!\n");
```

```
  break;
 }

 param->value = (crypt->ops->get_flags(crypt->priv) &
   IEEE80211_CRYPTO_TKIP_COUNTERMEASURES)
? 1 : 0;

 break;

case IW_AUTH_DROP_UNENCRYPTED:
 param->value = ieee->drop_unencrypted;
 break;

case IW_AUTH_80211_AUTH_ALG:
 param->value = priv->ieee->sec.auth_mode;
 break;

case IW_AUTH_WPA_ENABLED:
 param->value = ieee->wpa_enabled;
 break;

case IW_AUTH_RX_UNENCRYPTED_EAPOL:
 param->value = ieee->ieee802_1x;
 break;

case IW_AUTH_ROAMING_CONTROL:
case IW_AUTH_PRIVACY_INVOKED:
 param->value = ieee->privacy_invoked;
 break;

default:
 return -EOPNOTSUPP;
 }
 return 0;
}

/* SIOCSIWENCODEEXT */
static int ipw2100_wx_set_encodeext(struct net_device *dev,
       struct iw_request_info *info,
       union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 return ieee80211_wx_set_encodeext(priv->ieee, info, wrqu, extra);
}

/* SIOCGIWENCODEEXT */
static int ipw2100_wx_get_encodeext(struct net_device *dev,
       struct iw_request_info *info,
```

```
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 return
 ieee80211_wx_get_encodeext(priv->ieee, info, wrqu, extra);
}

/* SIOCSIWMLME */
static int ipw2100_wx_set_mlme(struct net_device *dev,
          struct iw_request_info *info,
          union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 struct iw_mlme *mlme = (struct iw_mlme *)extra;
 u16 reason;

 reason = cpu_to_le16(mlme->reason_code);

 switch (mlme->cmd) {
 case IW_MLME_DEAUTH:
  // silently ignore
  break;

 case IW_MLME_DISASSOC:
  ipw2100_disassociate_bssid(priv);
  break;

 default:
  return -EOPNOTSUPP;
 }
 return 0;
}

/*
*
* IWPRIV handlers
*
*/
#ifdef CONFIG_IPW2100_MONITOR
static int ipw2100_wx_set_promisc(struct net_device *dev,
      struct iw_request_info *info,
      union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int *parms = (int *)extra;
 int enable = (parms[0] > 0);
 int err = 0;
```

```
 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err
 = -EIO;
  goto done;
 }

 if (enable) {
  if (priv->ieee->iw_mode == IW_MODE_MONITOR) {
   err = ipw2100_set_channel(priv, parms[1], 0);
   goto done;
  }
  priv->channel = parms[1];
  err = ipw2100_switch_mode(priv, IW_MODE_MONITOR);
 } else {
  if (priv->ieee->iw_mode == IW_MODE_MONITOR)
   err = ipw2100_switch_mode(priv, priv->last_mode);
 }
     done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_reset(struct net_device *dev,
       struct iw_request_info *info,
       union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 if (priv->status & STATUS_INITIALIZED)
  schedule_reset(priv);
 return 0;
}

#endif

static int ipw2100_wx_set_powermode(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err = 0, mode = *(int *)extra;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err
 = -EIO;
  goto done;
 }
```

```
 if ((mode < 1) || (mode > POWER_MODES))
  mode = IPW_POWER_AUTO;

 if (priv->power_mode != mode)
  err = ipw2100_set_power_mode(priv, mode);
     done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

#define MAX_POWER_STRING 80
static int ipw2100_wx_get_powermode(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int level = IPW_POWER_LEVEL(priv->power_mode);
 s32 timeout, period;

 if (!(priv->power_mode & IPW_POWER_ENABLED)) {
  snprintf(extra, MAX_POWER_STRING,
    "Power save level: %d (Off)", level);
 } else {
  switch (level) {
  case IPW_POWER_MODE_CAM:
   snprintf(extra, MAX_POWER_STRING,
     "Power save level: %d (None)", level);
   break;
  case IPW_POWER_AUTO:
   snprintf(extra, MAX_POWER_STRING,
     "Power save level: %d (Auto)", 0);
   break;
  default:
   timeout
 = timeout_duration[level - 1] / 1000;
   period = period_duration[level - 1] / 1000;
   snprintf(extra, MAX_POWER_STRING,
     "Power save level: %d "
     "(Timeout %dms, Period %dms)",
     level, timeout, period);
  }
 }
```

```
 wrqu->data.length = strlen(extra) + 1;

 return 0;
}

static int ipw2100_wx_set_preamble(struct net_device *dev,
       struct iw_request_info *info,
       union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err, mode = *(int *)extra;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 if (mode == 1)
  priv->config |= CFG_LONG_PREAMBLE;
 else if (mode == 0)
  priv->config &= ~CFG_LONG_PREAMBLE;
 else {
  err = -EINVAL;
  goto done;
 }

 err = ipw2100_system_config(priv, 0);

    done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_preamble(struct net_device *dev,
       struct iw_request_info *info,
       union iwreq_data
 *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 if (priv->config & CFG_LONG_PREAMBLE)
  snprintf(wrqu->name, IFNAMSIZ, "long (1)");
 else
```

```
 snprintf(wrqu->name, IFNAMSIZ, "auto (0)");

 return 0;
}

#ifdef CONFIG_IPW2100_MONITOR
static int ipw2100_wx_set_crc_check(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data *wrqu, char *extra)
{
 struct ipw2100_priv *priv = ieee80211_priv(dev);
 int err, mode = *(int *)extra;

 mutex_lock(&priv->action_mutex);
 if (!(priv->status & STATUS_INITIALIZED)) {
  err = -EIO;
  goto done;
 }

 if (mode == 1)
  priv->config |= CFG_CRC_CHECK;
 else if (mode == 0)
  priv->config &= ~CFG_CRC_CHECK;
 else {
  err = -EINVAL;
  goto done;
 }
 err = 0;

    done:
 mutex_unlock(&priv->action_mutex);
 return err;
}

static int ipw2100_wx_get_crc_check(struct net_device *dev,
        struct iw_request_info *info,
        union iwreq_data
 *wrqu, char *extra)
{
 /*
  * This can be called at any time.  No action lock required
  */

 struct ipw2100_priv *priv = ieee80211_priv(dev);

 if (priv->config & CFG_CRC_CHECK)
  snprintf(wrqu->name, IFNAMSIZ, "CRC checked (1)");
 else
```

```
  snprintf(wrqu->name, IFNAMSIZ, "CRC ignored (0)");

 return 0;
}
#endif    /* CONFIG_IPW2100_MONITOR */

static iw_handler ipw2100_wx_handlers[] = {
 NULL,   /* SIOCSIWCOMMIT */
 ipw2100_wx_get_name, /* SIOCGIWNAME */
 NULL,   /* SIOCSIWNWID */
 NULL,   /* SIOCGIWNWID */
 ipw2100_wx_set_freq, /* SIOCSIWFREQ */
 ipw2100_wx_get_freq, /* SIOCGIWFREQ */
 ipw2100_wx_set_mode, /* SIOCSIWMODE */
 ipw2100_wx_get_mode, /* SIOCGIWMODE */
 NULL,   /* SIOCSIWSENS */
 NULL,   /* SIOCGIWSENS */
 NULL,   /* SIOCSIWRANGE */
 ipw2100_wx_get_range, /* SIOCGIWRANGE */
 NULL,   /* SIOCSIWPRIV */
 NULL,   /* SIOCGIWPRIV */
 NULL,   /* SIOCSIWSTATS */
 NULL,   /* SIOCGIWSTATS */
 NULL,   /* SIOCSIWSPY */
 NULL,   /* SIOCGIWSPY */
 NULL,   /* SIOCGIWTHRSPY */
 NULL,   /*
 SIOCWIWTHRSPY */
 ipw2100_wx_set_wap, /* SIOCSIWAP */
 ipw2100_wx_get_wap, /* SIOCGIWAP */
 ipw2100_wx_set_mlme, /* SIOCSIWMLME */
 NULL,   /* SIOCGIWAPLIST -- deprecated */
 ipw2100_wx_set_scan, /* SIOCSIWSCAN */
 ipw2100_wx_get_scan, /* SIOCGIWSCAN */
 ipw2100_wx_set_essid, /* SIOCSIWESSID */
 ipw2100_wx_get_essid, /* SIOCGIWESSID */
 ipw2100_wx_set_nick, /* SIOCSIWNICKN */
 ipw2100_wx_get_nick, /* SIOCGIWNICKN */
 NULL,   /* -- hole -- */
 NULL,   /* -- hole -- */
 ipw2100_wx_set_rate, /* SIOCSIWRATE */
 ipw2100_wx_get_rate, /* SIOCGIWRATE */
 ipw2100_wx_set_rts, /* SIOCSIWRTS */
 ipw2100_wx_get_rts, /* SIOCGIWRTS */
 ipw2100_wx_set_frag, /* SIOCSIWFRAG */
 ipw2100_wx_get_frag, /* SIOCGIWFRAG */
 ipw2100_wx_set_txpow, /* SIOCSIWTXPOW */
 ipw2100_wx_get_txpow, /* SIOCGIWTXPOW */
```

```
 ipw2100_wx_set_retry, /* SIOCSIWRETRY */
 ipw2100_wx_get_retry, /* SIOCGIWRETRY */
 ipw2100_wx_set_encode, /* SIOCSIWENCODE */
 ipw2100_wx_get_encode, /* SIOCGIWENCODE */
 ipw2100_wx_set_power, /* SIOCSIWPOWER
 */
 ipw2100_wx_get_power, /* SIOCGIWPOWER */
 NULL,   /* -- hole -- */
 NULL,   /* -- hole -- */
 ipw2100_wx_set_genie, /* SIOCSIWGENIE */
 ipw2100_wx_get_genie, /* SIOCGIWGENIE */
 ipw2100_wx_set_auth, /* SIOCSIWAUTH */
 ipw2100_wx_get_auth, /* SIOCGIWAUTH */
 ipw2100_wx_set_encodeext, /* SIOCSIWENCODEEXT */
 ipw2100_wx_get_encodeext, /* SIOCGIWENCODEEXT */
 NULL,   /* SIOCSIWPMKSA */
};

#define IPW2100_PRIV_SET_MONITOR SIOCIWFIRSTPRIV
#define IPW2100_PRIV_RESET  SIOCIWFIRSTPRIV+1
#define IPW2100_PRIV_SET_POWER  SIOCIWFIRSTPRIV+2
#define IPW2100_PRIV_GET_POWER  SIOCIWFIRSTPRIV+3
#define IPW2100_PRIV_SET_LONGPREAMBLE SIOCIWFIRSTPRIV+4
#define IPW2100_PRIV_GET_LONGPREAMBLE SIOCIWFIRSTPRIV+5
#define IPW2100_PRIV_SET_CRC_CHECK SIOCIWFIRSTPRIV+6
#define IPW2100_PRIV_GET_CRC_CHECK SIOCIWFIRSTPRIV+7

static const struct iw_priv_args ipw2100_private_args[] = {

#ifdef CONFIG_IPW2100_MONITOR
 {
  IPW2100_PRIV_SET_MONITOR,
  IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 2, 0, "monitor"},
 {

 IPW2100_PRIV_RESET,
  IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 0, 0, "reset"},
#endif    /* CONFIG_IPW2100_MONITOR */

 {
  IPW2100_PRIV_SET_POWER,
  IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1, 0, "set_power"},
 {
  IPW2100_PRIV_GET_POWER,
  0, IW_PRIV_TYPE_CHAR | IW_PRIV_SIZE_FIXED | MAX_POWER_STRING,
  "get_power"},
 {
  IPW2100_PRIV_SET_LONGPREAMBLE,
```

```
  IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1, 0, "set_preamble"},
 {
  IPW2100_PRIV_GET_LONGPREAMBLE,
  0, IW_PRIV_TYPE_CHAR | IW_PRIV_SIZE_FIXED | IFNAMSIZ, "get_preamble"},
#ifdef CONFIG_IPW2100_MONITOR
 {
  IPW2100_PRIV_SET_CRC_CHECK,
  IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1, 0, "set_crc_check"},
 {
  IPW2100_PRIV_GET_CRC_CHECK,
  0, IW_PRIV_TYPE_CHAR | IW_PRIV_SIZE_FIXED | IFNAMSIZ, "get_crc_check"},
#endif    /* CONFIG_IPW2100_MONITOR */
};

static iw_handler ipw2100_private_handler[] = {
#ifdef CONFIG_IPW2100_MONITOR
 ipw2100_wx_set_promisc,
 ipw2100_wx_reset,
#else    /* CONFIG_IPW2100_MONITOR */
 NULL,
 NULL,
#endif    /* CONFIG_IPW2100_MONITOR
 */
 ipw2100_wx_set_powermode,
 ipw2100_wx_get_powermode,
 ipw2100_wx_set_preamble,
 ipw2100_wx_get_preamble,
#ifdef CONFIG_IPW2100_MONITOR
 ipw2100_wx_set_crc_check,
 ipw2100_wx_get_crc_check,
#else    /* CONFIG_IPW2100_MONITOR */
 NULL,
 NULL,
#endif    /* CONFIG_IPW2100_MONITOR */
};

/*
* Get wireless statistics.
* Called by /proc/net/wireless
* Also called by SIOCGIWSTATS
*/
static struct iw_statistics *ipw2100_wx_wireless_stats(struct net_device *dev)
{
 enum {
  POOR = 30,
  FAIR = 60,
  GOOD = 80,
  VERY_GOOD = 90,
```

```
 EXCELLENT = 95,
 PERFECT = 100
};
int rssi_qual;
int tx_qual;
int beacon_qual;

struct ipw2100_priv *priv = ieee80211_priv(dev);
struct iw_statistics *wstats;
u32 rssi, quality, tx_retries, missed_beacons, tx_failures;
u32 ord_len = sizeof(u32);

if (!priv)
 return (struct iw_statistics *)NULL;

wstats = &priv->wstats;

/* if hw is disabled, then ipw2100_get_ordinal() can't be called.
 * ipw2100_wx_wireless_stats seems to be called
before fw is
 * initialized.  STATUS_ASSOCIATED will only be set if the hw is up
 * and associated; if not associcated, the values are all meaningless
 * anyway, so set them all to NULL and INVALID */
if (!(priv->status & STATUS_ASSOCIATED)) {
 wstats->miss.beacon = 0;
 wstats->discard.retries = 0;
 wstats->qual.qual = 0;
 wstats->qual.level = 0;
 wstats->qual.noise = 0;
 wstats->qual.updated = 7;
 wstats->qual.updated |= IW_QUAL_NOISE_INVALID |
    IW_QUAL_QUAL_INVALID | IW_QUAL_LEVEL_INVALID;
 return wstats;
}

if (ipw2100_get_ordinal(priv, IPW_ORD_STAT_PERCENT_MISSED_BCNS,
   &missed_beacons, &ord_len))
 goto fail_get_ordinal;

/* If we don't have a connection the quality and level is 0 */
if (!(priv->status & STATUS_ASSOCIATED)) {
 wstats->qual.qual = 0;
 wstats->qual.level = 0;
} else {
 if (ipw2100_get_ordinal(priv, IPW_ORD_RSSI_AVG_CURR,
    &rssi, &ord_len))
  goto fail_get_ordinal;
 wstats->qual.level = rssi + IPW2100_RSSI_TO_DBM;
```

```
 if (rssi < 10)
  rssi_qual
= rssi * POOR / 10;
 else if (rssi < 15)
  rssi_qual = (rssi - 10) * (FAIR - POOR) / 5 + POOR;
 else if (rssi < 20)
  rssi_qual = (rssi - 15) * (GOOD - FAIR) / 5 + FAIR;
 else if (rssi < 30)
  rssi_qual = (rssi - 20) * (VERY_GOOD - GOOD) /
     10 + GOOD;
 else
  rssi_qual = (rssi - 30) * (PERFECT - VERY_GOOD) /
     10 + VERY_GOOD;

 if (ipw2100_get_ordinal(priv, IPW_ORD_STAT_PERCENT_RETRIES,
    &tx_retries, &ord_len))
  goto fail_get_ordinal;

 if (tx_retries > 75)
  tx_qual = (90 - tx_retries) * POOR / 15;
 else if (tx_retries > 70)
  tx_qual = (75 - tx_retries) * (FAIR - POOR) / 5 + POOR;
 else if (tx_retries > 65)
  tx_qual = (70 - tx_retries) * (GOOD - FAIR) / 5 + FAIR;
 else if (tx_retries > 50)
  tx_qual = (65 - tx_retries) * (VERY_GOOD - GOOD) /
     15 + GOOD;
 else
  tx_qual = (50 - tx_retries) *
     (PERFECT - VERY_GOOD) / 50 + VERY_GOOD;

 if (missed_beacons > 50)
  beacon_qual = (60 - missed_beacons) * POOR / 10;
 else if (missed_beacons
> 40)
  beacon_qual = (50 - missed_beacons) * (FAIR - POOR) /
     10 + POOR;
 else if (missed_beacons > 32)
  beacon_qual = (40 - missed_beacons) * (GOOD - FAIR) /
     18 + FAIR;
 else if (missed_beacons > 20)
  beacon_qual = (32 - missed_beacons) *
     (VERY_GOOD - GOOD) / 20 + GOOD;
 else
  beacon_qual = (20 - missed_beacons) *
     (PERFECT - VERY_GOOD) / 20 + VERY_GOOD;

 quality = min(beacon_qual, min(tx_qual, rssi_qual));
```

```
#ifdef CONFIG_IPW2100_DEBUG
  if (beacon_qual == quality)
   IPW_DEBUG_WX("Quality clamped by Missed Beacons\n");
  else if (tx_qual == quality)
   IPW_DEBUG_WX("Quality clamped by Tx Retries\n");
  else if (quality != 100)
   IPW_DEBUG_WX("Quality clamped by Signal Strength\n");
  else
   IPW_DEBUG_WX("Quality not clamped.\n");
#endif

  wstats->qual.qual = quality;
  wstats->qual.level = rssi + IPW2100_RSSI_TO_DBM;
 }

 wstats->qual.noise = 0;
 wstats->qual.updated = 7;
 wstats->qual.updated |= IW_QUAL_NOISE_INVALID;

 /* FIXME:
 this is percent and not a # */
 wstats->miss.beacon = missed_beacons;

 if (ipw2100_get_ordinal(priv, IPW_ORD_STAT_TX_FAILURES,
    &tx_failures, &ord_len))
  goto fail_get_ordinal;
 wstats->discard.retries = tx_failures;

 return wstats;

     fail_get_ordinal:
 IPW_DEBUG_WX("failed querying ordinals.\n");

 return (struct iw_statistics *)NULL;
}

static struct iw_handler_def ipw2100_wx_handler_def = {
 .standard = ipw2100_wx_handlers,
 .num_standard = sizeof(ipw2100_wx_handlers) / sizeof(iw_handler),
 .num_private = sizeof(ipw2100_private_handler) / sizeof(iw_handler),
 .num_private_args = sizeof(ipw2100_private_args) /
     sizeof(struct iw_priv_args),
 .private = (iw_handler *) ipw2100_private_handler,
 .private_args = (struct iw_priv_args *)ipw2100_private_args,
 .get_wireless_stats = ipw2100_wx_wireless_stats,
};
```

```
static void ipw2100_wx_event_work(struct ipw2100_priv *priv)
{
 union iwreq_data wrqu;
 int len = ETH_ALEN;

 if (priv->status & STATUS_STOPPING)
  return;

 mutex_lock(&priv->action_mutex);

 IPW_DEBUG_WX("enter\n");

 mutex_unlock(&priv->action_mutex);

 wrqu.ap_addr.sa_family
 = ARPHRD_ETHER;

 /* Fetch BSSID from the hardware */
 if (!(priv->status & (STATUS_ASSOCIATING | STATUS_ASSOCIATED)) ||
     priv->status & STATUS_RF_KILL_MASK ||
     ipw2100_get_ordinal(priv, IPW_ORD_STAT_ASSN_AP_BSSID,
    &priv->bssid, &len)) {
  memset(wrqu.ap_addr.sa_data, 0, ETH_ALEN);
 } else {
  /* We now have the BSSID, so can finish setting to the full
   * associated state */
  memcpy(wrqu.ap_addr.sa_data, priv->bssid, ETH_ALEN);
  memcpy(priv->ieee->bssid, priv->bssid, ETH_ALEN);
  priv->status &= ~STATUS_ASSOCIATING;
  priv->status |= STATUS_ASSOCIATED;
  netif_carrier_on(priv->net_dev);
  netif_wake_queue(priv->net_dev);
 }

 if (!(priv->status & STATUS_ASSOCIATED)) {
  IPW_DEBUG_WX("Configuring ESSID\n");
  mutex_lock(&priv->action_mutex);
  /* This is a disassociation event, so kick the firmware to
   * look for another AP */
  if (priv->config & CFG_STATIC_ESSID)
   ipw2100_set_essid(priv,
 priv->essid, priv->essid_len,
       0);
  else
   ipw2100_set_essid(priv, NULL, 0, 0);
  mutex_unlock(&priv->action_mutex);
 }
```

```
 wireless_send_event(priv->net_dev, SIOCGIWAP, &wrqu, NULL);
}

#define IPW2100_FW_MAJOR_VERSION 1
#define IPW2100_FW_MINOR_VERSION 3

#define IPW2100_FW_MINOR(x) ((x & 0xff) >> 8)
#define IPW2100_FW_MAJOR(x) (x & 0xff)

#define IPW2100_FW_VERSION ((IPW2100_FW_MINOR_VERSION << 8) | \
                            IPW2100_FW_MAJOR_VERSION)

#define IPW2100_FW_PREFIX "ipw2100-" __stringify(IPW2100_FW_MAJOR_VERSION) \
"." __stringify(IPW2100_FW_MINOR_VERSION)

#define IPW2100_FW_NAME(x) IPW2100_FW_PREFIX "" x ".fw"

/*

BINARY FIRMWARE HEADER FORMAT

offset     length   desc
0          2        version
2          2        mode == 0:BSS,1:IBSS,2:MONITOR
4          4        fw_len
8          4        uc_len
C          fw_len   firmware data
12 + fw_len uc_len   microcode data

*/

struct ipw2100_fw_header {
 short version;
 short
 mode;
 unsigned int fw_size;
 unsigned int uc_size;
} __attribute__ ((packed));

static int ipw2100_mod_firmware_load(struct ipw2100_fw *fw)
{
 struct ipw2100_fw_header *h =
     (struct ipw2100_fw_header *)fw->fw_entry->data;

 if (IPW2100_FW_MAJOR(h->version) != IPW2100_FW_MAJOR_VERSION) {
  printk(KERN_WARNING DRV_NAME ": Firmware image not compatible "
         "(detected version id of %u). "
         "See Documentation/networking/README.ipw2100\n",
```

```
        h->version);
 return 1;
 }

 fw->version = h->version;
 fw->fw.data = fw->fw_entry->data + sizeof(struct ipw2100_fw_header);
 fw->fw.size = h->fw_size;
 fw->uc.data = fw->fw.data + h->fw_size;
 fw->uc.size = h->uc_size;

 return 0;
}

static int ipw2100_get_firmware(struct ipw2100_priv *priv,
    struct ipw2100_fw *fw)
{
 char *fw_name;
 int rc;

 IPW_DEBUG_INFO("%s: Using hotplug firmware load.\n",
        priv->net_dev->name);

 switch (priv->ieee->iw_mode) {
 case IW_MODE_ADHOC:
  fw_name = IPW2100_FW_NAME("-i");
  break;
#ifdef
 CONFIG_IPW2100_MONITOR
 case IW_MODE_MONITOR:
  fw_name = IPW2100_FW_NAME("-p");
  break;
#endif
 case IW_MODE_INFRA:
 default:
  fw_name = IPW2100_FW_NAME("");
  break;
 }

 rc = request_firmware(&fw->fw_entry, fw_name, &priv->pci_dev->dev);

 if (rc < 0) {
  printk(KERN_ERR DRV_NAME ": "
        "%s: Firmware '%s' not available or load failed.\n",
        priv->net_dev->name, fw_name);
  return rc;
 }
 IPW_DEBUG_INFO("firmware data %p size %zd\n", fw->fw_entry->data,
        fw->fw_entry->size);
```

```
 ipw2100_mod_firmware_load(fw);

 return 0;
}

static void ipw2100_release_firmware(struct ipw2100_priv *priv,
        struct ipw2100_fw *fw)
{
 fw->version = 0;
 if (fw->fw_entry)
  release_firmware(fw->fw_entry);
 fw->fw_entry = NULL;
}

static int ipw2100_get_fwversion(struct ipw2100_priv *priv, char *buf,
     size_t max)
{
 char ver[MAX_FW_VERSION_LEN];
 u32 len = MAX_FW_VERSION_LEN;
 u32 tmp;
 int i;
 /* firmware version is an ascii string (max len of 14) */
 if (ipw2100_get_ordinal(priv,
 IPW_ORD_STAT_FW_VER_NUM, ver, &len))
  return -EIO;
 tmp = max;
 if (len >= max)
  len = max - 1;
 for (i = 0; i < len; i++)
  buf[i] = ver[i];
 buf[i] = '\0';
 return tmp;
}

static int ipw2100_get_ucodeversion(struct ipw2100_priv *priv, char *buf,
        size_t max)
{
 u32 ver;
 u32 len = sizeof(ver);
 /* microcode version is a 32 bit integer */
 if (ipw2100_get_ordinal(priv, IPW_ORD_UCODE_VERSION, &ver, &len))
  return -EIO;
 return snprintf(buf, max, "%08X", ver);
}

/*
* On exit, the firmware will have been freed from the fw list
```

```
*/
static int ipw2100_fw_download(struct ipw2100_priv *priv, struct ipw2100_fw *fw)
{
 /* firmware is constructed of N contiguous entries, each entry is
  * structured as:
  *
  * offset    sie         desc
  * 0         4           address to write to
  * 4         2           length of data run
  * 6         length      data
  */
 unsigned int addr;
 unsigned short len;

 const unsigned char *firmware_data = fw->fw.data;
 unsigned
 int firmware_data_left = fw->fw.size;

 while (firmware_data_left > 0) {
  addr = *(u32 *) (firmware_data);
  firmware_data += 4;
  firmware_data_left -= 4;

  len = *(u16 *) (firmware_data);
  firmware_data += 2;
  firmware_data_left -= 2;

  if (len > 32) {
   printk(KERN_ERR DRV_NAME ": "
          "Invalid firmware run-length of %d bytes\n",
          len);
   return -EINVAL;
  }

  write_nic_memory(priv->net_dev, addr, len, firmware_data);
  firmware_data += len;
  firmware_data_left -= len;
 }

 return 0;
}

struct symbol_alive_response {
 u8 cmd_id;
 u8 seq_num;
 u8 ucode_rev;
 u8 eeprom_valid;
 u16 valid_flags;
```

```
 u8 IEEE_addr[6];
 u16 flags;
 u16 pcb_rev;
 u16 clock_settle_time; // 1us LSB
 u16 powerup_settle_time; // 1us LSB
 u16 hop_settle_time; // 1us LSB
 u8 date[3];  // month, day, year
 u8 time[2];  // hours, minutes
 u8 ucode_valid;
};

static int ipw2100_ucode_download(struct ipw2100_priv *priv,
      struct ipw2100_fw *fw)
{
 struct net_device *dev = priv->net_dev;
 const
 unsigned char *microcode_data = fw->uc.data;
 unsigned int microcode_data_left = fw->uc.size;
 void __iomem *reg = (void __iomem *)dev->base_addr;

 struct symbol_alive_response response;
 int i, j;
 u8 data;

 /* Symbol control */
 write_nic_word(dev, IPW2100_CONTROL_REG, 0x703);
 readl(reg);
 write_nic_word(dev, IPW2100_CONTROL_REG, 0x707);
 readl(reg);

 /* HW config */
 write_nic_byte(dev, 0x210014, 0x72); /* fifo width =16 */
 readl(reg);
 write_nic_byte(dev, 0x210014, 0x72); /* fifo width =16 */
 readl(reg);

 /* EN_CS_ACCESS bit to reset control store pointer */
 write_nic_byte(dev, 0x210000, 0x40);
 readl(reg);
 write_nic_byte(dev, 0x210000, 0x0);
 readl(reg);
 write_nic_byte(dev, 0x210000, 0x40);
 readl(reg);

 /* copy microcode from buffer into Symbol */

 while (microcode_data_left > 0) {
  write_nic_byte(dev, 0x210010, *microcode_data++);
```

```
 write_nic_byte(dev, 0x210010, *microcode_data++);
 microcode_data_left -= 2;
}

/* EN_CS_ACCESS bit to reset the control store
pointer */
write_nic_byte(dev, 0x210000, 0x0);
readl(reg);

/* Enable System (Reg 0)
 * first enable causes garbage in RX FIFO */
write_nic_byte(dev, 0x210000, 0x0);
readl(reg);
write_nic_byte(dev, 0x210000, 0x80);
readl(reg);

/* Reset External Baseband Reg */
write_nic_word(dev, IPW2100_CONTROL_REG, 0x703);
readl(reg);
write_nic_word(dev, IPW2100_CONTROL_REG, 0x707);
readl(reg);

/* HW Config (Reg 5) */
write_nic_byte(dev, 0x210014, 0x72); // fifo width =16
readl(reg);
write_nic_byte(dev, 0x210014, 0x72); // fifo width =16
readl(reg);

/* Enable System (Reg 0)
 * second enable should be OK */
write_nic_byte(dev, 0x210000, 0x00); // clear enable system
readl(reg);
write_nic_byte(dev, 0x210000, 0x80); // set enable system

/* check Symbol is enabled - upped this from 5 as it wasn't always
 * catching the update */
for (i = 0; i < 10; i++) {
 udelay(10);

 /* check Dino is enabled bit */
 read_nic_byte(dev, 0x210000, &data);
 if (data & 0x1)
  break;
}

if
(i == 10) {
 printk(KERN_ERR DRV_NAME ": %s: Error initializing Symbol\n",
```

```
        dev->name);
 return -EIO;
 }

 /* Get Symbol alive response */
 for (i = 0; i < 30; i++) {
  /* Read alive response structure */
  for (j = 0;
       j < (sizeof(struct symbol_alive_response) >> 1); j++)
   read_nic_word(dev, 0x210004, ((u16 *) & response) + j);

  if ((response.cmd_id == 1) && (response.ucode_valid == 0x1))
   break;
  udelay(10);
 }

 if (i == 30) {
  printk(KERN_ERR DRV_NAME
         ": %s: No response from Symbol - hw not alive\n",
         dev->name);
  printk_buf(IPW_DL_ERROR, (u8 *) & response, sizeof(response));
  return -EIO;
 }

 return 0;
}
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/ipw2100.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/*** -*- linux-c -*- ************************************************************

    Driver for Atmel at76c502 at76c504 and at76c506 wireless cards.

        Copyright 2005 Dan Williams and Red Hat, Inc.

   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 software 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 Atmel wireless lan drivers; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
USA

**/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/atmel.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/atmel.h.svn-base
No license file was found, but licenses were detected in source scan.

/*
*
* Copyright (C) 2002 Intersil Americas Inc.
* Copyright (C) 2003-2004 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>_
*
* 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
*
* 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
*
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/isl_38xx.c
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/isl_38xx.c.svn-base
No license file was found, but licenses were detected in source scan.

/* hermes.h
*
* Driver core for the "Hermes" wireless MAC controller, as used in
* the Lucent Orinoco and Cabletron RoamAbout cards. It should also
* work on the hfa3841 and hfa3842 MAC controller chips used in the
* Prism I & II chipsets.
*
* This is not a complete driver, just low-level access routines for

* the MAC controller itself.
*
* Based on the prism2 driver from Absolute Value Systems' linux-wlan
* project, the Linux wvlan_cs driver, Lucent's HCF-Light
* (wvlan_hcf.c) library, and the NetBSD wireless driver.
*
* Copyright (C) 2000, David Gibson, Linuxcare Australia.
* (C) Copyright David Gibson, IBM Corp. 2001-2003.
*
* Portions taken from hfa384x.h, Copyright (C) 1999 AbsoluteValue Systems, Inc. All Rights Reserved.
*
* This file distributed under the GPL, version 2.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hermes.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/hermes.h.svn-base
No license file was found, but licenses were detected in source scan.

/*
*
* Copyright (C) 2002 Intersil Americas Inc.
* Copyright (C) 2003 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
*
* 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
*
* 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
*
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_mgt.h
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_mgt.h.svn-base
No license file was found, but licenses were detected in source scan.

/* zd_rf.h

*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_rf.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_rf.h.svn-base
No license file was found, but licenses were detected in source scan.

/*
*
* Copyright (C) 2002 Intersil Americas Inc.
*   (C) 2003,2004 Aurelien Alleaume <slts@free.fr>
*   (C) 2003 Herbert Valerio Riedel <hvr@gnu.org>
*   (C) 2003 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
*
* 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
*
* 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
*
 */

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/isl_ioctl.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/isl_ioctl.c

No license file was found, but licenses were detected in source scan.

```
/*
*
*  Copyright (C) 2002 Intersil Americas Inc.
*
*  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
*
*  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
*
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_eth.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_eth.h.svn-base
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/isl_38xx.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/isl_38xx.h.svn-base
No license file was found, but licenses were detected in source scan.

```
/*
*
*  Copyright (C) 2002 Intersil Americas Inc.
*  Copyright (C) 2003 Herbert Valerio Riedel <hvr@gnu.org>
*  Copyright (C) 2003 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
*  Copyright (C) 2003 Aurelien Alleaume <slts@free.fr>
*
*  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
*
*  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
*
*/

Found
 in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_dev.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_dev.h.svn-base
No license file was found, but licenses were detected in source scan.

/* orinoco.c - (formerly known as dldwd_cs.c and orinoco_cs.c)
*
* A driver for Hermes or Prism 2 chipset based PCMCIA wireless
* adaptors, with Lucent/Agere, Intersil or Symbol firmware.
*
* Current maintainers (as of 29 September 2003) are:
* Pavel Roskin <proski AT gnu.org>
* and David Gibson <hermes AT gibson.dropbear.id.au>
*
* (C) Copyright David Gibson, IBM Corporation 2001-2003.
* Copyright (C) 2000 David Gibson, Linuxcare Australia.
* With some help from :
* Copyright (C) 2001 Jean Tourrilhes, HP Labs
* Copyright (C) 2001 Benjamin Herrenschmidt
*
* Based on dummy_cs.c 1.27 2000/06/12 21:27:25
*
* Portions based on wvlan_cs.c 1.0.6, Copyright Andreas Neuhaus <andy
* AT fasta.fh-dortmund.de>
*   http://www.stud.fh-dortmund.de/~andy/wvlan/
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance
 with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* The initial developer of the original code is David A. Hinds
* <dahinds AT users.sourceforge.net>. Portions created by David
* A. Hinds are Copyright (C) 1999 David A. Hinds. All Rights
* Reserved.

```
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above.  If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace
 them with the notice and
* other provisions required by the GPL.  If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.  */

/*
* TODO
* o Handle de-encapsulation within network layer, provide 802.11
*   headers (patch from Thomas 'Dent' Mirlacher)
* o Fix possible races in SPY handling.
* o Disconnect wireless extensions from fundamental configuration.
* o (maybe) Software WEP support (patch from Stano Meduna).
* o (maybe) Use multiple Tx buffers - driver handling queue
*   rather than firmware.
*/

/* Locking and synchronization:
*
* The basic principle is that everything is serialized through a
* single spinlock, priv->lock.  The lock is used in user, bh and irq
* context, so when taken outside hardirq context it should always be
* taken with interrupts disabled.  The lock protects both the
* hardware and the struct orinoco_private.
*
* Another flag, priv->hw_unavailable indicates that the
 hardware is
* unavailable for an extended period of time (e.g. suspended, or in
* the middle of a hard reset).  This flag is protected by the
* spinlock.  All code which touches the hardware should check the
* flag after taking the lock, and if it is set, give up on whatever
* they are doing and drop the lock again.  The orinoco_lock()
* function handles this (it unlocks and returns -EBUSY if
* hw_unavailable is non-zero).
*/

#define DRIVER_NAME "orinoco"

#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
```

```
#include <linux/netdevice.h>
#include <linux/etherdevice.h>
#include <linux/ethtool.h>
#include <linux/if_arp.h>
#include <linux/wireless.h>
#include <net/iw_handler.h>
#include <net/ieee80211.h>

#include "hermes_rid.h"
#include "orinoco.h"

/********************************************************************/
/* Module information                                               */
/********************************************************************/

MODULE_AUTHOR("Pavel
 Roskin <proski@gnu.org> & David Gibson <hermes@gibson.dropbear.id.au>");
MODULE_DESCRIPTION("Driver for Lucent Orinoco, Prism II based and similar wireless cards");
MODULE_LICENSE("Dual MPL/GPL");

/* Level of debugging. Used in the macros in orinoco.h */
#ifdef ORINOCO_DEBUG
int orinoco_debug = ORINOCO_DEBUG;
module_param(orinoco_debug, int, 0644);
MODULE_PARM_DESC(orinoco_debug, "Debug level");
EXPORT_SYMBOL(orinoco_debug);
#endif

static int suppress_linkstatus; /* = 0 */
module_param(suppress_linkstatus, bool, 0644);
MODULE_PARM_DESC(suppress_linkstatus, "Don't log link status changes");
static int ignore_disconnect; /* = 0 */
module_param(ignore_disconnect, int, 0644);
MODULE_PARM_DESC(ignore_disconnect, "Don't report lost link to the network layer");

static int force_monitor; /* = 0 */
module_param(force_monitor, int, 0644);
MODULE_PARM_DESC(force_monitor, "Allow monitor mode for all firmware versions");

/********************************************************************/
/*
 Compile time configuration and compatibility stuff              */
/********************************************************************/

/* We do this this way to avoid ifdefs in the actual code */
#ifdef WIRELESS_SPY
#define SPY_NUMBER(priv) (priv->spy_data.spy_number)
#else
```

```
#define SPY_NUMBER(priv) 0
#endif /* WIRELESS_SPY */

/********************************************************************/
/* Internal constants                                               */
/********************************************************************/

/* 802.2 LLC/SNAP header used for Ethernet encapsulation over 802.11 */
static const u8 encaps_hdr[] = {0xaa, 0xaa, 0x03, 0x00, 0x00, 0x00};
#define ENCAPS_OVERHEAD  (sizeof(encaps_hdr) + 2)

#define ORINOCO_MIN_MTU  256
#define ORINOCO_MAX_MTU  (IEEE80211_DATA_LEN - ENCAPS_OVERHEAD)

#define SYMBOL_MAX_VER_LEN (14)
#define USER_BAP  0
#define IRQ_BAP   1
#define MAX_IRQLOOPS_PER_IRQ 10
#define MAX_IRQLOOPS_PER_JIFFY (20000/HZ)
 /* Based on a guestimate of
       * how many events the
       * device could
       * legitimately generate */
#define SMALL_KEY_SIZE  5
#define LARGE_KEY_SIZE  13
#define TX_NICBUF_SIZE_BUG 1585  /* Bug in Symbol firmware */

#define DUMMY_FID  0xFFFF

/*#define MAX_MULTICAST(priv) (priv->firmware_type == FIRMWARE_TYPE_AGERE ? \
 HERMES_MAX_MULTICAST : 0)*/
#define MAX_MULTICAST(priv) (HERMES_MAX_MULTICAST)

#define ORINOCO_INTEN   (HERMES_EV_RX | HERMES_EV_ALLOC \
    | HERMES_EV_TX | HERMES_EV_TXEXC \
    | HERMES_EV_WTERR | HERMES_EV_INFO \
    | HERMES_EV_INFDROP )

#define MAX_RID_LEN 1024

static const struct iw_handler_def orinoco_handler_def;
static const struct ethtool_ops orinoco_ethtool_ops;

/********************************************************************/
/* Data tables                                                      */
/********************************************************************/

/* The frequency of each channel in MHz */
```

```
static const
 long channel_frequency[] = {
 2412, 2417, 2422, 2427, 2432, 2437, 2442,
 2447, 2452, 2457, 2462, 2467, 2472, 2484
};
#define NUM_CHANNELS ARRAY_SIZE(channel_frequency)

/* This tables gives the actual meanings of the bitrate IDs returned
* by the firmware. */
static struct {
 int bitrate; /* in 100s of kilobits */
 int automatic;
 u16 agere_txratectrl;
 u16 intersil_txratectrl;
} bitrate_table[] = {
 {110, 1,  3, 15}, /* Entry 0 is the default */
 {10,  0,  1,  1},
 {10,  1,  1,  1},
 {20,  0,  2,  2},
 {20,  1,  6,  3},
 {55,  0,  4,  4},
 {55,  1,  7,  7},
 {110, 0,  5,  8},
};
#define BITRATE_TABLE_SIZE ARRAY_SIZE(bitrate_table)

/********************************************************************/
/* Data types                                                       */
/********************************************************************/

/* Beginning of the Tx descriptor, used in TxExc handling */
struct hermes_txexc_data {
 struct hermes_tx_descriptor desc;
 __le16 frame_ctl;
 __le16
 duration_id;
 u8 addr1[ETH_ALEN];
} __attribute__ ((packed));

/* Rx frame header except compatibility 802.3 header */
struct hermes_rx_descriptor {
 /* Control */
 __le16 status;
 __le32 time;
 u8 silence;
 u8 signal;
 u8 rate;
 u8 rxflow;
```

```
 __le32 reserved;

 /* 802.11 header */
 __le16 frame_ctl;
 __le16 duration_id;
 u8 addr1[ETH_ALEN];
 u8 addr2[ETH_ALEN];
 u8 addr3[ETH_ALEN];
 __le16 seq_ctl;
 u8 addr4[ETH_ALEN];

 /* Data length */
 __le16 data_len;
} __attribute__ ((packed));

/********************************************************************/
/* Function prototypes                                              */
/********************************************************************/

static int __orinoco_program_rids(struct net_device *dev);
static void __orinoco_set_multicast_list(struct net_device *dev);

/********************************************************************/
/* Internal helper functions                                        */
/********************************************************************/

static
 inline void set_port_type(struct orinoco_private *priv)
{
 switch (priv->iw_mode) {
 case IW_MODE_INFRA:
  priv->port_type = 1;
  priv->createibss = 0;
  break;
 case IW_MODE_ADHOC:
  if (priv->prefer_port3) {
   priv->port_type = 3;
   priv->createibss = 0;
  } else {
   priv->port_type = priv->ibss_port;
   priv->createibss = 1;
  }
  break;
 case IW_MODE_MONITOR:
  priv->port_type = 3;
  priv->createibss = 0;
  break;
 default:
```

```
  printk(KERN_ERR "%s: Invalid priv->iw_mode in set_port_type()\n",
         priv->ndev->name);
 }
}

/********************************************************************/
/* Device methods                                                   */
/********************************************************************/

static int orinoco_open(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 unsigned long flags;
 int err;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 err
 = __orinoco_up(dev);

 if (! err)
  priv->open = 1;

 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_stop(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int err = 0;

 /* We mustn't use orinoco_lock() here, because we need to be
    able to close the interface even if hw_unavailable is set
    (e.g. as we're released after a PC Card removal) */
 spin_lock_irq(&priv->lock);

 priv->open = 0;

 err = __orinoco_down(dev);

 spin_unlock_irq(&priv->lock);

 return err;
}
```

```
static struct net_device_stats *orinoco_get_stats(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);

 return &priv->stats;
}

static struct iw_statistics *orinoco_get_wireless_stats(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 struct iw_statistics *wstats = &priv->wstats;
 int err;
 unsigned long flags;

 if (! netif_device_present(dev)) {
  printk(KERN_WARNING "%s: get_wireless_stats() called
 while device not present\n",
        dev->name);
  return NULL; /* FIXME: Can we do better than this? */
 }

 /* If busy, return the old stats.  Returning NULL may cause
  * the interface to disappear from /proc/net/wireless */
 if (orinoco_lock(priv, &flags) != 0)
  return wstats;

 /* We can't really wait for the tallies inquiry command to
  * complete, so we just use the previous results and trigger
  * a new tallies inquiry command for next time - Jean II */
 /* FIXME: Really we should wait for the inquiry to come back -
  * as it is the stats we give don't make a whole lot of sense.
  * Unfortunately, it's not clear how to do that within the
  * wireless extensions framework: I think we're in user
  * context, but a lock seems to be held by the time we get in
  * here so we're not safe to sleep here. */
 hermes_inquire(hw, HERMES_INQ_TALLIES);

 if (priv->iw_mode == IW_MODE_ADHOC) {
  memset(&wstats->qual, 0, sizeof(wstats->qual));
  /* If a spy address is defined, we report
 stats of the
   * first spy address - Jean II */
  if (SPY_NUMBER(priv)) {
   wstats->qual.qual = priv->spy_data.spy_stat[0].qual;
   wstats->qual.level = priv->spy_data.spy_stat[0].level;
   wstats->qual.noise = priv->spy_data.spy_stat[0].noise;
   wstats->qual.updated = priv->spy_data.spy_stat[0].updated;
```

```
 }
} else {
 struct {
  __le16 qual, signal, noise, unused;
 } __attribute__ ((packed)) cq;

 err = HERMES_READ_RECORD(hw, USER_BAP,
     HERMES_RID_COMMSQUALITY, &cq);

 if (!err) {
  wstats->qual.qual = (int)le16_to_cpu(cq.qual);
  wstats->qual.level = (int)le16_to_cpu(cq.signal) - 0x95;
  wstats->qual.noise = (int)le16_to_cpu(cq.noise) - 0x95;
  wstats->qual.updated = 7;
 }
}

 orinoco_unlock(priv, &flags);
 return wstats;
}

static void orinoco_set_multicast_list(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0) {
  printk(KERN_DEBUG "%s: orinoco_set_multicast_list() "

 "called when hw_unavailable\n", dev->name);
  return;
 }

 __orinoco_set_multicast_list(dev);
 orinoco_unlock(priv, &flags);
}

static int orinoco_change_mtu(struct net_device *dev, int new_mtu)
{
 struct orinoco_private *priv = netdev_priv(dev);

 if ( (new_mtu < ORINOCO_MIN_MTU) || (new_mtu > ORINOCO_MAX_MTU) )
  return -EINVAL;

 if ( (new_mtu + ENCAPS_OVERHEAD + IEEE80211_HLEN) >
     (priv->nicbuf_size - ETH_HLEN) )
  return -EINVAL;
```

```
 dev->mtu = new_mtu;

 return 0;
}

/*********************************************************************/
/* Tx path                                                           */
/*********************************************************************/

static int orinoco_xmit(struct sk_buff *skb, struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct net_device_stats *stats = &priv->stats;
 hermes_t *hw = &priv->hw;
 int err = 0;
 u16 txfid = priv->txfid;
 struct ethhdr *eh;
 int data_off;
 struct hermes_tx_descriptor desc;
 unsigned
 long flags;

 if (! netif_running(dev)) {
  printk(KERN_ERR "%s: Tx on stopped device!\n",
         dev->name);
  return NETDEV_TX_BUSY;
 }

 if (netif_queue_stopped(dev)) {
  printk(KERN_DEBUG "%s: Tx while transmitter busy!\n",
         dev->name);
  return NETDEV_TX_BUSY;
 }

 if (orinoco_lock(priv, &flags) != 0) {
  printk(KERN_ERR "%s: orinoco_xmit() called while hw_unavailable\n",
         dev->name);
  return NETDEV_TX_BUSY;
 }

 if (! netif_carrier_ok(dev) || (priv->iw_mode == IW_MODE_MONITOR)) {
  /* Oops, the firmware hasn't established a connection,
                 silently drop the packet (this seems to be the
                 safest approach). */
  goto drop;
 }

 /* Check packet length */
```

```
if (skb->len < ETH_HLEN)
 goto drop;

eh = (struct ethhdr *)skb->data;

memset(&desc, 0, sizeof(desc));
desc.tx_control = cpu_to_le16(HERMES_TXCTRL_TX_OK | HERMES_TXCTRL_TX_EX);
err = hermes_bap_pwrite(hw, USER_BAP, &desc, sizeof(desc), txfid, 0);
if (err) {
 if
(net_ratelimit())
  printk(KERN_ERR "%s: Error %d writing Tx descriptor "
        "to BAP\n", dev->name, err);
 goto busy;
}

/* Clear the 802.11 header and data length fields - some
 * firmwares (e.g. Lucent/Agere 8.xx) appear to get confused
 * if this isn't done. */
hermes_clear_words(hw, HERMES_DATA0,
     HERMES_802_3_OFFSET - HERMES_802_11_OFFSET);

/* Encapsulate Ethernet-II frames */
if (ntohs(eh->h_proto) > ETH_DATA_LEN) { /* Ethernet-II frame */
 struct header_struct {
  struct ethhdr eth; /* 802.3 header */
  u8 encap[6];  /* 802.2 header */
 } __attribute__ ((packed)) hdr;

 /* Strip destination and source from the data */
 skb_pull(skb, 2 * ETH_ALEN);
 data_off = HERMES_802_2_OFFSET + sizeof(encaps_hdr);

 /* And move them to a separate header */
 memcpy(&hdr.eth, eh, 2 * ETH_ALEN);
 hdr.eth.h_proto = htons(sizeof(encaps_hdr) + skb->len);
 memcpy(hdr.encap, encaps_hdr, sizeof(encaps_hdr));

 err = hermes_bap_pwrite(hw, USER_BAP, &hdr, sizeof(hdr),
    txfid,
HERMES_802_3_OFFSET);
 if (err) {
  if (net_ratelimit())
   printk(KERN_ERR "%s: Error %d writing packet "
         "header to BAP\n", dev->name, err);
  goto busy;
 }
} else { /* IEEE 802.3 frame */
```

```
 data_off = HERMES_802_3_OFFSET;
 }

 err = hermes_bap_pwrite(hw, USER_BAP, skb->data, skb->len,
    txfid, data_off);
 if (err) {
  printk(KERN_ERR "%s: Error %d writing packet to BAP\n",
         dev->name, err);
  goto busy;
 }

 /* Finally, we actually initiate the send */
 netif_stop_queue(dev);

 err = hermes_docmd_wait(hw, HERMES_CMD_TX | HERMES_CMD_RECL,
    txfid, NULL);
 if (err) {
  netif_start_queue(dev);
  if (net_ratelimit())
   printk(KERN_ERR "%s: Error %d transmitting packet\n",
    dev->name, err);
  goto busy;
 }

 dev->trans_start = jiffies;
 stats->tx_bytes += data_off + skb->len;
 goto ok;

drop:
 stats->tx_errors++;
 stats->tx_dropped++;

ok:
 orinoco_unlock(priv, &flags);
 dev_kfree_skb(skb);
 return NETDEV_TX_OK;

busy:
 if (err == -EIO)
  schedule_work(&priv->reset_work);
 orinoco_unlock(priv,
 &flags);
 return NETDEV_TX_BUSY;
}

static void __orinoco_ev_alloc(struct net_device *dev, hermes_t *hw)
{
 struct orinoco_private *priv = netdev_priv(dev);
```

```
 u16 fid = hermes_read_regn(hw, ALLOCFID);

 if (fid != priv->txfid) {
  if (fid != DUMMY_FID)
   printk(KERN_WARNING "%s: Allocate event on unexpected fid (%04X)\n",
          dev->name, fid);
  return;
 }

 hermes_write_regn(hw, ALLOCFID, DUMMY_FID);
}

static void __orinoco_ev_tx(struct net_device *dev, hermes_t *hw)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct net_device_stats *stats = &priv->stats;

 stats->tx_packets++;

 netif_wake_queue(dev);

 hermes_write_regn(hw, TXCOMPLFID, DUMMY_FID);
}

static void __orinoco_ev_txexc(struct net_device *dev, hermes_t *hw)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct net_device_stats *stats = &priv->stats;
 u16 fid = hermes_read_regn(hw, TXCOMPLFID);
 u16 status;
 struct hermes_txexc_data hdr;
 int err
 = 0;

 if (fid == DUMMY_FID)
  return; /* Nothing's really happened */

 /* Read part of the frame header - we need status and addr1 */
 err = hermes_bap_pread(hw, IRQ_BAP, &hdr,
          sizeof(struct hermes_txexc_data),
          fid, 0);

 hermes_write_regn(hw, TXCOMPLFID, DUMMY_FID);
 stats->tx_errors++;

 if (err) {
  printk(KERN_WARNING "%s: Unable to read descriptor on Tx error "
         "(FID=%04X error %d)\n",
```

```
        dev->name, fid, err);
 return;
 }

 DEBUG(1, "%s: Tx error, err %d (FID=%04X)\n", dev->name,
       err, fid);

 /* We produce a TXDROP event only for retry or lifetime
  * exceeded, because that's the only status that really mean
  * that this particular node went away.
  * Other errors means that *we* screwed up. - Jean II */
 status = le16_to_cpu(hdr.desc.status);
 if (status & (HERMES_TXSTAT_RETRYERR | HERMES_TXSTAT_AGEDERR)) {
  union iwreq_data wrqu;

  /* Copy 802.11 dest address.
   * We use the 802.11 header because the frame may
   * not be 802.3
 or may be mangled...
   * In Ad-Hoc mode, it will be the node address.
   * In managed mode, it will be most likely the AP addr
   * User space will figure out how to convert it to
   * whatever it needs (IP address or else).
   * - Jean II */
  memcpy(wrqu.addr.sa_data, hdr.addr1, ETH_ALEN);
  wrqu.addr.sa_family = ARPHRD_ETHER;

  /* Send event to user space */
  wireless_send_event(dev, IWEVTXDROP, &wrqu, NULL);
 }

 netif_wake_queue(dev);
}

static void orinoco_tx_timeout(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct net_device_stats *stats = &priv->stats;
 struct hermes *hw = &priv->hw;

 printk(KERN_WARNING "%s: Tx timeout! "
        "ALLOCFID=%04x, TXCOMPLFID=%04x, EVSTAT=%04x\n",
        dev->name, hermes_read_regn(hw, ALLOCFID),
        hermes_read_regn(hw, TXCOMPLFID), hermes_read_regn(hw, EVSTAT));

 stats->tx_errors++;

 schedule_work(&priv->reset_work);
```

```
}

/*********************************************************************/
/* Rx
 path (data frames)                                */
/*********************************************************************/

/* Does the frame have a SNAP header indicating it should be
* de-encapsulated to Ethernet-II? */
static inline int is_ethersnap(void *_hdr)
{
 u8 *hdr = _hdr;

 /* We de-encapsulate all packets which, a) have SNAP headers
  * (i.e. SSAP=DSAP=0xaa and CTRL=0x3 in the 802.2 LLC header
  * and where b) the OUI of the SNAP header is 00:00:00 or
  * 00:00:f8 - we need both because different APs appear to use
  * different OUIs for some reason */
 return (memcmp(hdr, &encaps_hdr, 5) == 0)
  && ( (hdr[5] == 0x00) || (hdr[5] == 0xf8) );
}

static inline void orinoco_spy_gather(struct net_device *dev, u_char *mac,
        int level, int noise)
{
 struct iw_quality wstats;
 wstats.level = level - 0x95;
 wstats.noise = noise - 0x95;
 wstats.qual = (level > noise) ? (level - noise) : 0;
 wstats.updated = 7;
 /* Update spy records */
 wireless_spy_update(dev,
 mac, &wstats);
}

static void orinoco_stat_gather(struct net_device *dev,
    struct sk_buff *skb,
    struct hermes_rx_descriptor *desc)
{
 struct orinoco_private *priv = netdev_priv(dev);

 /* Using spy support with lots of Rx packets, like in an
  * infrastructure (AP), will really slow down everything, because
  * the MAC address must be compared to each entry of the spy list.
  * If the user really asks for it (set some address in the
  * spy list), we do it, but he will pay the price.
  * Note that to get here, you need both WIRELESS_SPY
  * compiled in AND some addresses in the list !!!
```

```
 */
/* Note : gcc will optimise the whole section away if
 * WIRELESS_SPY is not defined... - Jean II */
if (SPY_NUMBER(priv)) {
 orinoco_spy_gather(dev, skb->mac.raw + ETH_ALEN,
     desc->signal, desc->silence);
}
}

/*
* orinoco_rx_monitor - handle received monitor frames.
*
* Arguments:
* dev  network device
* rxfid  received FID
* desc  rx descriptor of the frame
*
* Call context:
 interrupt
*/
static void orinoco_rx_monitor(struct net_device *dev, u16 rxfid,
         struct hermes_rx_descriptor *desc)
{
 u32 hdrlen = 30; /* return full header by default */
 u32 datalen = 0;
 u16 fc;
 int err;
 int len;
 struct sk_buff *skb;
 struct orinoco_private *priv = netdev_priv(dev);
 struct net_device_stats *stats = &priv->stats;
 hermes_t *hw = &priv->hw;

 len = le16_to_cpu(desc->data_len);

 /* Determine the size of the header and the data */
 fc = le16_to_cpu(desc->frame_ctl);
 switch (fc & IEEE80211_FCTL_FTYPE) {
 case IEEE80211_FTYPE_DATA:
  if ((fc & IEEE80211_FCTL_TODS)
      && (fc & IEEE80211_FCTL_FROMDS))
   hdrlen = 30;
  else
   hdrlen = 24;
  datalen = len;
  break;
 case IEEE80211_FTYPE_MGMT:
  hdrlen = 24;
```

```
 datalen = len;
 break;
case IEEE80211_FTYPE_CTL:
 switch (fc & IEEE80211_FCTL_STYPE) {
 case IEEE80211_STYPE_PSPOLL:
 case IEEE80211_STYPE_RTS:
 case IEEE80211_STYPE_CFEND:
 case IEEE80211_STYPE_CFENDACK:
  hdrlen = 16;
  break;
 case
IEEE80211_STYPE_CTS:
 case IEEE80211_STYPE_ACK:
  hdrlen = 10;
  break;
 }
 break;
default:
 /* Unknown frame type */
 break;
}

/* sanity check the length */
if (datalen > IEEE80211_DATA_LEN + 12) {
 printk(KERN_DEBUG "%s: oversized monitor frame, "
       "data length = %d\n", dev->name, datalen);
 stats->rx_length_errors++;
 goto update_stats;
}

skb = dev_alloc_skb(hdrlen + datalen);
if (!skb) {
 printk(KERN_WARNING "%s: Cannot allocate skb for monitor frame\n",
       dev->name);
 goto update_stats;
}

/* Copy the 802.11 header to the skb */
memcpy(skb_put(skb, hdrlen), &(desc->frame_ctl), hdrlen);
skb->mac.raw = skb->data;

/* If any, copy the data from the card to the skb */
if (datalen > 0) {
 err = hermes_bap_pread(hw, IRQ_BAP, skb_put(skb, datalen),
         ALIGN(datalen, 2), rxfid,
         HERMES_802_2_OFFSET);
 if (err) {
  printk(KERN_ERR "%s: error %d reading monitor frame\n",
```

```
        dev->name, err);
  goto drop;
 }
}

skb->dev
= dev;
skb->ip_summed = CHECKSUM_NONE;
skb->pkt_type = PACKET_OTHERHOST;
skb->protocol = __constant_htons(ETH_P_802_2);

dev->last_rx = jiffies;
stats->rx_packets++;
stats->rx_bytes += skb->len;

netif_rx(skb);
return;

drop:
dev_kfree_skb_irq(skb);
update_stats:
stats->rx_errors++;
stats->rx_dropped++;
}

static void __orinoco_ev_rx(struct net_device *dev, hermes_t *hw)
{
struct orinoco_private *priv = netdev_priv(dev);
struct net_device_stats *stats = &priv->stats;
struct iw_statistics *wstats = &priv->wstats;
struct sk_buff *skb = NULL;
u16 rxfid, status, fc;
int length;
struct hermes_rx_descriptor desc;
struct ethhdr *hdr;
int err;

rxfid = hermes_read_regn(hw, RXFID);

err = hermes_bap_pread(hw, IRQ_BAP, &desc, sizeof(desc),
        rxfid, 0);
if (err) {
 printk(KERN_ERR "%s: error %d reading Rx descriptor. "
       "Frame dropped.\n", dev->name, err);
 goto update_stats;
}

status = le16_to_cpu(desc.status);
```

```
if (status & HERMES_RXSTAT_BADCRC)
{
 DEBUG(1, "%s: Bad CRC on Rx. Frame dropped.\n",
       dev->name);
 stats->rx_crc_errors++;
 goto update_stats;
}

/* Handle frames in monitor mode */
if (priv->iw_mode == IW_MODE_MONITOR) {
 orinoco_rx_monitor(dev, rxfid, &desc);
 return;
}

if (status & HERMES_RXSTAT_UNDECRYPTABLE) {
 DEBUG(1, "%s: Undecryptable frame on Rx. Frame dropped.\n",
       dev->name);
 wstats->discard.code++;
 goto update_stats;
}

length = le16_to_cpu(desc.data_len);
fc = le16_to_cpu(desc.frame_ctl);

/* Sanity checks */
if (length < 3) { /* No for even an 802.2 LLC header */
 /* At least on Symbol firmware with PCF we get quite a
                lot of these legitimately - Poll frames with no
                data. */
 return;
}
if (length > IEEE80211_DATA_LEN) {
 printk(KERN_WARNING "%s: Oversized frame received (%d bytes)\n",
        dev->name, length);
 stats->rx_length_errors++;
 goto update_stats;
}

/* We need space for the packet data itself, plus an ethernet

  header, plus 2 bytes so we can align the IP header on a
   32bit boundary, plus 1 byte so we can read in odd length
   packets from the card, which has an IO granularity of 16
   bits */
skb = dev_alloc_skb(length+ETH_HLEN+2+1);
if (!skb) {
 printk(KERN_WARNING "%s: Can't allocate skb for Rx\n",
```

```
       dev->name);
 goto update_stats;
}

/* We'll prepend the header, so reserve space for it.  The worst
   case is no decapsulation, when 802.3 header is prepended and
   nothing is removed.  2 is for aligning the IP header.  */
skb_reserve(skb, ETH_HLEN + 2);

err = hermes_bap_pread(hw, IRQ_BAP, skb_put(skb, length),
        ALIGN(length, 2), rxfid,
        HERMES_802_2_OFFSET);
if (err) {
 printk(KERN_ERR "%s: error %d reading frame. "
       "Frame dropped.\n", dev->name, err);
 goto drop;
}

/* Handle decapsulation
 * In most cases, the firmware tell us about SNAP frames.
 * For some reason, the SNAP frames sent by LinkSys APs
 * are not properly recognised
by most firmwares.
 * So, check ourselves */
if (length >= ENCAPS_OVERHEAD &&
    (((status & HERMES_RXSTAT_MSGTYPE) == HERMES_RXSTAT_1042) ||
     ((status & HERMES_RXSTAT_MSGTYPE) == HERMES_RXSTAT_TUNNEL) ||
     is_ethersnap(skb->data))) {
 /* These indicate a SNAP within 802.2 LLC within
    802.11 frame which we'll need to de-encapsulate to
    the original EthernetII frame. */
 hdr = (struct ethhdr *)skb_push(skb, ETH_HLEN - ENCAPS_OVERHEAD);
} else {
 /* 802.3 frame - prepend 802.3 header as is */
 hdr = (struct ethhdr *)skb_push(skb, ETH_HLEN);
 hdr->h_proto = htons(length);
}
memcpy(hdr->h_dest, desc.addr1, ETH_ALEN);
if (fc & IEEE80211_FCTL_FROMDS)
 memcpy(hdr->h_source, desc.addr3, ETH_ALEN);
else
 memcpy(hdr->h_source, desc.addr2, ETH_ALEN);

dev->last_rx = jiffies;
skb->dev = dev;
skb->protocol = eth_type_trans(skb, dev);
skb->ip_summed = CHECKSUM_NONE;
if (fc & IEEE80211_FCTL_TODS)
```

```
 skb->pkt_type = PACKET_OTHERHOST;

 /* Process the wireless
 stats if needed */
 orinoco_stat_gather(dev, skb, &desc);

 /* Pass the packet to the networking stack */
 netif_rx(skb);
 stats->rx_packets++;
 stats->rx_bytes += length;

 return;

drop:
 dev_kfree_skb_irq(skb);
update_stats:
 stats->rx_errors++;
 stats->rx_dropped++;
}

/********************************************************************/
/* Rx path (info frames)                                            */
/********************************************************************/

static void print_linkstatus(struct net_device *dev, u16 status)
{
 char * s;

 if (suppress_linkstatus)
  return;

 switch (status) {
 case HERMES_LINKSTATUS_NOT_CONNECTED:
  s = "Not Connected";
  break;
 case HERMES_LINKSTATUS_CONNECTED:
  s = "Connected";
  break;
 case HERMES_LINKSTATUS_DISCONNECTED:
  s = "Disconnected";
  break;
 case HERMES_LINKSTATUS_AP_CHANGE:
  s = "AP Changed";
  break;
 case HERMES_LINKSTATUS_AP_OUT_OF_RANGE:
  s = "AP Out of Range";
  break;
 case HERMES_LINKSTATUS_AP_IN_RANGE:
```

```
 s
= "AP In Range";
 break;
case HERMES_LINKSTATUS_ASSOC_FAILED:
 s = "Association Failed";
 break;
default:
 s = "UNKNOWN";
}

printk(KERN_INFO "%s: New link status: %s (%04x)\n",
       dev->name, s, status);
}

/* Search scan results for requested BSSID, join it if found */
static void orinoco_join_ap(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
 int err;
 unsigned long flags;
 struct join_req {
  u8 bssid[ETH_ALEN];
  __le16 channel;
 } __attribute__ ((packed)) req;
 const int atom_len = offsetof(struct prism2_scan_apinfo, atim);
 struct prism2_scan_apinfo *atom = NULL;
 int offset = 4;
 int found = 0;
 u8 *buf;
 u16 len;

 /* Allocate buffer for scan results */
 buf = kmalloc(MAX_SCAN_LEN, GFP_KERNEL);
 if (! buf)
  return;

 if (orinoco_lock(priv, &flags) != 0)
  goto fail_lock;

 /* Sanity checks in case user changed something in the meantime */
 if (! priv->bssid_fixed)
  goto out;

 if (strlen(priv->desired_essid)
 == 0)
  goto out;
```

```
 /* Read scan results from the firmware */
 err = hermes_read_ltv(hw, USER_BAP,
        HERMES_RID_SCANRESULTSTABLE,
        MAX_SCAN_LEN, &len, buf);
 if (err) {
  printk(KERN_ERR "%s: Cannot read scan results\n",
        dev->name);
  goto out;
 }

 len = HERMES_RECLEN_TO_BYTES(len);

 /* Go through the scan results looking for the channel of the AP
  * we were requested to join */
 for (; offset + atom_len <= len; offset += atom_len) {
  atom = (struct prism2_scan_apinfo *) (buf + offset);
  if (memcmp(&atom->bssid, priv->desired_bssid, ETH_ALEN) == 0) {
   found = 1;
   break;
  }
 }

 if (! found) {
  DEBUG(1, "%s: Requested AP not found in scan results\n",
        dev->name);
  goto out;
 }

 memcpy(req.bssid, priv->desired_bssid, ETH_ALEN);
 req.channel = atom->channel; /* both are little-endian */
 err = HERMES_WRITE_RECORD(hw, USER_BAP, HERMES_RID_CNFJOINREQUEST,
      &req);
 if (err)
  printk(KERN_ERR "%s: Error issuing join request\n",
 dev->name);

out:
 orinoco_unlock(priv, &flags);

fail_lock:
 kfree(buf);
}

/* Send new BSSID to userspace */
static void orinoco_send_wevents(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
```

```
 union iwreq_data wrqu;
 int err;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return;

 err = hermes_read_ltv(hw, IRQ_BAP, HERMES_RID_CURRENTBSSID,
        ETH_ALEN, NULL, wrqu.ap_addr.sa_data);
 if (err != 0)
  goto out;

 wrqu.ap_addr.sa_family = ARPHRD_ETHER;

 /* Send event to user space */
 wireless_send_event(dev, SIOCGIWAP, &wrqu, NULL);

out:
 orinoco_unlock(priv, &flags);
}

static void __orinoco_ev_info(struct net_device *dev, hermes_t *hw)
{
 struct orinoco_private *priv = netdev_priv(dev);
 u16 infofid;
 struct {
  __le16 len;
  __le16 type;
 } __attribute__ ((packed)) info;
 int len, type;
 int err;

 /* This is an answer to an INQUIRE command that we did earlier,
  * or an information "event" generated
 by the card
  * The controller return to us a pseudo frame containing
  * the information in question - Jean II */
 infofid = hermes_read_regn(hw, INFOFID);

 /* Read the info frame header - don't try too hard */
 err = hermes_bap_pread(hw, IRQ_BAP, &info, sizeof(info),
         infofid, 0);
 if (err) {
  printk(KERN_ERR "%s: error %d reading info frame. "
        "Frame dropped.\n", dev->name, err);
  return;
 }
```

```
len = HERMES_RECLEN_TO_BYTES(le16_to_cpu(info.len));
type = le16_to_cpu(info.type);

switch (type) {
case HERMES_INQ_TALLIES: {
 struct hermes_tallies_frame tallies;
 struct iw_statistics *wstats = &priv->wstats;

 if (len > sizeof(tallies)) {
  printk(KERN_WARNING "%s: Tallies frame too long (%d bytes)\n",
         dev->name, len);
  len = sizeof(tallies);
 }

 err = hermes_bap_pread(hw, IRQ_BAP, &tallies, len,
          infofid, sizeof(info));
 if (err)
  break;

 /* Increment our various counters */
 /* wstats->discard.nwid - no wrong BSSID stuff
*/
 wstats->discard.code +=
  le16_to_cpu(tallies.RxWEPUndecryptable);
 if (len == sizeof(tallies))
  wstats->discard.code +=
   le16_to_cpu(tallies.RxDiscards_WEPICVError) +
   le16_to_cpu(tallies.RxDiscards_WEPExcluded);
 wstats->discard.misc +=
  le16_to_cpu(tallies.TxDiscardsWrongSA);
 wstats->discard.fragment +=
  le16_to_cpu(tallies.RxMsgInBadMsgFragments);
 wstats->discard.retries +=
  le16_to_cpu(tallies.TxRetryLimitExceeded);
 /* wstats->miss.beacon - no match */
}
break;
case HERMES_INQ_LINKSTATUS: {
 struct hermes_linkstatus linkstatus;
 u16 newstatus;
 int connected;

 if (priv->iw_mode == IW_MODE_MONITOR)
  break;

 if (len != sizeof(linkstatus)) {
  printk(KERN_WARNING "%s: Unexpected size for linkstatus frame (%d bytes)\n",
         dev->name, len);
```

```
 break;
 }

 err = hermes_bap_pread(hw, IRQ_BAP, &linkstatus, len,
        infofid, sizeof(info));
 if (err)
  break;
 newstatus = le16_to_cpu(linkstatus.linkstatus);

 /* Symbol firmware
uses "out of range" to signal that
  * the hostscan frame can be requested.  */
 if (newstatus == HERMES_LINKSTATUS_AP_OUT_OF_RANGE &&
     priv->firmware_type == FIRMWARE_TYPE_SYMBOL &&
     priv->has_hostscan && priv->scan_inprogress) {
  hermes_inquire(hw, HERMES_INQ_HOSTSCAN_SYMBOL);
  break;
 }

 connected = (newstatus == HERMES_LINKSTATUS_CONNECTED)
  || (newstatus == HERMES_LINKSTATUS_AP_CHANGE)
  || (newstatus == HERMES_LINKSTATUS_AP_IN_RANGE);

 if (connected)
  netif_carrier_on(dev);
 else if (!ignore_disconnect)
  netif_carrier_off(dev);

 if (newstatus != priv->last_linkstatus) {
  priv->last_linkstatus = newstatus;
  print_linkstatus(dev, newstatus);
  /* The info frame contains only one word which is the
   * status (see hermes.h). The status is pretty boring
   * in itself, that's why we export the new BSSID...
   * Jean II */
  schedule_work(&priv->wevent_work);
 }
}
break;
case HERMES_INQ_SCAN:
 if (!priv->scan_inprogress && priv->bssid_fixed
&&
     priv->firmware_type == FIRMWARE_TYPE_INTERSIL) {
  schedule_work(&priv->join_work);
  break;
 }
 /* fall through */
case HERMES_INQ_HOSTSCAN:
```

```
 case HERMES_INQ_HOSTSCAN_SYMBOL: {
  /* Result of a scanning. Contains information about
   * cells in the vicinity - Jean II */
  union iwreq_data wrqu;
  unsigned char *buf;

  /* Sanity check */
  if (len > 4096) {
   printk(KERN_WARNING "%s: Scan results too large (%d bytes)\n",
          dev->name, len);
   break;
  }

  /* We are a strict producer. If the previous scan results
   * have not been consumed, we just have to drop this
   * frame. We can't remove the previous results ourselves,
   * that would be *very* racy... Jean II */
  if (priv->scan_result != NULL) {
   printk(KERN_WARNING "%s: Previous scan results not consumed, dropping info frame.\n", dev->name);
   break;
  }

  /* Allocate buffer for results */
  buf = kmalloc(len, GFP_ATOMIC);
  if (buf == NULL)
   /* No memory, so can't printk()... */
   break;

  /*
 Read scan data */
  err = hermes_bap_pread(hw, IRQ_BAP, (void *) buf, len,
           infofid, sizeof(info));
  if (err) {
   kfree(buf);
   break;
  }

#ifdef ORINOCO_DEBUG
  {
   int i;
   printk(KERN_DEBUG "Scan result [%02X", buf[0]);
   for(i = 1; i < (len * 2); i++)
    printk(":%02X", buf[i]);
   printk("]\n");
  }
#endif /* ORINOCO_DEBUG */

  /* Allow the clients to access the results */
```

```
 priv->scan_len = len;
 priv->scan_result = buf;

 /* Send an empty event to user space.
  * We don't send the received data on the event because
  * it would require us to do complex transcoding, and
  * we want to minimise the work done in the irq handler
  * Use a request to extract the data - Jean II */
 wrqu.data.length = 0;
 wrqu.data.flags = 0;
 wireless_send_event(dev, SIOCGIWSCAN, &wrqu, NULL);
 }
 break;
 case HERMES_INQ_SEC_STAT_AGERE:
 /* Security status (Agere specific) */
 /* Ignore this frame for now */
 if (priv->firmware_type == FIRMWARE_TYPE_AGERE)
  break;
 /*
 fall through */
 default:
 printk(KERN_DEBUG "%s: Unknown information frame received: "
        "type 0x%04x, length %d\n", dev->name, type, len);
 /* We don't actually do anything about it */
 break;
 }
}

static void __orinoco_ev_infdrop(struct net_device *dev, hermes_t *hw)
{
 if (net_ratelimit())
  printk(KERN_DEBUG "%s: Information frame lost.\n", dev->name);
}

/********************************************************************/
/* Internal hardware control routines                               */
/********************************************************************/

int __orinoco_up(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
 int err;

 netif_carrier_off(dev); /* just to make sure */

 err = __orinoco_program_rids(dev);
 if (err) {
```

```
 printk(KERN_ERR "%s: Error %d configuring card\n",
        dev->name, err);
 return err;
}

/* Fire things up again */
hermes_set_irqmask(hw, ORINOCO_INTEN);
err =
hermes_enable_port(hw, 0);
if (err) {
 printk(KERN_ERR "%s: Error %d enabling MAC port\n",
        dev->name, err);
 return err;
}

netif_start_queue(dev);

return 0;
}

int __orinoco_down(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
 int err;

 netif_stop_queue(dev);

 if (! priv->hw_unavailable) {
  if (! priv->broken_disableport) {
   err = hermes_disable_port(hw, 0);
   if (err) {
    /* Some firmwares (e.g. Intersil 1.3.x) seem
     * to have problems disabling the port, oh
     * well, too bad. */
    printk(KERN_WARNING "%s: Error %d disabling MAC port\n",
           dev->name, err);
    priv->broken_disableport = 1;
   }
  }
  hermes_set_irqmask(hw, 0);
  hermes_write_regn(hw, EVACK, 0xffff);
 }

 /* firmware will have to reassociate */
 netif_carrier_off(dev);
 priv->last_linkstatus = 0xffff;
```

```
 return 0;
}

static int orinoco_allocate_fid(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct
 hermes *hw = &priv->hw;
 int err;

 err = hermes_allocate(hw, priv->nicbuf_size, &priv->txfid);
 if (err == -EIO && priv->nicbuf_size > TX_NICBUF_SIZE_BUG) {
  /* Try workaround for old Symbol firmware bug */
  printk(KERN_WARNING "%s: firmware ALLOC bug detected "
         "(old Symbol firmware?). Trying to work around... ",
         dev->name);

  priv->nicbuf_size = TX_NICBUF_SIZE_BUG;
  err = hermes_allocate(hw, priv->nicbuf_size, &priv->txfid);
  if (err)
   printk("failed!\n");
  else
   printk("ok.\n");
 }

 return err;
}

int orinoco_reinit_firmware(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
 int err;

 err = hermes_init(hw);
 if (!err)
  err = orinoco_allocate_fid(dev);

 return err;
}

static int __orinoco_hw_set_bitrate(struct orinoco_private *priv)
{
 hermes_t *hw = &priv->hw;
 int err = 0;

 if (priv->bitratemode >= BITRATE_TABLE_SIZE) {
  printk(KERN_ERR "%s: BUG: Invalid bitrate mode %d\n",
```

```
      priv->ndev->name, priv->bitratemode);
 return -EINVAL;
 }

 switch (priv->firmware_type) {
 case FIRMWARE_TYPE_AGERE:
  err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFTXRATECONTROL,
       bitrate_table[priv->bitratemode].agere_txratectrl);
  break;
 case FIRMWARE_TYPE_INTERSIL:
 case FIRMWARE_TYPE_SYMBOL:
  err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFTXRATECONTROL,
       bitrate_table[priv->bitratemode].intersil_txratectrl);
  break;
 default:
  BUG();
 }

 return err;
}

/* Set fixed AP address */
static int __orinoco_hw_set_wap(struct orinoco_private *priv)
{
 int roaming_flag;
 int err = 0;
 hermes_t *hw = &priv->hw;

 switch (priv->firmware_type) {
 case FIRMWARE_TYPE_AGERE:
  /* not supported */
  break;
 case FIRMWARE_TYPE_INTERSIL:
  if (priv->bssid_fixed)
   roaming_flag = 2;
  else
   roaming_flag = 1;

  err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFROAMINGMODE,
       roaming_flag);
  break;
 case FIRMWARE_TYPE_SYMBOL:
  err
```

```
 = HERMES_WRITE_RECORD(hw, USER_BAP,
       HERMES_RID_CNFMANDATORYBSSID_SYMBOL,
       &priv->desired_bssid);
  break;
 }
 return err;
}

/* Change the WEP keys and/or the current keys.  Can be called
* either from __orinoco_hw_setup_wep() or directly from
* orinoco_ioctl_setiwencode().  In the later case the association
* with the AP is not broken (if the firmware can handle it),
* which is needed for 802.1x implementations. */
static int __orinoco_hw_setup_wepkeys(struct orinoco_private *priv)
{
 hermes_t *hw = &priv->hw;
 int err = 0;

 switch (priv->firmware_type) {
 case FIRMWARE_TYPE_AGERE:
  err = HERMES_WRITE_RECORD(hw, USER_BAP,
       HERMES_RID_CNFWEPKEYS_AGERE,
       &priv->keys);
  if (err)
   return err;
  err = hermes_write_wordrec(hw, USER_BAP,
        HERMES_RID_CNFTXKEY_AGERE,
        priv->tx_key);
  if (err)
   return err;
  break;
 case FIRMWARE_TYPE_INTERSIL:
 case FIRMWARE_TYPE_SYMBOL:
  {
   int keylen;
   int i;

   /* Force uniform key
 length to work around firmware bugs */
   keylen = le16_to_cpu(priv->keys[priv->tx_key].len);

   if (keylen > LARGE_KEY_SIZE) {
    printk(KERN_ERR "%s: BUG: Key %d has oversize length %d.\n",
          priv->ndev->name, priv->tx_key, keylen);
    return -E2BIG;
   }

   /* Write all 4 keys */
```

```
  for(i = 0; i < ORINOCO_MAX_KEYS; i++) {
   err = hermes_write_ltv(hw, USER_BAP,
           HERMES_RID_CNFDEFAULTKEY0 + i,
           HERMES_BYTES_TO_RECLEN(keylen),
           priv->keys[i].data);
   if (err)
    return err;
  }

  /* Write the index of the key used in transmission */
  err = hermes_write_wordrec(hw, USER_BAP,
        HERMES_RID_CNFWEPDEFAULTKEYID,
        priv->tx_key);
  if (err)
   return err;
 }
 break;
 }

 return 0;
}

static int __orinoco_hw_setup_wep(struct orinoco_private *priv)
{
 hermes_t *hw = &priv->hw;
 int err = 0;
 int master_wep_flag;
 int auth_flag;

 if (priv->wep_on)
  __orinoco_hw_setup_wepkeys(priv);

 if (priv->wep_restrict)
  auth_flag
 = HERMES_AUTH_SHARED_KEY;
 else
  auth_flag = HERMES_AUTH_OPEN;

 switch (priv->firmware_type) {
 case FIRMWARE_TYPE_AGERE: /* Agere style WEP */
  if (priv->wep_on) {
   /* Enable the shared-key authentication. */
   err = hermes_write_wordrec(hw, USER_BAP,
         HERMES_RID_CNFAUTHENTICATION_AGERE,
         auth_flag);
  }
  err = hermes_write_wordrec(hw, USER_BAP,
        HERMES_RID_CNFWEPENABLED_AGERE,
```

```
        priv->wep_on);
  if (err)
   return err;
  break;

 case FIRMWARE_TYPE_INTERSIL: /* Intersil style WEP */
 case FIRMWARE_TYPE_SYMBOL: /* Symbol style WEP */
  if (priv->wep_on) {
   if (priv->wep_restrict ||
       (priv->firmware_type == FIRMWARE_TYPE_SYMBOL))
    master_wep_flag = HERMES_WEP_PRIVACY_INVOKED |
        HERMES_WEP_EXCL_UNENCRYPTED;
   else
    master_wep_flag = HERMES_WEP_PRIVACY_INVOKED;

   err = hermes_write_wordrec(hw, USER_BAP,
         HERMES_RID_CNFAUTHENTICATION,
         auth_flag);
   if (err)
    return err;
  } else
   master_wep_flag = 0;

  if
 (priv->iw_mode == IW_MODE_MONITOR)
   master_wep_flag |= HERMES_WEP_HOST_DECRYPT;

  /* Master WEP setting : on/off */
  err = hermes_write_wordrec(hw, USER_BAP,
        HERMES_RID_CNFWEPFLAGS_INTERSIL,
        master_wep_flag);
  if (err)
   return err;

  break;
 }

 return 0;
}

static int __orinoco_program_rids(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err;
 struct hermes_idstring idbuf;

 /* Set the MAC address */
```

```
err = hermes_write_ltv(hw, USER_BAP, HERMES_RID_CNFOWNMACADDR,
        HERMES_BYTES_TO_RECLEN(ETH_ALEN), dev->dev_addr);
if (err) {
 printk(KERN_ERR "%s: Error %d setting MAC address\n",
        dev->name, err);
 return err;
}

/* Set up the link mode */
err = hermes_write_wordrec(hw, USER_BAP, HERMES_RID_CNFPORTTYPE,
      priv->port_type);
if (err) {
 printk(KERN_ERR "%s: Error %d setting port type\n",
        dev->name, err);
 return err;
}
/* Set the channel/frequency */
if (priv->channel
!= 0 && priv->iw_mode != IW_MODE_INFRA) {
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFOWNCHANNEL,
       priv->channel);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting channel %d\n",
         dev->name, err, priv->channel);
  return err;
 }
}

if (priv->has_ibss) {
 u16 createibss;

 if ((strlen(priv->desired_essid) == 0) && (priv->createibss)) {
  printk(KERN_WARNING "%s: This firmware requires an "
         "ESSID in IBSS-Ad-Hoc mode.\n", dev->name);
  /* With wvlan_cs, in this case, we would crash.
   * hopefully, this driver will behave better...
   * Jean II */
  createibss = 0;
 } else {
  createibss = priv->createibss;
 }

 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFCREATEIBSS,
       createibss);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting CREATEIBSS\n",
```

```
        dev->name, err);
 return err;
 }
}

/* Set the desired BSSID */
err = __orinoco_hw_set_wap(priv);
if (err) {
 printk(KERN_ERR
"%s: Error %d setting AP address\n",
       dev->name, err);
 return err;
}
/* Set the desired ESSID */
idbuf.len = cpu_to_le16(strlen(priv->desired_essid));
memcpy(&idbuf.val, priv->desired_essid, sizeof(idbuf.val));
/* WinXP wants partner to configure OWNSSID even in IBSS mode. (jimc) */
err = hermes_write_ltv(hw, USER_BAP, HERMES_RID_CNFOWNSSID,
        HERMES_BYTES_TO_RECLEN(strlen(priv->desired_essid)+2),
        &idbuf);
if (err) {
 printk(KERN_ERR "%s: Error %d setting OWNSSID\n",
       dev->name, err);
 return err;
}
err = hermes_write_ltv(hw, USER_BAP, HERMES_RID_CNFDESIREDSSID,
        HERMES_BYTES_TO_RECLEN(strlen(priv->desired_essid)+2),
        &idbuf);
if (err) {
 printk(KERN_ERR "%s: Error %d setting DESIREDSSID\n",
       dev->name, err);
 return err;
}

/* Set the station name */
idbuf.len = cpu_to_le16(strlen(priv->nick));
memcpy(&idbuf.val, priv->nick, sizeof(idbuf.val));
err = hermes_write_ltv(hw, USER_BAP, HERMES_RID_CNFOWNNAME,

     HERMES_BYTES_TO_RECLEN(strlen(priv->nick)+2),
        &idbuf);
if (err) {
 printk(KERN_ERR "%s: Error %d setting nickname\n",
       dev->name, err);
 return err;
}

/* Set AP density */
```

```
if (priv->has_sensitivity) {
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFSYSTEMSCALE,
       priv->ap_density);
 if (err) {
  printk(KERN_WARNING "%s: Error %d setting SYSTEMSCALE.  "
         "Disabling sensitivity control\n",
         dev->name, err);

  priv->has_sensitivity = 0;
 }
}

/* Set RTS threshold */
err = hermes_write_wordrec(hw, USER_BAP, HERMES_RID_CNFRTSTHRESHOLD,
      priv->rts_thresh);
if (err) {
 printk(KERN_ERR "%s: Error %d setting RTS threshold\n",
        dev->name, err);
 return err;
}

/* Set fragmentation threshold or MWO robustness */
if (priv->has_mwo)
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFMWOROBUST_AGERE,
       priv->mwo_robust);
else
 err = hermes_write_wordrec(hw, USER_BAP,

  HERMES_RID_CNFFRAGMENTATIONTHRESHOLD,
       priv->frag_thresh);
if (err) {
 printk(KERN_ERR "%s: Error %d setting fragmentation\n",
        dev->name, err);
 return err;
}

/* Set bitrate */
err = __orinoco_hw_set_bitrate(priv);
if (err) {
 printk(KERN_ERR "%s: Error %d setting bitrate\n",
        dev->name, err);
 return err;
}

/* Set power management */
if (priv->has_pm) {
```

```
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFPMENABLED,
       priv->pm_on);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting up PM\n",
         dev->name, err);
  return err;
 }

 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFMULTICASTRECEIVE,
       priv->pm_mcast);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting up PM\n",
         dev->name, err);
  return err;
 }
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFMAXSLEEPDURATION,
       priv->pm_period);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting up PM\n",

      dev->name, err);
  return err;
 }
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFPMHOLDOVERDURATION,
       priv->pm_timeout);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting up PM\n",
         dev->name, err);
  return err;
 }
}

/* Set preamble - only for Symbol so far... */
if (priv->has_preamble) {
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFPREAMBLE_SYMBOL,
       priv->preamble);
 if (err) {
  printk(KERN_ERR "%s: Error %d setting preamble\n",
         dev->name, err);
  return err;
 }
}
```

```
 /* Set up encryption */
 if (priv->has_wep) {
  err = __orinoco_hw_setup_wep(priv);
  if (err) {
   printk(KERN_ERR "%s: Error %d activating WEP\n",
          dev->name, err);
   return err;
  }
 }

 if (priv->iw_mode == IW_MODE_MONITOR) {
  /* Enable monitor mode */
  dev->type = ARPHRD_IEEE80211;
  err = hermes_docmd_wait(hw, HERMES_CMD_TEST |
         HERMES_TEST_MONITOR, 0, NULL);
 } else {
  /* Disable monitor mode */
  dev->type
 = ARPHRD_ETHER;
  err = hermes_docmd_wait(hw, HERMES_CMD_TEST |
         HERMES_TEST_STOP, 0, NULL);
 }
 if (err)
  return err;

 /* Set promiscuity / multicast*/
 priv->promiscuous = 0;
 priv->mc_count = 0;

 /* FIXME: what about netif_tx_lock */
 __orinoco_set_multicast_list(dev);

 return 0;
}

/* FIXME: return int? */
static void
__orinoco_set_multicast_list(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err = 0;
 int promisc, mc_count;

 /* The Hermes doesn't seem to have an allmulti mode, so we go
  * into promiscuous mode and let the upper levels deal. */
 if ( (dev->flags & IFF_PROMISC) || (dev->flags & IFF_ALLMULTI) ||
      (dev->mc_count > MAX_MULTICAST(priv)) ) {
```

```
 promisc = 1;
 mc_count = 0;
} else {
 promisc = 0;
 mc_count = dev->mc_count;
}

if (promisc != priv->promiscuous) {
 err = hermes_write_wordrec(hw, USER_BAP,
       HERMES_RID_CNFPROMISCUOUSMODE,
       promisc);
 if (err) {
  printk(KERN_ERR
"%s: Error %d setting PROMISCUOUSMODE to 1.\n",
         dev->name, err);
 } else
  priv->promiscuous = promisc;
}

if (! promisc && (mc_count || priv->mc_count) ) {
 struct dev_mc_list *p = dev->mc_list;
 struct hermes_multicast mclist;
 int i;

 for (i = 0; i < mc_count; i++) {
  /* paranoia: is list shorter than mc_count? */
  BUG_ON(! p);
  /* paranoia: bad address size in list? */
  BUG_ON(p->dmi_addrlen != ETH_ALEN);

  memcpy(mclist.addr[i], p->dmi_addr, ETH_ALEN);
  p = p->next;
 }

 if (p)
  printk(KERN_WARNING "%s: Multicast list is "
         "longer than mc_count\n", dev->name);

 err = hermes_write_ltv(hw, USER_BAP, HERMES_RID_CNFGROUPADDRESSES,
          HERMES_BYTES_TO_RECLEN(priv->mc_count * ETH_ALEN),
          &mclist);
 if (err)
  printk(KERN_ERR "%s: Error %d setting multicast list.\n",
         dev->name, err);
 else
  priv->mc_count = mc_count;
}
```

```
 /* Since we can set the promiscuous flag when it wasn't asked
    for,
 make sure the net_device knows about it. */
 if (priv->promiscuous)
  dev->flags |= IFF_PROMISC;
 else
  dev->flags &= ~IFF_PROMISC;
}

/* This must be called from user context, without locks held - use
* schedule_work() */
static void orinoco_reset(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
 int err;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  /* When the hardware becomes available again, whatever
   * detects that is responsible for re-initializing
   * it. So no need for anything further */
  return;

 netif_stop_queue(dev);

 /* Shut off interrupts.  Depending on what state the hardware
  * is in, this might not work, but we'll try anyway */
 hermes_set_irqmask(hw, 0);
 hermes_write_regn(hw, EVACK, 0xffff);

 priv->hw_unavailable++;
 priv->last_linkstatus = 0xffff; /* firmware will have to reassociate */
 netif_carrier_off(dev);

 orinoco_unlock(priv, &flags);

 /* Scanning support: Cleanup of
 driver struct */
 kfree(priv->scan_result);
 priv->scan_result = NULL;
 priv->scan_inprogress = 0;

 if (priv->hard_reset) {
  err = (*priv->hard_reset)(priv);
  if (err) {
   printk(KERN_ERR "%s: orinoco_reset: Error %d "
          "performing hard reset\n", dev->name, err);
```

```
  goto disable;
 }
 }

 err = orinoco_reinit_firmware(dev);
 if (err) {
  printk(KERN_ERR "%s: orinoco_reset: Error %d re-initializing firmware\n",
         dev->name, err);
  goto disable;
 }

 spin_lock_irq(&priv->lock); /* This has to be called from user context */

 priv->hw_unavailable--;

 /* priv->open or priv->hw_unavailable might have changed while
  * we dropped the lock */
 if (priv->open && (! priv->hw_unavailable)) {
  err = __orinoco_up(dev);
  if (err) {
   printk(KERN_ERR "%s: orinoco_reset: Error %d reenabling card\n",
          dev->name, err);
  } else
   dev->trans_start = jiffies;
 }

 spin_unlock_irq(&priv->lock);

 return;
disable:
 hermes_set_irqmask(hw, 0);
 netif_device_detach(dev);
 printk(KERN_ERR
 "%s: Device has been disabled!\n", dev->name);
}

/********************************************************************/
/* Interrupt handler                                                */
/********************************************************************/

static void __orinoco_ev_tick(struct net_device *dev, hermes_t *hw)
{
 printk(KERN_DEBUG "%s: TICK\n", dev->name);
}

static void __orinoco_ev_wterr(struct net_device *dev, hermes_t *hw)
{
 /* This seems to happen a fair bit under load, but ignoring it
```

```
   seems to work fine...*/
 printk(KERN_DEBUG "%s: MAC controller error (WTERR). Ignoring.\n",
        dev->name);
}

irqreturn_t orinoco_interrupt(int irq, void *dev_id)
{
 struct net_device *dev = dev_id;
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int count = MAX_IRQLOOPS_PER_IRQ;
 u16 evstat, events;
 /* These are used to detect a runaway interrupt situation */
 /* If we get more than MAX_IRQLOOPS_PER_JIFFY iterations in a jiffy,

 * we panic and shut down the hardware */
 static int last_irq_jiffy = 0; /* jiffies value the last time
     * we were called */
 static int loops_this_jiffy = 0;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0) {
  /* If hw is unavailable - we don't know if the irq was
   * for us or not */
  return IRQ_HANDLED;
 }

 evstat = hermes_read_regn(hw, EVSTAT);
 events = evstat & hw->inten;
 if (! events) {
  orinoco_unlock(priv, &flags);
  return IRQ_NONE;
 }

 if (jiffies != last_irq_jiffy)
  loops_this_jiffy = 0;
 last_irq_jiffy = jiffies;

 while (events && count--) {
  if (++loops_this_jiffy > MAX_IRQLOOPS_PER_JIFFY) {
   printk(KERN_WARNING "%s: IRQ handler is looping too "
          "much! Resetting.\n", dev->name);
   /* Disable interrupts for now */
   hermes_set_irqmask(hw, 0);
   schedule_work(&priv->reset_work);
   break;
  }
```

```
 /* Check the card hasn't been removed */
 if (! hermes_present(hw)) {
  DEBUG(0, "orinoco_interrupt(): card removed\n");
  break;
 }

 if
(events & HERMES_EV_TICK)
  __orinoco_ev_tick(dev, hw);
 if (events & HERMES_EV_WTERR)
  __orinoco_ev_wterr(dev, hw);
 if (events & HERMES_EV_INFDROP)
  __orinoco_ev_infdrop(dev, hw);
 if (events & HERMES_EV_INFO)
  __orinoco_ev_info(dev, hw);
 if (events & HERMES_EV_RX)
  __orinoco_ev_rx(dev, hw);
 if (events & HERMES_EV_TXEXC)
  __orinoco_ev_txexc(dev, hw);
 if (events & HERMES_EV_TX)
  __orinoco_ev_tx(dev, hw);
 if (events & HERMES_EV_ALLOC)
  __orinoco_ev_alloc(dev, hw);

 hermes_write_regn(hw, EVACK, evstat);

 evstat = hermes_read_regn(hw, EVSTAT);
 events = evstat & hw->inten;
 };

 orinoco_unlock(priv, &flags);
 return IRQ_HANDLED;
}

/********************************************************************/
/* Initialization                                                   */
/********************************************************************/

struct comp_id {
 u16 id, variant, major, minor;
} __attribute__ ((packed));

static inline fwtype_t determine_firmware_type(struct
 comp_id *nic_id)
{
 if (nic_id->id < 0x8000)
  return FIRMWARE_TYPE_AGERE;
 else if (nic_id->id == 0x8000 && nic_id->major == 0)
```

```
 return FIRMWARE_TYPE_SYMBOL;
 else
 return FIRMWARE_TYPE_INTERSIL;
}

/* Set priv->firmware type, determine firmware properties */
static int determine_firmware(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err;
 struct comp_id nic_id, sta_id;
 unsigned int firmver;
 char tmp[SYMBOL_MAX_VER_LEN+1];

 /* Get the hardware version */
 err = HERMES_READ_RECORD(hw, USER_BAP, HERMES_RID_NICID, &nic_id);
 if (err) {
  printk(KERN_ERR "%s: Cannot read hardware identity: error %d\n",
         dev->name, err);
  return err;
 }

 le16_to_cpus(&nic_id.id);
 le16_to_cpus(&nic_id.variant);
 le16_to_cpus(&nic_id.major);
 le16_to_cpus(&nic_id.minor);
 printk(KERN_DEBUG "%s: Hardware identity %04x:%04x:%04x:%04x\n",
        dev->name, nic_id.id, nic_id.variant,
        nic_id.major,
 nic_id.minor);

 priv->firmware_type = determine_firmware_type(&nic_id);

 /* Get the firmware version */
 err = HERMES_READ_RECORD(hw, USER_BAP, HERMES_RID_STAID, &sta_id);
 if (err) {
  printk(KERN_ERR "%s: Cannot read station identity: error %d\n",
         dev->name, err);
  return err;
 }

 le16_to_cpus(&sta_id.id);
 le16_to_cpus(&sta_id.variant);
 le16_to_cpus(&sta_id.major);
 le16_to_cpus(&sta_id.minor);
 printk(KERN_DEBUG "%s: Station identity  %04x:%04x:%04x:%04x\n",
        dev->name, sta_id.id, sta_id.variant,
```

```
      sta_id.major, sta_id.minor);

switch (sta_id.id) {
case 0x15:
 printk(KERN_ERR "%s: Primary firmware is active\n",
       dev->name);
 return -ENODEV;
case 0x14b:
 printk(KERN_ERR "%s: Tertiary firmware is active\n",
       dev->name);
 return -ENODEV;
case 0x1f: /* Intersil, Agere, Symbol Spectrum24 */
case 0x21: /* Symbol Spectrum24 Trilogy */
 break;
default:
 printk(KERN_NOTICE "%s: Unknown station ID, please report\n",
       dev->name);
 break;
}

/*
Default capabilities */
priv->has_sensitivity = 1;
priv->has_mwo = 0;
priv->has_preamble = 0;
priv->has_port3 = 1;
priv->has_ibss = 1;
priv->has_wep = 0;
priv->has_big_wep = 0;

/* Determine capabilities from the firmware version */
switch (priv->firmware_type) {
case FIRMWARE_TYPE_AGERE:
 /* Lucent Wavelan IEEE, Lucent Orinoco, Cabletron RoamAbout,
    ELSA, Melco, HP, IBM, Dell 1150, Compaq 110/210 */
 snprintf(priv->fw_name, sizeof(priv->fw_name) - 1,
   "Lucent/Agere %d.%02d", sta_id.major, sta_id.minor);

 firmver = ((unsigned long)sta_id.major << 16) | sta_id.minor;

 priv->has_ibss = (firmver >= 0x60006);
 priv->has_wep = (firmver >= 0x40020);
 priv->has_big_wep = 1; /* FIXME: this is wrong - how do we tell
     Gold cards from the others? */
 priv->has_mwo = (firmver >= 0x60000);
 priv->has_pm = (firmver >= 0x40020); /* Don't work in 7.52 ? */
 priv->ibss_port = 1;
 priv->has_hostscan = (firmver >= 0x8000a);
```

```
 priv->broken_monitor = (firmver
>= 0x80000);

 /* Tested with Agere firmware :
  * 1.16 ; 4.08 ; 4.52 ; 6.04 ; 6.16 ; 7.28 => Jean II
  * Tested CableTron firmware : 4.32 => Anton */
 break;
case FIRMWARE_TYPE_SYMBOL:
 /* Symbol , 3Com AirConnect, Intel, Ericsson WLAN */
 /* Intel MAC : 00:02:B3:* */
 /* 3Com MAC : 00:50:DA:* */
 memset(tmp, 0, sizeof(tmp));
 /* Get the Symbol firmware version */
 err = hermes_read_ltv(hw, USER_BAP,
       HERMES_RID_SECONDARYVERSION_SYMBOL,
       SYMBOL_MAX_VER_LEN, NULL, &tmp);
 if (err) {
  printk(KERN_WARNING
       "%s: Error %d reading Symbol firmware info. Wildly guessing capabilities...\n",
       dev->name, err);
  firmver = 0;
  tmp[0] = '\0';
 } else {
  /* The firmware revision is a string, the format is
   * something like : "V2.20-01".
   * Quick and dirty parsing... - Jean II
   */
  firmver = ((tmp[1] - '0') << 16) | ((tmp[3] - '0') << 12)
   | ((tmp[4] - '0') << 8) | ((tmp[6] - '0') << 4)
   | (tmp[7] - '0');

  tmp[SYMBOL_MAX_VER_LEN]
= '\0';
 }

 snprintf(priv->fw_name, sizeof(priv->fw_name) - 1,
   "Symbol %s", tmp);

 priv->has_ibss = (firmver >= 0x20000);
 priv->has_wep = (firmver >= 0x15012);
 priv->has_big_wep = (firmver >= 0x20000);
 priv->has_pm = (firmver >= 0x20000 && firmver < 0x22000) ||
       (firmver >= 0x29000 && firmver < 0x30000) ||
       firmver >= 0x31000;
 priv->has_preamble = (firmver >= 0x20000);
 priv->ibss_port = 4;
 priv->broken_disableport = (firmver == 0x25013) ||
     (firmver >= 0x30000 && firmver <= 0x31000);
```

```
 priv->has_hostscan = (firmver >= 0x31001) ||
       (firmver >= 0x29057 && firmver < 0x30000);
 /* Tested with Intel firmware : 0x20015 => Jean II */
 /* Tested with 3Com firmware : 0x15012 & 0x22001 => Jean II */
 break;
case FIRMWARE_TYPE_INTERSIL:
 /* D-Link, Linksys, Adtron, ZoomAir, and many others...
  * Samsung, Compaq 100/200 and Proxim are slightly
  * different and less well tested */
 /* D-Link MAC : 00:40:05:* */
 /* Addtron
MAC : 00:90:D1:* */
 snprintf(priv->fw_name, sizeof(priv->fw_name) - 1,
   "Intersil %d.%d.%d", sta_id.major, sta_id.minor,
   sta_id.variant);

 firmver = ((unsigned long)sta_id.major << 16) |
  ((unsigned long)sta_id.minor << 8) | sta_id.variant;

 priv->has_ibss = (firmver >= 0x000700); /* FIXME */
 priv->has_big_wep = priv->has_wep = (firmver >= 0x000800);
 priv->has_pm = (firmver >= 0x000700);
 priv->has_hostscan = (firmver >= 0x010301);

 if (firmver >= 0x000800)
  priv->ibss_port = 0;
 else {
  printk(KERN_NOTICE "%s: Intersil firmware earlier "
        "than v0.8.x - several features not supported\n",
        dev->name);
  priv->ibss_port = 1;
 }
 break;
}
printk(KERN_DEBUG "%s: Firmware determined as %s\n", dev->name,
      priv->fw_name);

return 0;
}

static int orinoco_init(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err = 0;
 struct hermes_idstring nickbuf;
 u16 reclen;
 int len;
```

```
/*
No need to lock, the hw_unavailable flag is already set in
 * alloc_orinocodev() */
priv->nicbuf_size = IEEE80211_FRAME_LEN + ETH_HLEN;

/* Initialize the firmware */
err = hermes_init(hw);
if (err != 0) {
 printk(KERN_ERR "%s: failed to initialize firmware (err = %d)\n",
        dev->name, err);
 goto out;
}

err = determine_firmware(dev);
if (err != 0) {
 printk(KERN_ERR "%s: Incompatible firmware, aborting\n",
        dev->name);
 goto out;
}

if (priv->has_port3)
 printk(KERN_DEBUG "%s: Ad-hoc demo mode supported\n", dev->name);
if (priv->has_ibss)
 printk(KERN_DEBUG "%s: IEEE standard IBSS ad-hoc mode supported\n",
        dev->name);
if (priv->has_wep) {
 printk(KERN_DEBUG "%s: WEP supported, ", dev->name);
 if (priv->has_big_wep)
  printk("104-bit key\n");
 else
  printk("40-bit key\n");
}

/* Get the MAC address */
err = hermes_read_ltv(hw, USER_BAP, HERMES_RID_CNFOWNMACADDR,
        ETH_ALEN, NULL, dev->dev_addr);
if (err) {
 printk(KERN_WARNING
"%s: failed to read MAC address!\n",
        dev->name);
 goto out;
}

printk(KERN_DEBUG "%s: MAC address %02X:%02X:%02X:%02X:%02X:%02X\n",
       dev->name, dev->dev_addr[0], dev->dev_addr[1],
       dev->dev_addr[2], dev->dev_addr[3], dev->dev_addr[4],
       dev->dev_addr[5]);
```

```
/* Get the station name */
err = hermes_read_ltv(hw, USER_BAP, HERMES_RID_CNFOWNNAME,
       sizeof(nickbuf), &reclen, &nickbuf);
if (err) {
 printk(KERN_ERR "%s: failed to read station name\n",
       dev->name);
 goto out;
}
if (nickbuf.len)
 len = min(IW_ESSID_MAX_SIZE, (int)le16_to_cpu(nickbuf.len));
else
 len = min(IW_ESSID_MAX_SIZE, 2 * reclen);
memcpy(priv->nick, &nickbuf.val, len);
priv->nick[len] = '\0';

printk(KERN_DEBUG "%s: Station name \"%s\"\n", dev->name, priv->nick);

err = orinoco_allocate_fid(dev);
if (err) {
 printk(KERN_ERR "%s: failed to allocate NIC buffer!\n",
       dev->name);
 goto out;
}

/* Get allowed channels */
err = hermes_read_wordrec(hw,
USER_BAP, HERMES_RID_CHANNELLIST,
    &priv->channel_mask);
if (err) {
 printk(KERN_ERR "%s: failed to read channel list!\n",
       dev->name);
 goto out;
}

/* Get initial AP density */
err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CNFSYSTEMSCALE,
    &priv->ap_density);
if (err || priv->ap_density < 1 || priv->ap_density > 3) {
 priv->has_sensitivity = 0;
}

/* Get initial RTS threshold */
err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CNFRTSTHRESHOLD,
    &priv->rts_thresh);
if (err) {
 printk(KERN_ERR "%s: failed to read RTS threshold!\n",
       dev->name);
```

```
 goto out;
}

/* Get initial fragmentation settings */
if (priv->has_mwo)
 err = hermes_read_wordrec(hw, USER_BAP,
     HERMES_RID_CNFMWOROBUST_AGERE,
     &priv->mwo_robust);
else
 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CNFFRAGMENTATIONTHRESHOLD,
     &priv->frag_thresh);
if (err) {
 printk(KERN_ERR "%s: failed to read fragmentation settings!\n",
       dev->name);
 goto
out;
}

/* Power management setup */
if (priv->has_pm) {
 priv->pm_on = 0;
 priv->pm_mcast = 1;
 err = hermes_read_wordrec(hw, USER_BAP,
     HERMES_RID_CNFMAXSLEEPDURATION,
     &priv->pm_period);
 if (err) {
  printk(KERN_ERR "%s: failed to read power management period!\n",
        dev->name);
  goto out;
 }
 err = hermes_read_wordrec(hw, USER_BAP,
     HERMES_RID_CNFPMHOLDOVERDURATION,
     &priv->pm_timeout);
 if (err) {
  printk(KERN_ERR "%s: failed to read power management timeout!\n",
        dev->name);
  goto out;
 }
}

/* Preamble setup */
if (priv->has_preamble) {
 err = hermes_read_wordrec(hw, USER_BAP,
     HERMES_RID_CNFPREAMBLE_SYMBOL,
     &priv->preamble);
 if (err)
  goto out;
}
```

```
 /* Set up the default configuration */
 priv->iw_mode = IW_MODE_INFRA;
 /* By default use IEEE/IBSS ad-hoc mode if we have it */
 priv->prefer_port3 = priv->has_port3 && (! priv->has_ibss);
 set_port_type(priv);
 priv->channel = 0; /* use firmware
 default */

 priv->promiscuous = 0;
 priv->wep_on = 0;
 priv->tx_key = 0;

 /* Make the hardware available, as long as it hasn't been
  * removed elsewhere (e.g. by PCMCIA hot unplug) */
 spin_lock_irq(&priv->lock);
 priv->hw_unavailable--;
 spin_unlock_irq(&priv->lock);

 printk(KERN_DEBUG "%s: ready\n", dev->name);

out:
 return err;
}

struct net_device *alloc_orinocodev(int sizeof_card,
       int (*hard_reset)(struct orinoco_private *))
{
 struct net_device *dev;
 struct orinoco_private *priv;

 dev = alloc_etherdev(sizeof(struct orinoco_private) + sizeof_card);
 if (! dev)
  return NULL;
 priv = netdev_priv(dev);
 priv->ndev = dev;
 if (sizeof_card)
  priv->card = (void *)((unsigned long)priv
        + sizeof(struct orinoco_private));
 else
  priv->card = NULL;

 /* Setup / override net_device fields */
 dev->init = orinoco_init;
 dev->hard_start_xmit = orinoco_xmit;
 dev->tx_timeout = orinoco_tx_timeout;
 dev->watchdog_timeo = HZ; /* 1 second timeout */
 dev->get_stats =
```

```
 orinoco_get_stats;
 dev->ethtool_ops = &orinoco_ethtool_ops;
 dev->wireless_handlers = (struct iw_handler_def *)&orinoco_handler_def;
#ifdef WIRELESS_SPY
 priv->wireless_data.spy_data = &priv->spy_data;
 dev->wireless_data = &priv->wireless_data;
#endif
 dev->change_mtu = orinoco_change_mtu;
 dev->set_multicast_list = orinoco_set_multicast_list;
 /* we use the default eth_mac_addr for setting the MAC addr */

 /* Set up default callbacks */
 dev->open = orinoco_open;
 dev->stop = orinoco_stop;
 priv->hard_reset = hard_reset;

 spin_lock_init(&priv->lock);
 priv->open = 0;
 priv->hw_unavailable = 1; /* orinoco_init() must clear this
      * before anything else touches the
      * hardware */
 INIT_WORK(&priv->reset_work, (void (*)(void *))orinoco_reset, dev);
 INIT_WORK(&priv->join_work, (void (*)(void *))orinoco_join_ap, dev);
 INIT_WORK(&priv->wevent_work, (void (*)(void *))orinoco_send_wevents, dev);

 netif_carrier_off(dev);
 priv->last_linkstatus = 0xffff;

 return dev;

}

void
 free_orinocodev(struct net_device *dev)
{
 struct orinoco_private *priv = netdev_priv(dev);

 kfree(priv->scan_result);
 free_netdev(dev);
}

/********************************************************************/
/* Wireless extensions                                              */
/********************************************************************/

/* Return : < 0 -> error code ; >= 0 -> length */
static int orinoco_hw_get_essid(struct orinoco_private *priv, int *active,
    char buf[IW_ESSID_MAX_SIZE+1])
```

```
{
 hermes_t *hw = &priv->hw;
 int err = 0;
 struct hermes_idstring essidbuf;
 char *p = (char *)(&essidbuf.val);
 int len;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 if (strlen(priv->desired_essid) > 0) {
  /* We read the desired SSID from the hardware rather
     than from priv->desired_essid, just in case the
     firmware is allowed to change it on us. I'm not
     sure about this */
  /* My guess is that the OWNSSID should always be whatever

 * we set to the card, whereas CURRENT_SSID is the one that
   * may change... - Jean II */
  u16 rid;

  *active = 1;

  rid = (priv->port_type == 3) ? HERMES_RID_CNFOWNSSID :
   HERMES_RID_CNFDESIREDSSID;

  err = hermes_read_ltv(hw, USER_BAP, rid, sizeof(essidbuf),
         NULL, &essidbuf);
  if (err)
   goto fail_unlock;
 } else {
  *active = 0;

  err = hermes_read_ltv(hw, USER_BAP, HERMES_RID_CURRENTSSID,
         sizeof(essidbuf), NULL, &essidbuf);
  if (err)
   goto fail_unlock;
 }

 len = le16_to_cpu(essidbuf.len);
 BUG_ON(len > IW_ESSID_MAX_SIZE);

 memset(buf, 0, IW_ESSID_MAX_SIZE);
 memcpy(buf, p, len);
 err = len;

fail_unlock:
```

```
 orinoco_unlock(priv, &flags);

 return err;
}

static long orinoco_hw_get_freq(struct orinoco_private *priv)
{

 hermes_t *hw = &priv->hw;
 int err = 0;
 u16 channel;
 long freq = 0;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CURRENTCHANNEL,
 &channel);
 if (err)
  goto out;

 /* Intersil firmware 1.3.5 returns 0 when the interface is down */
 if (channel == 0) {
  err = -EBUSY;
  goto out;
 }

 if ( (channel < 1) || (channel > NUM_CHANNELS) ) {
  printk(KERN_WARNING "%s: Channel out of range (%d)!\n",
         priv->ndev->name, channel);
  err = -EBUSY;
  goto out;

 }
 freq = channel_frequency[channel-1] * 100000;

out:
 orinoco_unlock(priv, &flags);

 if (err > 0)
  err = -EBUSY;
 return err ? err : freq;
}

static int orinoco_hw_get_bitratelist(struct orinoco_private *priv,
          int *numrates, s32 *rates, int max)
{
```

```
 hermes_t *hw = &priv->hw;
 struct hermes_idstring list;
 unsigned char *p = (unsigned char *)&list.val;
 int err = 0;
 int num;
 int i;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 err = hermes_read_ltv(hw, USER_BAP, HERMES_RID_SUPPORTEDDATARATES,
        sizeof(list), NULL, &list);
 orinoco_unlock(priv, &flags);

 if (err)
  return err;

 num = le16_to_cpu(list.len);
 *numrates
 = num;
 num = min(num, max);

 for (i = 0; i < num; i++) {
  rates[i] = (p[i] & 0x7f) * 500000; /* convert to bps */
 }

 return 0;
}

static int orinoco_ioctl_getname(struct net_device *dev,
     struct iw_request_info *info,
     char *name,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int numrates;
 int err;

 err = orinoco_hw_get_bitratelist(priv, &numrates, NULL, 0);

 if (!err && (numrates > 2))
  strcpy(name, "IEEE 802.11b");
 else
  strcpy(name, "IEEE 802.11-DS");

 return 0;
}
```

```
static int orinoco_ioctl_setwap(struct net_device *dev,
    struct iw_request_info *info,
    struct sockaddr *ap_addr,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int err = -EINPROGRESS;  /* Call commit handler */
 unsigned long flags;
 static const u8 off_addr[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00 };
 static const u8 any_addr[] = { 0xff, 0xff, 0xff, 0xff, 0xff, 0xff };

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 /* Enable
 automatic roaming - no sanity checks are needed */
 if (memcmp(&ap_addr->sa_data, off_addr, ETH_ALEN) == 0 ||
     memcmp(&ap_addr->sa_data, any_addr, ETH_ALEN) == 0) {
  priv->bssid_fixed = 0;
  memset(priv->desired_bssid, 0, ETH_ALEN);

  /* "off" means keep existing connection */
  if (ap_addr->sa_data[0] == 0) {
   __orinoco_hw_set_wap(priv);
   err = 0;
  }
  goto out;
 }

 if (priv->firmware_type == FIRMWARE_TYPE_AGERE) {
  printk(KERN_WARNING "%s: Lucent/Agere firmware doesn't "
         "support manual roaming\n",
         dev->name);
  err = -EOPNOTSUPP;
  goto out;
 }

 if (priv->iw_mode != IW_MODE_INFRA) {
  printk(KERN_WARNING "%s: Manual roaming supported only in "
         "managed mode\n", dev->name);
  err = -EOPNOTSUPP;
  goto out;
 }

 /* Intersil firmware hangs without Desired ESSID */
 if (priv->firmware_type == FIRMWARE_TYPE_INTERSIL &&
     strlen(priv->desired_essid) == 0) {
```

```
 printk(KERN_WARNING "%s: Desired ESSID must be set for "
        "manual roaming\n",
 dev->name);
 err = -EOPNOTSUPP;
 goto out;
 }

 /* Finally, enable manual roaming */
 priv->bssid_fixed = 1;
 memcpy(priv->desired_bssid, &ap_addr->sa_data, ETH_ALEN);

out:
 orinoco_unlock(priv, &flags);
 return err;
}

static int orinoco_ioctl_getwap(struct net_device *dev,
    struct iw_request_info *info,
    struct sockaddr *ap_addr,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);

 hermes_t *hw = &priv->hw;
 int err = 0;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 ap_addr->sa_family = ARPHRD_ETHER;
 err = hermes_read_ltv(hw, USER_BAP, HERMES_RID_CURRENTBSSID,
        ETH_ALEN, NULL, ap_addr->sa_data);

 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_setmode(struct net_device *dev,
     struct iw_request_info *info,
     u32 *mode,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int err = -EINPROGRESS;  /* Call commit handler */
 unsigned long flags;
```

```
 if (priv->iw_mode
 == *mode)
  return 0;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 switch (*mode) {
 case IW_MODE_ADHOC:
  if (!priv->has_ibss && !priv->has_port3)
   err = -EOPNOTSUPP;
  break;

 case IW_MODE_INFRA:
  break;

 case IW_MODE_MONITOR:
  if (priv->broken_monitor && !force_monitor) {
   printk(KERN_WARNING "%s: Monitor mode support is "
          "buggy in this firmware, not enabling\n",
          dev->name);
   err = -EOPNOTSUPP;
  }
  break;

 default:
  err = -EOPNOTSUPP;
  break;
 }

 if (err == -EINPROGRESS) {
  priv->iw_mode = *mode;
  set_port_type(priv);
 }

 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_getmode(struct net_device *dev,
     struct iw_request_info *info,
     u32 *mode,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);

 *mode = priv->iw_mode;
```

```
 return 0;
}

static int orinoco_ioctl_getiwrange(struct net_device *dev,
       struct iw_request_info *info,
       struct iw_point *rrq,
       char *extra)
{
 struct
 orinoco_private *priv = netdev_priv(dev);
 int err = 0;
 struct iw_range *range = (struct iw_range *) extra;
 int numrates;
 int i, k;

 rrq->length = sizeof(struct iw_range);
 memset(range, 0, sizeof(struct iw_range));

 range->we_version_compiled = WIRELESS_EXT;
 range->we_version_source = 14;

 /* Set available channels/frequencies */
 range->num_channels = NUM_CHANNELS;
 k = 0;
 for (i = 0; i < NUM_CHANNELS; i++) {
  if (priv->channel_mask & (1 << i)) {
   range->freq[k].i = i + 1;
   range->freq[k].m = channel_frequency[i] * 100000;
   range->freq[k].e = 1;
   k++;
  }

  if (k >= IW_MAX_FREQUENCIES)
   break;
 }
 range->num_frequency = k;
 range->sensitivity = 3;

 if (priv->has_wep) {
  range->max_encoding_tokens = ORINOCO_MAX_KEYS;
  range->encoding_size[0] = SMALL_KEY_SIZE;
  range->num_encoding_sizes = 1;

  if (priv->has_big_wep) {
   range->encoding_size[1] = LARGE_KEY_SIZE;
   range->num_encoding_sizes = 2;
  }
 }
```

```
if ((priv->iw_mode == IW_MODE_ADHOC) && (!SPY_NUMBER(priv))){
 /*
Quality stats meaningless in ad-hoc mode */
} else {
 range->max_qual.qual = 0x8b - 0x2f;
 range->max_qual.level = 0x2f - 0x95 - 1;
 range->max_qual.noise = 0x2f - 0x95 - 1;
 /* Need to get better values */
 range->avg_qual.qual = 0x24;
 range->avg_qual.level = 0xC2;
 range->avg_qual.noise = 0x9E;
}

err = orinoco_hw_get_bitratelist(priv, &numrates,
    range->bitrate, IW_MAX_BITRATES);
if (err)
 return err;
range->num_bitrates = numrates;

/* Set an indication of the max TCP throughput in bit/s that we can
 * expect using this interface. May be use for QoS stuff...
 * Jean II */
if (numrates > 2)
 range->throughput = 5 * 1000 * 1000; /* ~5 Mb/s */
else
 range->throughput = 1.5 * 1000 * 1000; /* ~1.5 Mb/s */

range->min_rts = 0;
range->max_rts = 2347;
range->min_frag = 256;
range->max_frag = 2346;

range->min_pmp = 0;
range->max_pmp = 65535000;
range->min_pmt = 0;
range->max_pmt = 65535 * 1000; /* ??? */
range->pmp_flags = IW_POWER_PERIOD;
range->pmt_flags
= IW_POWER_TIMEOUT;
range->pm_capa = IW_POWER_PERIOD | IW_POWER_TIMEOUT | IW_POWER_UNICAST_R;

range->retry_capa = IW_RETRY_LIMIT | IW_RETRY_LIFETIME;
range->retry_flags = IW_RETRY_LIMIT;
range->r_time_flags = IW_RETRY_LIFETIME;
range->min_retry = 0;
range->max_retry = 65535; /* ??? */
range->min_r_time = 0;
```

```
 range->max_r_time = 65535 * 1000; /* ??? */

 /* Event capability (kernel) */
 IW_EVENT_CAPA_SET_KERNEL(range->event_capa);
 /* Event capability (driver) */
 IW_EVENT_CAPA_SET(range->event_capa, SIOCGIWTHRSPY);
 IW_EVENT_CAPA_SET(range->event_capa, SIOCGIWAP);
 IW_EVENT_CAPA_SET(range->event_capa, SIOCGIWSCAN);
 IW_EVENT_CAPA_SET(range->event_capa, IWEVTXDROP);

 return 0;
}

static int orinoco_ioctl_setiwencode(struct net_device *dev,
        struct iw_request_info *info,
        struct iw_point *erq,
        char *keybuf)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int index = (erq->flags & IW_ENCODE_INDEX) - 1;
 int setindex = priv->tx_key;
 int
 enable = priv->wep_on;
 int restricted = priv->wep_restrict;
 u16 xlen = 0;
 int err = -EINPROGRESS;  /* Call commit handler */
 unsigned long flags;

 if (! priv->has_wep)
  return -EOPNOTSUPP;

 if (erq->pointer) {
  /* We actually have a key to set - check its length */
  if (erq->length > LARGE_KEY_SIZE)
   return -E2BIG;

  if ( (erq->length > SMALL_KEY_SIZE) && !priv->has_big_wep )
   return -E2BIG;
 }

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 if (erq->length > 0) {
  if ((index < 0) || (index >= ORINOCO_MAX_KEYS))
   index = priv->tx_key;

  /* Adjust key length to a supported value */
```

```
 if (erq->length > SMALL_KEY_SIZE) {
  xlen = LARGE_KEY_SIZE;
 } else if (erq->length > 0) {
  xlen = SMALL_KEY_SIZE;
 } else
  xlen = 0;

 /* Switch on WEP if off */
 if ((!enable) && (xlen > 0)) {
  setindex = index;
  enable = 1;
 }
} else {
 /* Important note : if the user do "iwconfig eth0 enc off",
  * we will arrive there with an index of -1. This is
valid
  * but need to be taken care off... Jean II */
 if ((index < 0) || (index >= ORINOCO_MAX_KEYS)) {
  if((index != -1) || (erq->flags == 0)) {
   err = -EINVAL;
   goto out;
  }
 } else {
  /* Set the index : Check that the key is valid */
  if(priv->keys[index].len == 0) {
   err = -EINVAL;
   goto out;
  }
  setindex = index;
 }
}

if (erq->flags & IW_ENCODE_DISABLED)
 enable = 0;
if (erq->flags & IW_ENCODE_OPEN)
 restricted = 0;
if (erq->flags & IW_ENCODE_RESTRICTED)
 restricted = 1;

if (erq->pointer && erq->length > 0) {
 priv->keys[index].len = cpu_to_le16(xlen);
 memset(priv->keys[index].data, 0,
        sizeof(priv->keys[index].data));
 memcpy(priv->keys[index].data, keybuf, erq->length);
}
priv->tx_key = setindex;

/* Try fast key change if connected and only keys are changed */
```

```
 if (priv->wep_on && enable && (priv->wep_restrict == restricted) &&
     netif_carrier_ok(dev)) {
  err = __orinoco_hw_setup_wepkeys(priv);
  /* No need to commit if
 successful */
  goto out;
 }

 priv->wep_on = enable;
 priv->wep_restrict = restricted;

out:
 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_getiwencode(struct net_device *dev,
         struct iw_request_info *info,
         struct iw_point *erq,
         char *keybuf)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int index = (erq->flags & IW_ENCODE_INDEX) - 1;
 u16 xlen = 0;
 unsigned long flags;

 if (! priv->has_wep)
  return -EOPNOTSUPP;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 if ((index < 0) || (index >= ORINOCO_MAX_KEYS))
  index = priv->tx_key;

 erq->flags = 0;
 if (! priv->wep_on)
  erq->flags |= IW_ENCODE_DISABLED;
 erq->flags |= index + 1;

 if (priv->wep_restrict)
  erq->flags |= IW_ENCODE_RESTRICTED;
 else
  erq->flags |= IW_ENCODE_OPEN;

 xlen = le16_to_cpu(priv->keys[index].len);
```

```
 erq->length = xlen;

 memcpy(keybuf, priv->keys[index].data, ORINOCO_MAX_KEY_SIZE);

 orinoco_unlock(priv, &flags);
 return 0;
}

static int
 orinoco_ioctl_setessid(struct net_device *dev,
      struct iw_request_info *info,
      struct iw_point *erq,
      char *essidbuf)
{
 struct orinoco_private *priv = netdev_priv(dev);
 unsigned long flags;

 /* Note : ESSID is ignored in Ad-Hoc demo mode, but we can set it
  * anyway... - Jean II */

 /* Hum... Should not use Wireless Extension constant (may change),
  * should use our own... - Jean II */
 if (erq->length > IW_ESSID_MAX_SIZE)
  return -E2BIG;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 /* NULL the string (for NULL termination & ESSID = ANY) - Jean II */
 memset(priv->desired_essid, 0, sizeof(priv->desired_essid));

 /* If not ANY, get the new ESSID */
 if (erq->flags) {
  memcpy(priv->desired_essid, essidbuf, erq->length);
 }

 orinoco_unlock(priv, &flags);

 return -EINPROGRESS;  /* Call commit handler */
}

static int orinoco_ioctl_getessid(struct net_device *dev,
      struct iw_request_info *info,
      struct iw_point *erq,
      char *essidbuf)
{
 struct
```

```
 orinoco_private *priv = netdev_priv(dev);
 int active;
 int err = 0;
 unsigned long flags;

 if (netif_running(dev)) {
  err = orinoco_hw_get_essid(priv, &active, essidbuf);
  if (err < 0)
   return err;
  erq->length = err;
 } else {
  if (orinoco_lock(priv, &flags) != 0)
   return -EBUSY;
  memcpy(essidbuf, priv->desired_essid, IW_ESSID_MAX_SIZE);
  erq->length = strlen(priv->desired_essid);
  orinoco_unlock(priv, &flags);
 }

 erq->flags = 1;

 return 0;
}

static int orinoco_ioctl_setnick(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_point *nrq,
     char *nickbuf)
{
 struct orinoco_private *priv = netdev_priv(dev);
 unsigned long flags;

 if (nrq->length > IW_ESSID_MAX_SIZE)
  return -E2BIG;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 memset(priv->nick, 0, sizeof(priv->nick));
 memcpy(priv->nick, nickbuf, nrq->length);

 orinoco_unlock(priv, &flags);

 return -EINPROGRESS;  /* Call commit handler */
}

static int orinoco_ioctl_getnick(struct
 net_device *dev,
     struct iw_request_info *info,
```

```
    struct iw_point *nrq,
    char *nickbuf)
{
 struct orinoco_private *priv = netdev_priv(dev);
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 memcpy(nickbuf, priv->nick, IW_ESSID_MAX_SIZE);
 orinoco_unlock(priv, &flags);

 nrq->length = strlen(priv->nick);

 return 0;
}

static int orinoco_ioctl_setfreq(struct net_device *dev,
    struct iw_request_info *info,
    struct iw_freq *frq,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int chan = -1;
 unsigned long flags;
 int err = -EINPROGRESS;  /* Call commit handler */

 /* In infrastructure mode the AP sets the channel */
 if (priv->iw_mode == IW_MODE_INFRA)
  return -EBUSY;

 if ( (frq->e == 0) && (frq->m <= 1000) ) {
  /* Setting by channel number */
  chan = frq->m;
 } else {
  /* Setting by frequency - search the table */
  int mult = 1;
  int i;

  for (i = 0; i < (6 - frq->e); i++)
   mult *= 10;

  for (i
 = 0; i < NUM_CHANNELS; i++)
   if (frq->m == (channel_frequency[i] * mult))
    chan = i+1;
 }
```

```
 if ( (chan < 1) || (chan > NUM_CHANNELS) ||
      ! (priv->channel_mask & (1 << (chan-1)) ) )
  return -EINVAL;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 priv->channel = chan;
 if (priv->iw_mode == IW_MODE_MONITOR) {
  /* Fast channel change - no commit if successful */
  hermes_t *hw = &priv->hw;
  err = hermes_docmd_wait(hw, HERMES_CMD_TEST |
         HERMES_TEST_SET_CHANNEL,
     chan, NULL);
 }
 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_getfreq(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_freq *frq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int tmp;

 /* Locking done in there */
 tmp = orinoco_hw_get_freq(priv);
 if (tmp < 0) {
  return tmp;
 }

 frq->m = tmp;
 frq->e = 1;

 return 0;
}

static int orinoco_ioctl_getsens(struct net_device *dev,
     struct iw_request_info *info,

 struct iw_param *srq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
```

```
 u16 val;
 int err;
 unsigned long flags;

 if (!priv->has_sensitivity)
  return -EOPNOTSUPP;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;
 err = hermes_read_wordrec(hw, USER_BAP,
      HERMES_RID_CNFSYSTEMSCALE, &val);
 orinoco_unlock(priv, &flags);

 if (err)
  return err;

 srq->value = val;
 srq->fixed = 0; /* auto */

 return 0;
}

static int orinoco_ioctl_setsens(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_param *srq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int val = srq->value;
 unsigned long flags;

 if (!priv->has_sensitivity)
  return -EOPNOTSUPP;

 if ((val < 1) || (val > 3))
  return -EINVAL;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;
 priv->ap_density = val;
 orinoco_unlock(priv, &flags);

 return -EINPROGRESS;  /* Call commit handler */
}

static int orinoco_ioctl_setrts(struct
 net_device *dev,
    struct iw_request_info *info,
```

```
    struct iw_param *rrq,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int val = rrq->value;
 unsigned long flags;

 if (rrq->disabled)
  val = 2347;

 if ( (val < 0) || (val > 2347) )
  return -EINVAL;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 priv->rts_thresh = val;
 orinoco_unlock(priv, &flags);

 return -EINPROGRESS;  /* Call commit handler */
}

static int orinoco_ioctl_getrts(struct net_device *dev,
    struct iw_request_info *info,
    struct iw_param *rrq,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);

 rrq->value = priv->rts_thresh;
 rrq->disabled = (rrq->value == 2347);
 rrq->fixed = 1;

 return 0;
}

static int orinoco_ioctl_setfrag(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_param *frq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int err = -EINPROGRESS;  /* Call commit handler */
 unsigned long flags;

 if
 (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;
```

```
 if (priv->has_mwo) {
  if (frq->disabled)
   priv->mwo_robust = 0;
  else {
   if (frq->fixed)
    printk(KERN_WARNING "%s: Fixed fragmentation is "
           "not supported on this firmware. "
           "Using MWO robust instead.\n", dev->name);
   priv->mwo_robust = 1;
  }
 } else {
  if (frq->disabled)
   priv->frag_thresh = 2346;
  else {
   if ( (frq->value < 256) || (frq->value > 2346) )
    err = -EINVAL;
   else
    priv->frag_thresh = frq->value & ~0x1; /* must be even */
  }
 }

 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_getfrag(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_param *frq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err;
 u16 val;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 if (priv->has_mwo) {
  err = hermes_read_wordrec(hw, USER_BAP,
        HERMES_RID_CNFMWOROBUST_AGERE,

  &val);
  if (err)
   val = 0;
```

```
 frq->value = val ? 2347 : 0;
 frq->disabled = ! val;
 frq->fixed = 0;
 } else {
 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CNFFRAGMENTATIONTHRESHOLD,
      &val);
 if (err)
  val = 0;

 frq->value = val;
 frq->disabled = (val >= 2346);
 frq->fixed = 1;
 }

 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_setrate(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_param *rrq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int ratemode = -1;
 int bitrate; /* 100s of kilobits */
 int i;
 unsigned long flags;

 /* As the user space doesn't know our highest rate, it uses -1
  * to ask us to set the highest rate.  Test it using "iwconfig
  * ethX rate auto" - Jean II */
 if (rrq->value == -1)
  bitrate = 110;
 else {
  if (rrq->value % 100000)
   return -EINVAL;
  bitrate = rrq->value / 100000;
 }

 if ( (bitrate != 10) && (bitrate != 20)
 &&
      (bitrate != 55) && (bitrate != 110) )
  return -EINVAL;

 for (i = 0; i < BITRATE_TABLE_SIZE; i++)
```

```
 if ( (bitrate_table[i].bitrate == bitrate) &&
      (bitrate_table[i].automatic == ! rrq->fixed) ) {
  ratemode = i;
  break;
 }

if (ratemode == -1)
 return -EINVAL;

if (orinoco_lock(priv, &flags) != 0)
 return -EBUSY;
priv->bitratemode = ratemode;
orinoco_unlock(priv, &flags);

return -EINPROGRESS;
}

static int orinoco_ioctl_getrate(struct net_device *dev,
    struct iw_request_info *info,
    struct iw_param *rrq,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err = 0;
 int ratemode;
 int i;
 u16 val;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 ratemode = priv->bitratemode;

 BUG_ON((ratemode < 0) || (ratemode >= BITRATE_TABLE_SIZE));

 rrq->value = bitrate_table[ratemode].bitrate * 100000;
 rrq->fixed = ! bitrate_table[ratemode].automatic;
 rrq->disabled = 0;

 /* If the
 interface is running we try to find more about the
    current mode */
 if (netif_running(dev)) {
  err = hermes_read_wordrec(hw, USER_BAP,
       HERMES_RID_CURRENTTXRATE, &val);
  if (err)
```

```
  goto out;

 switch (priv->firmware_type) {
 case FIRMWARE_TYPE_AGERE: /* Lucent style rate */
  /* Note : in Lucent firmware, the return value of
   * HERMES_RID_CURRENTTXRATE is the bitrate in Mb/s,
   * and therefore is totally different from the
   * encoding of HERMES_RID_CNFTXRATECONTROL.
   * Don't forget that 6Mb/s is really 5.5Mb/s */
  if (val == 6)
   rrq->value = 5500000;
  else
   rrq->value = val * 1000000;
  break;
 case FIRMWARE_TYPE_INTERSIL: /* Intersil style rate */
 case FIRMWARE_TYPE_SYMBOL: /* Symbol style rate */
  for (i = 0; i < BITRATE_TABLE_SIZE; i++)
   if (bitrate_table[i].intersil_txratectrl == val) {
    ratemode = i;
    break;
   }
  if (i >= BITRATE_TABLE_SIZE)
   printk(KERN_INFO "%s: Unable to determine current bitrate (0x%04hx)\n",

       dev->name, val);

  rrq->value = bitrate_table[ratemode].bitrate * 100000;
  break;
 default:
  BUG();
 }
}

out:
 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_setpower(struct net_device *dev,
      struct iw_request_info *info,
      struct iw_param *prq,
      char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int err = -EINPROGRESS;  /* Call commit handler */
 unsigned long flags;
```

```
 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 if (prq->disabled) {
  priv->pm_on = 0;
 } else {
  switch (prq->flags & IW_POWER_MODE) {
  case IW_POWER_UNICAST_R:
   priv->pm_mcast = 0;
   priv->pm_on = 1;
   break;
  case IW_POWER_ALL_R:
   priv->pm_mcast = 1;
   priv->pm_on = 1;
   break;
  case IW_POWER_ON:
   /* No flags : but we may have a value - Jean II */
   break;
  default:
   err = -EINVAL;
   goto out;
  }

  if (prq->flags & IW_POWER_TIMEOUT) {
   priv->pm_on = 1;
   priv->pm_timeout = prq->value / 1000;
  }
  if (prq->flags
 & IW_POWER_PERIOD) {
   priv->pm_on = 1;
   priv->pm_period = prq->value / 1000;
  }
  /* It's valid to not have a value if we are just toggling
   * the flags... Jean II */
  if(!priv->pm_on) {
   err = -EINVAL;
   goto out;
  }
 }

out:
 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_getpower(struct net_device *dev,
      struct iw_request_info *info,
```

```
    struct iw_param *prq,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err = 0;
 u16 enable, period, timeout, mcast;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CNFPMENABLED, &enable);
 if (err)
  goto out;

 err = hermes_read_wordrec(hw, USER_BAP,
     HERMES_RID_CNFMAXSLEEPDURATION, &period);
 if (err)
  goto out;

 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_CNFPMHOLDOVERDURATION, &timeout);
 if (err)
  goto out;

 err = hermes_read_wordrec(hw, USER_BAP,
 HERMES_RID_CNFMULTICASTRECEIVE, &mcast);
 if (err)
  goto out;

 prq->disabled = !enable;
 /* Note : by default, display the period */
 if ((prq->flags & IW_POWER_TYPE) == IW_POWER_TIMEOUT) {
  prq->flags = IW_POWER_TIMEOUT;
  prq->value = timeout * 1000;
 } else {
  prq->flags = IW_POWER_PERIOD;
  prq->value = period * 1000;
 }
 if (mcast)
  prq->flags |= IW_POWER_ALL_R;
 else
  prq->flags |= IW_POWER_UNICAST_R;

out:
 orinoco_unlock(priv, &flags);

 return err;
```

```
}

static int orinoco_ioctl_getretry(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_param *rrq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err = 0;
 u16 short_limit, long_limit, lifetime;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_SHORTRETRYLIMIT,
     &short_limit);
 if (err)
  goto out;

 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_LONGRETRYLIMIT,

  &long_limit);
 if (err)
  goto out;

 err = hermes_read_wordrec(hw, USER_BAP, HERMES_RID_MAXTRANSMITLIFETIME,
     &lifetime);
 if (err)
  goto out;

 rrq->disabled = 0;  /* Can't be disabled */

 /* Note : by default, display the retry number */
 if ((rrq->flags & IW_RETRY_TYPE) == IW_RETRY_LIFETIME) {
  rrq->flags = IW_RETRY_LIFETIME;
  rrq->value = lifetime * 1000; /* ??? */
 } else {
  /* By default, display the min number */
  if ((rrq->flags & IW_RETRY_LONG)) {
   rrq->flags = IW_RETRY_LIMIT | IW_RETRY_LONG;
   rrq->value = long_limit;
  } else {
   rrq->flags = IW_RETRY_LIMIT;
   rrq->value = short_limit;
   if(short_limit != long_limit)
    rrq->flags |= IW_RETRY_SHORT;
```

```
  }
 }

out:
 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_reset(struct net_device *dev,
          struct iw_request_info *info,
          void *wrqu,
          char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);

 if (! capable(CAP_NET_ADMIN))
  return
 -EPERM;

 if (info->cmd == (SIOCIWFIRSTPRIV + 0x1)) {
  printk(KERN_DEBUG "%s: Forcing reset!\n", dev->name);

  /* Firmware reset */
  orinoco_reset(dev);
 } else {
  printk(KERN_DEBUG "%s: Force scheduling reset!\n", dev->name);

  schedule_work(&priv->reset_work);
 }

 return 0;
}

static int orinoco_ioctl_setibssport(struct net_device *dev,
         struct iw_request_info *info,
         void *wrqu,
         char *extra)

{
 struct orinoco_private *priv = netdev_priv(dev);
 int val = *( (int *) extra );
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 priv->ibss_port = val ;
```

```
 /* Actually update the mode we are using */
 set_port_type(priv);

 orinoco_unlock(priv, &flags);
 return -EINPROGRESS;  /* Call commit handler */
}

static int orinoco_ioctl_getibssport(struct net_device *dev,
        struct iw_request_info *info,
        void *wrqu,
        char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int *val = (int *) extra;

 *val
 = priv->ibss_port;
 return 0;
}

static int orinoco_ioctl_setport3(struct net_device *dev,
     struct iw_request_info *info,
     void *wrqu,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int val = *( (int *) extra );
 int err = 0;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 switch (val) {
 case 0: /* Try to do IEEE ad-hoc mode */
  if (! priv->has_ibss) {
   err = -EINVAL;
   break;
  }
  priv->prefer_port3 = 0;

  break;

 case 1: /* Try to do Lucent proprietary ad-hoc mode */
  if (! priv->has_port3) {
   err = -EINVAL;
   break;
```

```
 }
 priv->prefer_port3 = 1;
 break;

 default:
 err = -EINVAL;
 }

 if (! err) {
 /* Actually update the mode we are using */
 set_port_type(priv);
 err = -EINPROGRESS;
 }

 orinoco_unlock(priv, &flags);

 return err;
}

static int orinoco_ioctl_getport3(struct net_device *dev,
     struct iw_request_info *info,
     void *wrqu,
     char *extra)
{
 struct orinoco_private *priv =
 netdev_priv(dev);
 int *val = (int *) extra;

 *val = priv->prefer_port3;
 return 0;
}

static int orinoco_ioctl_setpreamble(struct net_device *dev,
        struct iw_request_info *info,
        void *wrqu,
        char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 unsigned long flags;
 int val;

 if (! priv->has_preamble)
 return -EOPNOTSUPP;

 /* 802.11b has recently defined some short preamble.
  * Basically, the Phy header has been reduced in size.
  * This increase performance, especially at high rates
  * (the preamble is transmitted at 1Mb/s), unfortunately
```

```
 * this give compatibility troubles... - Jean II */
 val = *( (int *) extra );

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 if (val)
  priv->preamble = 1;
 else
  priv->preamble = 0;

 orinoco_unlock(priv, &flags);

 return -EINPROGRESS;  /* Call commit handler */
}

static int orinoco_ioctl_getpreamble(struct net_device *dev,
        struct iw_request_info *info,
        void *wrqu,

   char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int *val = (int *) extra;

 if (! priv->has_preamble)
  return -EOPNOTSUPP;

 *val = priv->preamble;
 return 0;
}

/* ioctl interface to hermes_read_ltv()
* To use with iwpriv, pass the RID as the token argument, e.g.
* iwpriv get_rid [0xfc00]
* At least Wireless Tools 25 is required to use iwpriv.
* For Wireless Tools 25 and 26 append "dummy" are the end. */
static int orinoco_ioctl_getrid(struct net_device *dev,
    struct iw_request_info *info,
    struct iw_point *data,
    char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int rid = data->flags;
 u16 length;
 int err;
 unsigned long flags;
```

```
 /* It's a "get" function, but we don't want users to access the
  * WEP key and other raw firmware data */
 if (! capable(CAP_NET_ADMIN))
  return -EPERM;

 if (rid < 0xfc00 || rid > 0xffff)
  return -EINVAL;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 err = hermes_read_ltv(hw,
 USER_BAP, rid, MAX_RID_LEN, &length,
         extra);
 if (err)
  goto out;

 data->length = min_t(u16, HERMES_RECLEN_TO_BYTES(length),
        MAX_RID_LEN);

out:
 orinoco_unlock(priv, &flags);
 return err;
}

/* Trigger a scan (look for other cells in the vicinity */
static int orinoco_ioctl_setscan(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_param *srq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 hermes_t *hw = &priv->hw;
 int err = 0;
 unsigned long flags;

 /* Note : you may have realised that, as this is a SET operation,
  * this is privileged and therefore a normal user can't
  * perform scanning.
  * This is not an error, while the device perform scanning,
  * traffic doesn't flow, so it's a perfect DoS...
  * Jean II */

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 /* Scanning with port 0 disabled would fail */
```

```
if (!netif_running(dev)) {
 err = -ENETDOWN;
 goto out;
}

/* In monitor mode, the
scan results are always empty.
 * Probe responses are passed to the driver as received
 * frames and could be processed in software. */
if (priv->iw_mode == IW_MODE_MONITOR) {
 err = -EOPNOTSUPP;
 goto out;
}

/* Note : because we don't lock out the irq handler, the way
 * we access scan variables in priv is critical.
 * o scan_inprogress : not touched by irq handler
 * o scan_mode : not touched by irq handler
 * o scan_result : irq is strict producer, non-irq is strict
 *  consumer.
 * o scan_len : synchronised with scan_result
 * Before modifying anything on those variables, please think hard !
 * Jean II */

/* If there is still some left-over scan results, get rid of it */
if (priv->scan_result != NULL) {
 /* What's likely is that a client did crash or was killed
  * between triggering the scan request and reading the
  * results, so we need to reset everything.
  * Some clients that are too slow may suffer from that...
  * Jean II */
 kfree(priv->scan_result);
 priv->scan_result
= NULL;
}

/* Save flags */
priv->scan_mode = srq->flags;

/* Always trigger scanning, even if it's in progress.
 * This way, if the info frame get lost, we will recover somewhat
 * gracefully  - Jean II */

if (priv->has_hostscan) {
 switch (priv->firmware_type) {
 case FIRMWARE_TYPE_SYMBOL:
  err = hermes_write_wordrec(hw, USER_BAP,
        HERMES_RID_CNFHOSTSCAN_SYMBOL,
```

```
        HERMES_HOSTSCAN_SYMBOL_ONCE |
        HERMES_HOSTSCAN_SYMBOL_BCAST);
   break;
  case FIRMWARE_TYPE_INTERSIL: {
   __le16 req[3];

   req[0] = cpu_to_le16(0x3fff); /* All channels */
   req[1] = cpu_to_le16(0x0001); /* rate 1 Mbps */
   req[2] = 0;    /* Any ESSID */
   err = HERMES_WRITE_RECORD(hw, USER_BAP,
       HERMES_RID_CNFHOSTSCAN, &req);
  }
  break;
  case FIRMWARE_TYPE_AGERE:
   err = hermes_write_wordrec(hw, USER_BAP,
        HERMES_RID_CNFSCANSSID_AGERE,
        0); /* Any ESSID */
   if (err)
    break;

   err = hermes_inquire(hw, HERMES_INQ_SCAN);
   break;
  }
 }
 else
  err = hermes_inquire(hw, HERMES_INQ_SCAN);

 /* One more client */
 if (! err)
  priv->scan_inprogress = 1;

out:
 orinoco_unlock(priv, &flags);
 return err;
}

/* Translate scan data returned from the card to a card independant
* format that the Wireless Tools will understand - Jean II
* Return message length or -errno for fatal errors */
static inline int orinoco_translate_scan(struct net_device *dev,
      char *buffer,
      char *scan,
      int scan_len)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int   offset;  /* In the scan data */
 union hermes_scan_info *atom;
 int   atom_len;
```

```
u16   capabilities;
u16   channel;
struct iw_event  iwe;  /* Temporary buffer */
char *    current_ev = buffer;
char *    end_buf = buffer + IW_SCAN_MAX_DATA;

switch (priv->firmware_type) {
case FIRMWARE_TYPE_AGERE:
 atom_len = sizeof(struct agere_scan_apinfo);
 offset = 0;
 break;
case FIRMWARE_TYPE_SYMBOL:
 /* Lack of documentation necessitates this hack.
  *
Different firmwares have 68 or 76 byte long atoms.
  * We try modulo first.  If the length divides by both,
  * we check what would be the channel in the second
  * frame for a 68-byte atom.  76-byte atoms have 0 there.
  * Valid channel cannot be 0.  */
 if (scan_len % 76)
  atom_len = 68;
 else if (scan_len % 68)
  atom_len = 76;
 else if (scan_len >= 1292 && scan[68] == 0)
  atom_len = 76;
 else
  atom_len = 68;
 offset = 0;
 break;
case FIRMWARE_TYPE_INTERSIL:
 offset = 4;
 if (priv->has_hostscan) {
  atom_len = le16_to_cpup((__le16 *)scan);
  /* Sanity check for atom_len */
  if (atom_len < sizeof(struct prism2_scan_apinfo)) {
   printk(KERN_ERR "%s: Invalid atom_len in scan data: %d\n",
   dev->name, atom_len);
   return -EIO;
  }
 } else
  atom_len = offsetof(struct prism2_scan_apinfo, atim);
 break;
default:
 return -EOPNOTSUPP;
}

/* Check that we got an whole number of atoms */
if ((scan_len - offset) % atom_len) {
```

```
 printk(KERN_ERR "%s: Unexpected
scan data length %d, "
       "atom_len %d, offset %d\n", dev->name, scan_len,
       atom_len, offset);
 return -EIO;
}

/* Read the entries one by one */
for (; offset + atom_len <= scan_len; offset += atom_len) {
 /* Get next atom */
 atom = (union hermes_scan_info *) (scan + offset);

 /* First entry *MUST* be the AP MAC address */
 iwe.cmd = SIOCGIWAP;
 iwe.u.ap_addr.sa_family = ARPHRD_ETHER;
 memcpy(iwe.u.ap_addr.sa_data, atom->a.bssid, ETH_ALEN);
 current_ev = iwe_stream_add_event(current_ev, end_buf, &iwe, IW_EV_ADDR_LEN);

 /* Other entries will be displayed in the order we give them */

 /* Add the ESSID */
 iwe.u.data.length = le16_to_cpu(atom->a.essid_len);
 if (iwe.u.data.length > 32)
  iwe.u.data.length = 32;
 iwe.cmd = SIOCGIWESSID;
 iwe.u.data.flags = 1;
 current_ev = iwe_stream_add_point(current_ev, end_buf, &iwe, atom->a.essid);

 /* Add mode */
 iwe.cmd = SIOCGIWMODE;
 capabilities = le16_to_cpu(atom->a.capabilities);
 if (capabilities
& 0x3) {
  if (capabilities & 0x1)
   iwe.u.mode = IW_MODE_MASTER;
  else
   iwe.u.mode = IW_MODE_ADHOC;
  current_ev = iwe_stream_add_event(current_ev, end_buf, &iwe, IW_EV_UINT_LEN);
 }

 channel = atom->s.channel;
 if ( (channel >= 1) && (channel <= NUM_CHANNELS) ) {
  /* Add frequency */
  iwe.cmd = SIOCGIWFREQ;
  iwe.u.freq.m = channel_frequency[channel-1] * 100000;
  iwe.u.freq.e = 1;
  current_ev = iwe_stream_add_event(current_ev, end_buf,
        &iwe, IW_EV_FREQ_LEN);
```

```
 }

 /* Add quality statistics */
 iwe.cmd = IWEVQUAL;
 iwe.u.qual.updated = 0x10; /* no link quality */
 iwe.u.qual.level = (__u8) le16_to_cpu(atom->a.level) - 0x95;
 iwe.u.qual.noise = (__u8) le16_to_cpu(atom->a.noise) - 0x95;
 /* Wireless tools prior to 27.pre22 will show link quality
  * anyway, so we provide a reasonable value. */
 if (iwe.u.qual.level > iwe.u.qual.noise)
  iwe.u.qual.qual = iwe.u.qual.level - iwe.u.qual.noise;
 else
  iwe.u.qual.qual = 0;
 current_ev = iwe_stream_add_event(current_ev,
end_buf, &iwe, IW_EV_QUAL_LEN);

 /* Add encryption capability */
 iwe.cmd = SIOCGIWENCODE;
 if (capabilities & 0x10)
  iwe.u.data.flags = IW_ENCODE_ENABLED | IW_ENCODE_NOKEY;
 else
  iwe.u.data.flags = IW_ENCODE_DISABLED;
 iwe.u.data.length = 0;
 current_ev = iwe_stream_add_point(current_ev, end_buf, &iwe, atom->a.essid);

 /* Bit rate is not available in Lucent/Agere firmwares */
 if (priv->firmware_type != FIRMWARE_TYPE_AGERE) {
  char * current_val = current_ev + IW_EV_LCP_LEN;
  int i;
  int step;

  if (priv->firmware_type == FIRMWARE_TYPE_SYMBOL)
   step = 2;
  else
   step = 1;

  iwe.cmd = SIOCGIWRATE;
  /* Those two flags are ignored... */
  iwe.u.bitrate.fixed = iwe.u.bitrate.disabled = 0;
  /* Max 10 values */
  for (i = 0; i < 10; i += step) {
   /* NULL terminated */
   if (atom->p.rates[i] == 0x0)
    break;
   /* Bit rate given in 500 kb/s units (+ 0x80) */
   iwe.u.bitrate.value = ((atom->p.rates[i] & 0x7f) * 500000);
   current_val
```

```
 = iwe_stream_add_value(current_ev, current_val,
          end_buf, &iwe,
          IW_EV_PARAM_LEN);
   }
   /* Check if we added any event */
   if ((current_val - current_ev) > IW_EV_LCP_LEN)
    current_ev = current_val;
  }

  /* The other data in the scan result are not really
   * interesting, so for now drop it - Jean II */
 }
 return current_ev - buffer;
}

/* Return results of a scan */
static int orinoco_ioctl_getscan(struct net_device *dev,
     struct iw_request_info *info,
     struct iw_point *srq,
     char *extra)
{
 struct orinoco_private *priv = netdev_priv(dev);
 int err = 0;
 unsigned long flags;

 if (orinoco_lock(priv, &flags) != 0)
  return -EBUSY;

 /* If no results yet, ask to try again later */
 if (priv->scan_result == NULL) {
  if (priv->scan_inprogress)
   /* Important note : we don't want to block the caller
    * until results are ready for various reasons.
    * First, managing wait queues is complex and racy.
    * Second, we grab some
 rtnetlink lock before comming
    * here (in dev_ioctl()).
    * Third, we generate an Wireless Event, so the
    * caller can wait itself on that - Jean II */
   err = -EAGAIN;
  else
   /* Client error, no scan results...
    * The caller need to restart the scan. */
   err = -ENODATA;
 } else {
  /* We have some results to push back to user space */

  /* Translate to WE format */
```

```
  int ret = orinoco_translate_scan(dev, extra,
      priv->scan_result,
      priv->scan_len);

  if (ret < 0) {
   err = ret;
   kfree(priv->scan_result);
   priv->scan_result = NULL;
  } else {
   srq->length = ret;

   /* Return flags */
   srq->flags = (__u16) priv->scan_mode;

   /* In any case, Scan results will be cleaned up in the
    * reset function and when exiting the driver.
    * The person triggering the scanning may never come to
    * pick the results, so we need to do it in those places.
    * Jean II */

#ifdef SCAN_SINGLE_READ
   /* If you enable this option, only one client (the first

 * one) will be able to read the result (and only one
    * time). If there is multiple concurent clients that
    * want to read scan results, this behavior is not
    * advisable - Jean II */
   kfree(priv->scan_result);
   priv->scan_result = NULL;
#endif /* SCAN_SINGLE_READ */
   /* Here, if too much time has elapsed since last scan,
    * we may want to clean up scan results... - Jean II */
  }

  /* Scan is no longer in progress */
  priv->scan_inprogress = 0;
 }

 orinoco_unlock(priv, &flags);
 return err;
}

/* Commit handler, called after set operations */
static int orinoco_ioctl_commit(struct net_device *dev,
    struct iw_request_info *info,
    void *wrqu,
    char *extra)
{
```

```
 struct orinoco_private *priv = netdev_priv(dev);
 struct hermes *hw = &priv->hw;
 unsigned long flags;
 int err = 0;

 if (!priv->open)
  return 0;

 if (priv->broken_disableport) {
  orinoco_reset(dev);
  return 0;
 }

 if (orinoco_lock(priv, &flags) != 0)
  return err;

 err = hermes_disable_port(hw,
 0);
 if (err) {
  printk(KERN_WARNING "%s: Unable to disable port "
         "while reconfiguring card\n", dev->name);
  priv->broken_disableport = 1;
  goto out;
 }

 err = __orinoco_program_rids(dev);
 if (err) {
  printk(KERN_WARNING "%s: Unable to reconfigure card\n",
         dev->name);
  goto out;
 }

 err = hermes_enable_port(hw, 0);
 if (err) {
  printk(KERN_WARNING "%s: Unable to enable port while reconfiguring card\n",
         dev->name);
  goto out;
 }

out:
 if (err) {
  printk(KERN_WARNING "%s: Resetting instead...\n", dev->name);
  schedule_work(&priv->reset_work);
  err = 0;
 }

 orinoco_unlock(priv, &flags);
 return err;
```

```
}

static const struct iw_priv_args orinoco_privtab[] = {
 { SIOCIWFIRSTPRIV + 0x0, 0, 0, "force_reset" },
 { SIOCIWFIRSTPRIV + 0x1, 0, 0, "card_reset" },
 { SIOCIWFIRSTPRIV + 0x2, IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1,
   0, "set_port3" },
 { SIOCIWFIRSTPRIV + 0x3, 0, IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1,
   "get_port3" },
 { SIOCIWFIRSTPRIV
 + 0x4, IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1,
   0, "set_preamble" },
 { SIOCIWFIRSTPRIV + 0x5, 0, IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1,
   "get_preamble" },
 { SIOCIWFIRSTPRIV + 0x6, IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1,
   0, "set_ibssport" },
 { SIOCIWFIRSTPRIV + 0x7, 0, IW_PRIV_TYPE_INT | IW_PRIV_SIZE_FIXED | 1,
   "get_ibssport" },
 { SIOCIWFIRSTPRIV + 0x9, 0, IW_PRIV_TYPE_BYTE | MAX_RID_LEN,
   "get_rid" },
};


/*
* Structures to export the Wireless Handlers
*/

static const iw_handler orinoco_handler[] = {
 [SIOCSIWCOMMIT-SIOCIWFIRST] = (iw_handler) orinoco_ioctl_commit,
 [SIOCGIWNAME  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getname,
 [SIOCSIWFREQ  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setfreq,
 [SIOCGIWFREQ  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getfreq,
 [SIOCSIWMODE  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setmode,
 [SIOCGIWMODE  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getmode,
 [SIOCSIWSENS  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setsens,
 [SIOCGIWSENS
  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getsens,
 [SIOCGIWRANGE -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getiwrange,
 [SIOCSIWSPY   -SIOCIWFIRST] = (iw_handler) iw_handler_set_spy,
 [SIOCGIWSPY   -SIOCIWFIRST] = (iw_handler) iw_handler_get_spy,
 [SIOCSIWTHRSPY-SIOCIWFIRST] = (iw_handler) iw_handler_set_thrspy,
 [SIOCGIWTHRSPY-SIOCIWFIRST] = (iw_handler) iw_handler_get_thrspy,
 [SIOCSIWAP    -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setwap,
 [SIOCGIWAP    -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getwap,
 [SIOCSIWSCAN  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setscan,
 [SIOCGIWSCAN  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getscan,
 [SIOCSIWESSID -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setessid,
 [SIOCGIWESSID -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getessid,
```

```
 [SIOCSIWNICKN -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setnick,
 [SIOCGIWNICKN -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getnick,
 [SIOCSIWRATE  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setrate,
 [SIOCGIWRATE
  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getrate,
 [SIOCSIWRTS   -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setrts,
 [SIOCGIWRTS   -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getrts,
 [SIOCSIWFRAG  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setfrag,
 [SIOCGIWFRAG  -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getfrag,
 [SIOCGIWRETRY -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getretry,
 [SIOCSIWENCODE-SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setiwencode,
 [SIOCGIWENCODE-SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getiwencode,
 [SIOCSIWPOWER -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_setpower,
 [SIOCGIWPOWER -SIOCIWFIRST] = (iw_handler) orinoco_ioctl_getpower,
};


/*
 Added typecasting since we no longer use iwreq_data -- Moustafa
*/
static const iw_handler orinoco_private_handler[] = {
 [0] = (iw_handler) orinoco_ioctl_reset,
 [1] = (iw_handler) orinoco_ioctl_reset,
 [2] = (iw_handler) orinoco_ioctl_setport3,
 [3] = (iw_handler) orinoco_ioctl_getport3,
 [4]
 = (iw_handler) orinoco_ioctl_setpreamble,
 [5] = (iw_handler) orinoco_ioctl_getpreamble,
 [6] = (iw_handler) orinoco_ioctl_setibssport,
 [7] = (iw_handler) orinoco_ioctl_getibssport,
 [9] = (iw_handler) orinoco_ioctl_getrid,
};

static const struct iw_handler_def orinoco_handler_def = {
 .num_standard = ARRAY_SIZE(orinoco_handler),
 .num_private = ARRAY_SIZE(orinoco_private_handler),
 .num_private_args = ARRAY_SIZE(orinoco_privtab),
 .standard = orinoco_handler,
 .private = orinoco_private_handler,
 .private_args = orinoco_privtab,
 .get_wireless_stats = orinoco_get_wireless_stats,
};

static void orinoco_get_drvinfo(struct net_device *dev,
    struct ethtool_drvinfo *info)
{
 struct orinoco_private *priv = netdev_priv(dev);
```

```
 strncpy(info->driver, DRIVER_NAME, sizeof(info->driver) - 1);
 strncpy(info->version, DRIVER_VERSION, sizeof(info->version) - 1);
 strncpy(info->fw_version, priv->fw_name, sizeof(info->fw_version) - 1);
 if (dev->class_dev.dev)
  strncpy(info->bus_info, dev->class_dev.dev->bus_id,
   sizeof(info->bus_info)
 - 1);
 else
  snprintf(info->bus_info, sizeof(info->bus_info) - 1,
    "PCMCIA %p", priv->hw.iobase);
}

static const struct ethtool_ops orinoco_ethtool_ops = {
 .get_drvinfo = orinoco_get_drvinfo,
 .get_link = ethtool_op_get_link,
};

/********************************************************************/
/* Module initialization                                            */
/********************************************************************/

EXPORT_SYMBOL(alloc_orinocodev);
EXPORT_SYMBOL(free_orinocodev);

EXPORT_SYMBOL(__orinoco_up);
EXPORT_SYMBOL(__orinoco_down);
EXPORT_SYMBOL(orinoco_reinit_firmware);

EXPORT_SYMBOL(orinoco_interrupt);

/* Can't be declared "const" or the whole __initdata section will
* become const */
static char version[] __initdata = DRIVER_NAME " " DRIVER_VERSION
 " (David Gibson <hermes@gibson.dropbear.id.au>, "
 "Pavel Roskin <proski@gnu.org>, et al)";

static int __init init_orinoco(void)
{
 printk(KERN_DEBUG
 "%s\n", version);
 return 0;
}

static void __exit exit_orinoco(void)
{
}

module_init(init_orinoco);
```

module_exit(exit_orinoco);

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/orinoco.c.svn-base
No license file was found, but licenses were detected in source scan.

/* orinoco_plx.c
*
* Driver for Prism II devices which would usually be driven by orinoco_cs,
* but are connected to the PCI bus by a PLX9052.
*
* Current maintainers are:
* Pavel Roskin <proski AT gnu.org>
* and David Gibson <hermes AT gibson.dropbear.id.au>
*
* (C) Copyright David Gibson, IBM Corp. 2001-2003.
* Copyright (C) 2001 Daniel Barlow
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the
 GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above. If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and
* other provisions required by the GPL. If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.
*
* Here's the general details on how the PLX9052 adapter works:
*
* - Two PCI I/O address spaces, one 0x80 long which contains the
* PLX9052 registers, and one that's 0x40 long mapped to the PCMCIA
* slot I/O address space.
*
* - One PCI memory address space, mapped to the PCMCIA attribute space

```
* (containing the CIS).
*
* Using the later, you can read through the CIS data to make sure the
* card is compatible with
 the driver. Keep in mind that the PCMCIA
* spec specifies the CIS as the lower 8 bits of each word read from
* the CIS, so to read the bytes of the CIS, read every other byte
* (0,2,4,...). Passing that test, you need to enable the I/O address
* space on the PCMCIA card via the PCMCIA COR register. This is the
* first byte following the CIS. In my case (which may not have any
* relation to what's on the PRISM2 cards), COR was at offset 0x800
* within the PCI memory space. Write 0x41 to the COR register to
* enable I/O mode and to select level triggered interrupts. To
* confirm you actually succeeded, read the COR register back and make
* sure it actually got set to 0x41, in case you have an unexpected
* card inserted.
*
* Following that, you can treat the second PCI I/O address space (the
* one that's not 0x80 in length) as the PCMCIA I/O space.
*
* Note that in the Eumitcom's source for their drivers, they register
* the interrupt as edge triggered when registering it
 with the
* Windows kernel. I don't recall how to register edge triggered on
* Linux (if it can be done at all). But in some experimentation, I
* don't see much operational difference between using either
* interrupt mode. Don't mess with the interrupt mode in the COR
* register though, as the PLX9052 wants level triggers with the way
* the serial EEPROM configures it on the WL11000.
*
* There's some other little quirks related to timing that I bumped
* into, but I don't recall right now. Also, there's two variants of
* the WL11000 I've seen, revision A1 and T2. These seem to differ
* slightly in the timings configured in the wait-state generator in
* the PLX9052. There have also been some comments from Eumitcom that
* cards shouldn't be hot swapped, apparently due to risk of cooking
* the PLX9052. I'm unsure why they believe this, as I can't see
* anything in the design that would really cause a problem, except
* for crashing drivers not written to expect it. And having
 developed
* drivers for the WL11000, I'd say it's quite tricky to write code
* that will successfully deal with a hot unplug. Very odd things
* happen on the I/O side of things. But anyway, be warned. Despite
* that, I've hot-swapped a number of times during debugging and
* driver development for various reasons (stuck WAIT# line after the
* radio card's firmware locks up).
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/orinoco_plx.c
No license file was found, but licenses were detected in source scan.

/*** -*- linux-c -*- **

Driver for Atmel at76c502 at76c504 and at76c506 wireless cards.

Copyright 2000-2001 ATMEL Corporation.
Copyright 2003-2004 Simon Kelley.

This code was developed from version 2.1.1 of the Atmel drivers,
released by Atmel corp. under the GPL in December 2002. It also
includes code from the Linux aironet drivers (C) Benjamin Reed,
and the Linux PCMCIA package, (C) David Hinds and the Linux wireless
extensions, (C) Jean Tourrilhes.

The firmware module for reading the MAC address of the card comes from
net.russotto.AtmelMACFW, written by Matthew T. Russotto and copyright
by him. net.russotto.AtmelMACFW is used under the GPL license version 2.
This file contains the module in binary form and, under the terms
of the GPL, in source form. The source is located at the end of the file.

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 software 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 Atmel wireless lan drivers; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

For all queries about this code, please contact the current author,
Simon Kelley <simon@thekelleys.org.uk> and not Atmel Corporation.

Credit is due to HP UK and Cambridge Online Systems Ltd for supplying
hardware used during development of this
driver.

**/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/atmel.c.svn-

base
No license file was found, but licenses were detected in source scan.

/* zd_usb.h: Header for USB interface implemented by ZD1211 chip
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_usb.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_usb.h
No license file was found, but licenses were detected in source scan.

/* zd_chip.c
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_chip.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_chip.c.svn-base

No license file was found, but licenses were detected in source scan.

/**

Copyright(c) 2003 - 2006 Intel Corporation. All rights reserved.

802.11 status code portion of this file from ethereal-0.10.6:
Copyright 2000, Axis Communications AB
Ethereal - Network traffic analyzer
By Gerald Combs <gerald@ethereal.com>
Copyright 1998 Gerald Combs

This program is free software; you can redistribute it and/or modify it
under the terms of version 2 of the GNU General Public License as
published by the Free Software Foundation.

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.

Contact Information:
James P. Ketrenos <ipw2100-admin@linux.intel.com>
Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124-6497

**/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/ipw2200.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/ipw2200.c.svn-base
No license file was found, but licenses were detected in source scan.

/**

Copyright(c) 2003 - 2006 Intel Corporation. All rights reserved.

This program is free software; you can redistribute it and/or modify it
under the terms of version 2 of the GNU General Public License as
published by the Free Software Foundation.

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.

Contact Information:
James P. Ketrenos
<ipw2100-admin@linux.intel.com>
Intel Corporation, 5200 N.E. Elam Young Parkway, Hillsboro, OR 97124-6497

**********/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/ipw2200.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/ipw2200.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/ipw2100.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/ipw2100.h.svn-base
No license file was found, but licenses were detected in source scan.

MODULE_LICENSE("Dual MPL/GPL");

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/airport.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/orinoco_cs.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/spectrum_cs.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/orinoco_cs.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/airport.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/spectrum_cs.c.svn-base
No license file was found, but licenses were detected in source scan.

/*
* Definitions for the AT&T GIS (formerly NCR) WaveLAN PCMCIA card:
*   An Ethernet-like radio transceiver controlled by an Intel 82593
*   coprocessor.
*

```
*
*****************************************************************************
*   Copyright 1995
*   Anthony D. Joseph
*   Massachusetts Institute of Technology
*
*   Permission to use, copy, modify, and distribute this program
*   for any purpose and without fee is hereby granted, provided
*   that this copyright and permission notice appear on all copies
*   and supporting documentation, the name of M.I.T. not be used
*   in advertising or publicity pertaining to distribution of the
*   program without specific prior permission, and notice be given
*   in supporting documentation that copying and distribution is
*   by permission of M.I.T.  M.I.T. makes no representations about
*   the suitability of this software for any purpose.  It is pro-
 *   vided "as is" without express or implied warranty.
*****************************************************************************
*
*
* Credits:
*     Special thanks to Jan Hoogendoorn of AT&T GIS Utrecht for
*       providing extremely useful information about WaveLAN PCMCIA hardware
*
*     This driver is based upon several other drivers, in particular:
*       David Hinds' Linux driver for the PCMCIA 3c589 ethernet adapter
*       Bruce Janson's Linux driver for the AT-bus WaveLAN adapter
*  Anders Klemets' PCMCIA WaveLAN adapter driver
*       Robert Morris' BSDI driver for the PCMCIA WaveLAN adapter
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wavelan_cs.h
No license file was found, but licenses were detected in source scan.

```
/*** -*- linux-c -*- ************************************************************

    Driver for Atmel at76c502 at76c504 and at76c506 wireless cards.

      Copyright 2004 Simon Kelley.

  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 software 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 Atmel wireless lan drivers; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

**********/

Found
in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/atmel_pci.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/atmel_pci.c
No license file was found, but licenses were detected in source scan.

/* orinoco_tmd.c
*
* Driver for Prism II devices which would usually be driven by orinoco_cs,
* but are connected to the PCI bus by a TMD7160.
*
* Copyright (C) 2003 Joerg Dorchain <joerg AT dorchain.net>
* based heavily upon orinoco_plx.c Copyright (C) 2001 Daniel Barlow
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions
of the GPL are applicable instead of the
* above. If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and
* other provisions required by the GPL. If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.
*
* The actual driving is done by orinoco.c, this is just resource
* allocation stuff.
*

* This driver is modeled after the orinoco_plx driver. The main
* difference is that the TMD chip has only IO port ranges and doesn't
* provide access to the PCMCIA attribute space.
*
* Pheecom sells cards with the TMD chip as "ASIC version"
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/orinoco_tmd.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/orinoco_tmd.c
No license file was found, but licenses were detected in source scan.

/* orinoco_plx.c
*
* Driver for Prism II devices which would usually be driven by orinoco_cs,
* but are connected to the PCI bus by a PLX9052.
*
* Current maintainers are:
* Pavel Roskin <proski AT gnu.org>
* and David Gibson <hermes AT gibson.dropbear.id.au>
*
* (C) Copyright David Gibson, IBM Corp. 2001-2003.
* Copyright (C) 2001 Daniel Barlow
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the
 GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above. If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and
* other provisions required by the GPL. If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.
*
* Here's the general details on how the PLX9052 adapter works:

*
* - Two PCI I/O address spaces, one 0x80 long which contains the
* PLX9052 registers, and one that's 0x40 long mapped to the PCMCIA
* slot I/O address space.
*
* - One PCI memory address space, mapped to the PCMCIA attribute space
* (containing the CIS).
*
* Using the later, you can read through the CIS data to make sure the
* card is compatible with
 the driver. Keep in mind that the PCMCIA
* spec specifies the CIS as the lower 8 bits of each word read from
* the CIS, so to read the bytes of the CIS, read every other byte
* (0,2,4,...). Passing that test, you need to enable the I/O address
* space on the PCMCIA card via the PCMCIA COR register. This is the
* first byte following the CIS. In my case (which may not have any
* relation to what's on the PRISM2 cards), COR was at offset 0x800
* within the PCI memory space. Write 0x41 to the COR register to
* enable I/O mode and to select level triggered interrupts. To
* confirm you actually succeeded, read the COR register back and make
* sure it actually got set to 0x41, in case you have an unexpected
* card inserted.
*
* Following that, you can treat the second PCI I/O address space (the
* one that's not 0x80 in length) as the PCMCIA I/O space.
*
* Note that in the Eumitcom's source for their drivers, they register
* the interrupt as edge triggered when registering it
 with the
* Windows kernel. I don't recall how to register edge triggered on
* Linux (if it can be done at all). But in some experimentation, I
* don't see much operational difference between using either
* interrupt mode. Don't mess with the interrupt mode in the COR
* register though, as the PLX9052 wants level triggers with the way
* the serial EEPROM configures it on the WL11000.
*
* There's some other little quirks related to timing that I bumped
* into, but I don't recall right now. Also, there's two variants of
* the WL11000 I've seen, revision A1 and T2. These seem to differ
* slightly in the timings configured in the wait-state generator in
* the PLX9052. There have also been some comments from Eumitcom that
* cards shouldn't be hot swapped, apparently due to risk of cooking
* the PLX9052. I'm unsure why they believe this, as I can't see
* anything in the design that would really cause a problem, except
* for crashing drivers not written to expect it. And having
 developed
* drivers for the WL11000, I'd say it's quite tricky to write code
* that will successfully deal with a hot unplug. Very odd things

```
* happen on the I/O side of things. But anyway, be warned. Despite
* that, I've hot-swapped a number of times during debugging and
* driver development for various reasons (stuck WAIT# line after the
* radio card's firmware locks up).
*/

#define DRIVER_NAME "orinoco_plx"
#define PFX DRIVER_NAME ": "

#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/delay.h>
#include <linux/pci.h>
#include <pcmcia/cisreg.h>

#include "orinoco.h"
#include "orinoco_pci.h"

#define COR_OFFSET (0x3e0) /* COR attribute offset of Prism2 PC card */
#define COR_VALUE (COR_LEVEL_REQ | COR_FUNC_ENA) /* Enable PC card with interrupt in level trigger
*/
#define COR_RESET     (0x80) /* reset bit in the COR register */
#define PLX_RESET_TIME (500) /* milliseconds */

#define PLX_INTCSR  0x4c /* Interrupt Control & Status Register
 */
#define PLX_INTCSR_INTEN (1<<6) /* Interrupt Enable bit */

/*
* Do a soft reset of the card using the Configuration Option Register
*/
static int orinoco_plx_cor_reset(struct orinoco_private *priv)
{
 hermes_t *hw = &priv->hw;
 struct orinoco_pci_card *card = priv->card;
 unsigned long timeout;
 u16 reg;

 iowrite8(COR_VALUE | COR_RESET, card->attr_io + COR_OFFSET);
 mdelay(1);

 iowrite8(COR_VALUE, card->attr_io + COR_OFFSET);
 mdelay(1);

 /* Just in case, wait more until the card is no longer busy */
 timeout = jiffies + (PLX_RESET_TIME * HZ / 1000);
 reg = hermes_read_regn(hw, CMD);
```

```
 while (time_before(jiffies, timeout) && (reg & HERMES_CMD_BUSY)) {
  mdelay(1);
  reg = hermes_read_regn(hw, CMD);
 }

 /* Still busy? */
 if (reg & HERMES_CMD_BUSY) {
  printk(KERN_ERR PFX "Busy timeout\n");
  return -ETIMEDOUT;
 }

 return 0;
}

static int orinoco_plx_hw_init(struct orinoco_pci_card *card)
{
 int i;
 u32 csr_reg;
 static const u8 cis_magic[] = {
  0x01, 0x03, 0x00, 0x00, 0xff,
 0x17, 0x04, 0x67
 };

 printk(KERN_DEBUG PFX "CIS: ");
 for (i = 0; i < 16; i++) {
  printk("%02X:", ioread8(card->attr_io + (i << 1)));
 }
 printk("\n");

 /* Verify whether a supported PC card is present */
 /* FIXME: we probably need to be smarted about this */
 for (i = 0; i < sizeof(cis_magic); i++) {
  if (cis_magic[i] != ioread8(card->attr_io + (i << 1))) {
   printk(KERN_ERR PFX "The CIS value of Prism2 PC "
          "card is unexpected\n");
   return -ENODEV;
  }
 }

 /* bjoern: We need to tell the card to enable interrupts, in
    case the serial eprom didn't do this already.  See the
    PLX9052 data book, p8-1 and 8-24 for reference. */
 csr_reg = ioread32(card->bridge_io + PLX_INTCSR);
 if (!(csr_reg & PLX_INTCSR_INTEN)) {
  csr_reg |= PLX_INTCSR_INTEN;
  iowrite32(csr_reg, card->bridge_io + PLX_INTCSR);
  csr_reg = ioread32(card->bridge_io + PLX_INTCSR);
  if (!(csr_reg & PLX_INTCSR_INTEN)) {
```

```
  printk(KERN_ERR PFX "Cannot enable interrupts\n");
  return -EIO;
 }
}

return
0;
}

static int orinoco_plx_init_one(struct pci_dev *pdev,
    const struct pci_device_id *ent)
{
 int err;
 struct orinoco_private *priv;
 struct orinoco_pci_card *card;
 struct net_device *dev;
 void __iomem *hermes_io, *attr_io, *bridge_io;

 err = pci_enable_device(pdev);
 if (err) {
  printk(KERN_ERR PFX "Cannot enable PCI device\n");
  return err;
 }

 err = pci_request_regions(pdev, DRIVER_NAME);
 if (err) {
  printk(KERN_ERR PFX "Cannot obtain PCI resources\n");
  goto fail_resources;
 }

 bridge_io = pci_iomap(pdev, 1, 0);
 if (!bridge_io) {
  printk(KERN_ERR PFX "Cannot map bridge registers\n");
  err = -EIO;
  goto fail_map_bridge;
 }

 attr_io = pci_iomap(pdev, 2, 0);
 if (!attr_io) {
  printk(KERN_ERR PFX "Cannot map PCMCIA attributes\n");
  err = -EIO;
  goto fail_map_attr;
 }

 hermes_io = pci_iomap(pdev, 3, 0);
 if (!hermes_io) {
  printk(KERN_ERR PFX "Cannot map chipset registers\n");
  err = -EIO;
```

```
 goto fail_map_hermes;
}

/* Allocate network device */
dev
= alloc_orinocodev(sizeof(*card), orinoco_plx_cor_reset);
if (!dev) {
 printk(KERN_ERR PFX "Cannot allocate network device\n");
 err = -ENOMEM;
 goto fail_alloc;
}

priv = netdev_priv(dev);
card = priv->card;
card->bridge_io = bridge_io;
card->attr_io = attr_io;
SET_MODULE_OWNER(dev);
SET_NETDEV_DEV(dev, &pdev->dev);

hermes_struct_init(&priv->hw, hermes_io, HERMES_16BIT_REGSPACING);

err = request_irq(pdev->irq, orinoco_interrupt, IRQF_SHARED,
    dev->name, dev);
if (err) {
 printk(KERN_ERR PFX "Cannot allocate IRQ %d\n", pdev->irq);
 err = -EBUSY;
 goto fail_irq;
}

err = orinoco_plx_hw_init(card);
if (err) {
 printk(KERN_ERR PFX "Hardware initialization failed\n");
 goto fail;
}

err = orinoco_plx_cor_reset(priv);
if (err) {
 printk(KERN_ERR PFX "Initial reset failed\n");
 goto fail;
}

err = register_netdev(dev);
if (err) {
 printk(KERN_ERR PFX "Cannot register network device\n");
 goto fail;
}

pci_set_drvdata(pdev, dev);
```

```
 printk(KERN_DEBUG
 "%s: " DRIVER_NAME " at %s\n", dev->name,
       pci_name(pdev));

 return 0;

fail:
 free_irq(pdev->irq, dev);

fail_irq:
 pci_set_drvdata(pdev, NULL);
 free_orinocodev(dev);

fail_alloc:
 pci_iounmap(pdev, hermes_io);

fail_map_hermes:
 pci_iounmap(pdev, attr_io);

fail_map_attr:
 pci_iounmap(pdev, bridge_io);

fail_map_bridge:
 pci_release_regions(pdev);

fail_resources:
 pci_disable_device(pdev);

 return err;
}

static void __devexit orinoco_plx_remove_one(struct pci_dev *pdev)
{
 struct net_device *dev = pci_get_drvdata(pdev);
 struct orinoco_private *priv = netdev_priv(dev);
 struct orinoco_pci_card *card = priv->card;

 unregister_netdev(dev);
 free_irq(pdev->irq, dev);
 pci_set_drvdata(pdev, NULL);
 free_orinocodev(dev);
 pci_iounmap(pdev, priv->hw.iobase);
 pci_iounmap(pdev, card->attr_io);
 pci_iounmap(pdev, card->bridge_io);
 pci_release_regions(pdev);
 pci_disable_device(pdev);
}
```

```
static struct pci_device_id orinoco_plx_id_table[] = {
 {0x111a, 0x1023, PCI_ANY_ID,
 PCI_ANY_ID,}, /* Siemens SpeedStream SS1023 */
 {0x1385, 0x4100, PCI_ANY_ID, PCI_ANY_ID,}, /* Netgear MA301 */
 {0x15e8, 0x0130, PCI_ANY_ID, PCI_ANY_ID,}, /* Correga  - does this work? */
 {0x1638, 0x1100, PCI_ANY_ID, PCI_ANY_ID,}, /* SMC EZConnect SMC2602W,
       Eumitcom PCI WL11000,
       Addtron AWA-100 */
 {0x16ab, 0x1100, PCI_ANY_ID, PCI_ANY_ID,}, /* Global Sun Tech GL24110P */
 {0x16ab, 0x1101, PCI_ANY_ID, PCI_ANY_ID,}, /* Reported working, but unknown */
 {0x16ab, 0x1102, PCI_ANY_ID, PCI_ANY_ID,}, /* Linksys WDT11 */
 {0x16ec, 0x3685, PCI_ANY_ID, PCI_ANY_ID,}, /* USR 2415 */
 {0xec80, 0xec00, PCI_ANY_ID, PCI_ANY_ID,}, /* Belkin F5D6000 tested by
       Brendan W. McAdams <rit AT jacked-in.org> */
 {0x10b7, 0x7770, PCI_ANY_ID, PCI_ANY_ID,}, /* 3Com AirConnect PCI tested by
       Damien Persohn <damien AT persohn.net> */
 {0,},
};

MODULE_DEVICE_TABLE(pci, orinoco_plx_id_table);

static struct pci_driver orinoco_plx_driver = {
 .name  = DRIVER_NAME,
 .id_table =
 orinoco_plx_id_table,
 .probe  = orinoco_plx_init_one,
 .remove  = __devexit_p(orinoco_plx_remove_one),
 .suspend = orinoco_pci_suspend,
 .resume  = orinoco_pci_resume,
};

static char version[] __initdata = DRIVER_NAME " " DRIVER_VERSION
 " (Pavel Roskin <proski@gnu.org>,"
 " David Gibson <hermes@gibson.dropbear.id.au>,"
 " Daniel Barlow <dan@telent.net>)";
MODULE_AUTHOR("Daniel Barlow <dan@telent.net>");
MODULE_DESCRIPTION("Driver for wireless LAN cards using the PLX9052 PCI bridge");
MODULE_LICENSE("Dual MPL/GPL");

static int __init orinoco_plx_init(void)
{
 printk(KERN_DEBUG "%s\n", version);
 return pci_register_driver(&orinoco_plx_driver);
}

static void __exit orinoco_plx_exit(void)
{
 pci_unregister_driver(&orinoco_plx_driver);
```

```
}

module_init(orinoco_plx_init);
module_exit(orinoco_plx_exit);

/*
* Local variables:
*  c-indent-level: 8
*  c-basic-offset: 8
*  tab-width: 8
* End:
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/orinoco_plx.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/*** -*- linux-c -*- ************************************************************

    Driver for Atmel at76c502 at76c504 and at76c506 wireless cards.

      Copyright 2000-2001 ATMEL Corporation.
      Copyright 2003 Simon Kelley.

  This code was developed from version 2.1.1 of the Atmel drivers,
  released by Atmel corp. under the GPL in December 2002. It also
  includes code from the Linux aironet drivers (C) Benjamin Reed,
  and the Linux PCMCIA package, (C) David Hinds.

  For all queries about this code, please contact the current author,
  Simon Kelley <simon@thekelleys.org.uk> and not Atmel Corporation.

  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 software 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 Atmel wireless lan drivers; if not, write to the Free Software
  Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

**********/
/*
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.

In addition:

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.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/atmel_cs.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/atmel_cs.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/*

 Broadcom BCM43xx wireless driver

 PIO Transmission

 Copyright (c) 2005 Michael Buesch <mbuesch@freenet.de>

 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; see the file COPYING.  If not, write to
 the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
 Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-
base/bcm43xx_pio.c.svn-base
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_pio.c
No license file was found, but licenses were detected in source scan.

/*
*
*  Copyright (C) 2002 Intersil Americas Inc.
*  Copyright (C) 2003 Herbert Valerio Riedel <hvr@gnu.org>
*  Copyright (C) 2003 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
*
*  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
*
*  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
*
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_dev.c
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_dev.c.svn-base
No license file was found, but licenses were detected in source scan.

/*
* Copyright (C) 1997 Cullen Jennings
* Copyright (C) 1998 Elmer Joandiu, elmer@ylenurme.ee
* GNU General Public License applies
* This module provides support for the Arlan 655 card made by Aironet
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/arlan-main.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/arlan-main.c
No license file was found, but licenses were detected in source scan.

/*

 Broadcom BCM43xx wireless driver

 Transmission (TX/RX) related functions.

 Copyright (c) 2005 Martin Langer <martin-langer@gmx.de>,
                    Stefano Brivio <st3@riseup.net>
                    Michael Buesch <mbuesch@freenet.de>
                    Danny van Dyk <kugelfang@gentoo.org>
                    Andreas Jaggi <andreas.jaggi@waterwave.ch>

 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; see the file COPYING. If not, write to
the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_xmit.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_xmit.c
No license file was found, but licenses were detected in source scan.

/*===

Aironet driver for 4500 and 4800 series cards

This code is released under both the GPL version 2 and BSD licenses.
Either license may be used. The respective licenses are found at
the end of this file.

This code was developed by Benjamin Reed <breed@users.sourceforge.net>
including portions of which come from the Aironet PC4500
Developer's Reference Manual and used with permission. Copyright
(C) 1999 Benjamin Reed. All Rights Reserved. Permission to use
code in the Developer's manual was granted for this driver by
Aironet. Major code contributions were received from Javier Achirica
<achirica@users.sourceforge.net> and Jean Tourrilhes <jt@hpl.hp.com>.
Code was also integrated from the Cisco Aironet driver for Linux.
Support for MPI350 cards was added by Fabrice Bellet
<fabrice@bellet.info>.

===*/
/*
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.

In addition:

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.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/airo.c
No license file was found, but licenses were detected in source scan.

```
/*
*
*  Copyright (C) 2002 Intersil Americas Inc.
*  Copyright 2004 Jens Maurer <Jens.Maurer@gmx.net>
*
*  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
*
*  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
*
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_mgt.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_mgt.c.svn-base
No license file was found, but licenses were detected in source scan.

/*===

  Aironet driver for 4500 and 4800 series cards

  This code is released under both the GPL version 2 and BSD licenses.
  Either license may be used. The respective licenses are found at
  the end of this file.

  This code was developed by Benjamin Reed <breed@users.sourceforge.net>
  including portions of which come from the Aironet PC4500
  Developer's Reference Manual and used with permission. Copyright
  (C) 1999 Benjamin Reed. All Rights Reserved. Permission to use
  code in the Developer's manual was granted for this driver by
  Aironet. Major code contributions were received from Javier Achirica
  <achirica@users.sourceforge.net> and Jean Tourrilhes <jt@hpl.hp.com>.
  Code was also integrated from the Cisco Aironet driver for Linux.
  Support for MPI350 cards was added by Fabrice Bellet
  <fabrice@bellet.info>.

===*/

Found
 in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/airo.c.svn-base
No license file was found, but licenses were detected in source scan.

/*

 Broadcom BCM43xx wireless driver

 ethtool support

 Copyright (c) 2006 Jason Lunz <lunz@falooley.org>

 Some code in this file is derived from the 8139too.c driver
 Copyright (C) 2002 Jeff Garzik

 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; see the file COPYING. If not, write to
the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
Boston, MA 02110-1301,
USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_ethtool.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_ethtool.c
No license file was found, but licenses were detected in source scan.

/*
*
* Copyright (C) 2002 Intersil Americas Inc.
* Copyright (C) 2003 Herbert Valerio Riedel <hvr@gnu.org>
*
* 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
*
* 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
*
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_hotplug.c.svn-base
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_hotplug.c

No license file was found, but licenses were detected in source scan.

/* zd_chip.h
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_chip.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_chip.h
No license file was found, but licenses were detected in source scan.

/* orinoco_nortel.c
*
* Driver for Prism II devices which would usually be driven by orinoco_cs,
* but are connected to the PCI bus by a PCI-to-PCMCIA adapter used in
* Nortel emobility, Symbol LA-4113 and Symbol LA-4123.
*
* Copyright (C) 2002 Tobias Hoffmann
*   (C) 2003 Christoph Jungegger <disdos@traum404.de>
*
* Some of this code is borrowed from orinoco_plx.c
* Copyright (C) 2001 Daniel Barlow
* Some of this code is borrowed from orinoco_pci.c
* Copyright (C) 2001 Jean Tourrilhes
* Some of this code is "inspired" by linux-wlan-ng-0.1.10, but nothing
* has been copied from it. linux-wlan-ng-0.1.10 is originally :
* Copyright (C) 1999 AbsoluteValue Systems, Inc. All Rights Reserved.
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
 * at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"

* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above. If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and
* other provisions required by the GPL. If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/orinoco_nortel.c.svn-base
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/orinoco_nortel.c
No license file was found, but licenses were detected in source scan.

/*
* This software may only be used and distributed
* according to the terms of the GNU General Public License.
*
* This software was developed as a component of the
* Linux operating system.
* It is based on other device drivers and information
* either written or supplied by:
* Ajay Bakre (bakre@paul.rutgers.edu),
* Donald Becker (becker@scyld.com),
* Loeke Brederveld (Loeke.Brederveld@Utrecht.NCR.com),
* Anders Klemets (klemets@it.kth.se),
* Vladimir V. Kolpakov (w@stier.koenig.ru),
* Marc Meertens (Marc.Meertens@Utrecht.NCR.com),
* Pauline Middelink (middelin@polyware.iaf.nl),
* Robert Morris (rtm@das.harvard.edu),
* Jean Tourrilhes (jt@hplb.hpl.hp.com),
* Girish Welling (welling@paul.rutgers.edu),
*
* Thanks go also to:
* James Ashton (jaa101@syseng.anu.edu.au),
* Alan Cox (alan@redhat.com),
* Allan Creighton (allanc@cs.usyd.edu.au),
* Matthew Geier (matthew@cs.usyd.edu.au),
* Remo di Giovanni

(remo@cs.usyd.edu.au),
* Eckhard Grah (grah@wrcs1.urz.uni-wuppertal.de),
* Vipul Gupta (vgupta@cs.binghamton.edu),
* Mark Hagan (mhagan@wtcpost.daytonoh.NCR.COM),
* Tim Nicholson (tim@cs.usyd.edu.au),
* Ian Parkin (ian@cs.usyd.edu.au),
* John Rosenberg (johnr@cs.usyd.edu.au),
* George Rossi (george@phm.gov.au),
* Arthur Scott (arthur@cs.usyd.edu.au),
* Peter Storey,
* for their assistance and advice.
*
* Please send bug reports, updates, comments to:
*
* Bruce Janson Email: bruce@cs.usyd.edu.au
* Basser Department of Computer Science Phone: +61-2-9351-3423
* University of Sydney, N.S.W., 2006, AUSTRALIA Fax: +61-2-9351-3838
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wavelan.c
No license file was found, but licenses were detected in source scan.

/* hermes.c
*
* Driver core for the "Hermes" wireless MAC controller, as used in
* the Lucent Orinoco and Cabletron RoamAbout cards. It should also
* work on the hfa3841 and hfa3842 MAC controller chips used in the
* Prism II chipsets.
*
* This is not a complete driver, just low-level access routines for
* the MAC controller itself.
*
* Based on the prism2 driver from Absolute Value Systems' linux-wlan
* project, the Linux wvlan_cs driver, Lucent's HCF-Light
* (wvlan_hcf.c) library, and the NetBSD wireless driver (in no
* particular order).
*
* Copyright (C) 2000, David Gibson, Linuxcare Australia.
* (C) Copyright David Gibson, IBM Corp. 2001-2003.
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software

distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above. If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and
* other provisions required by the GPL. If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hermes.c
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/hermes.c.svn-base
No license file was found, but licenses were detected in source scan.

/* zd_util.h
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_util.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_util.h.svn-base
No license file was found, but licenses were detected in source scan.

/*

```
*
*  Copyright (C) 2002 Intersil Americas Inc.
*  Copyright (C) 2004 Aurelien Alleaume <slts@free.fr>
*  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
*
*  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
*
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/islpci_eth.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/islpci_eth.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/* zd_mac.h
*
* 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
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_mac.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_mac.h.svn-base
No license file was found, but licenses were detected in source scan.

```
/*
```

```
* Wavelan Pcmcia driver
*
*  Jean II - HPLB '96
*
* Reorganisation and extension of the driver.
* Original copyright follow. See wavelan_cs.p.h for details.
*
* This code is derived from Anthony D. Joseph's code and all the changes here
* are also under the original copyright below.
*
* This code supports version 2.00 of WaveLAN/PCMCIA cards (2.4GHz), and
* can work on Linux 2.0.36 with support of David Hinds' PCMCIA Card Services
*
* Joe Finney (joe@comp.lancs.ac.uk) at Lancaster University in UK added
* critical code in the routine to initialize the Modem Management Controller.
*
* Thanks to Alan Cox and Bruce Janson for their advice.
*
* -- Yunzhou Li (scip4166@nus.sg)
*
#ifdef WAVELAN_ROAMING
* Roaming support added 07/22/98 by Justin Seger (jseger@media.mit.edu)
* based on patch by Joe Finney from Lancaster University.
#endif
*
* Lucent (formerly AT&T GIS, formerly NCR) WaveLAN PCMCIA
 card: An
* Ethernet-like radio transceiver controlled by an Intel 82593 coprocessor.
*
*   A non-shared memory PCMCIA ethernet driver for linux
*
* ISA version modified to support PCMCIA by Anthony Joseph (adj@lcs.mit.edu)
*
*
* Joseph O'Sullivan & John Langford (josullvn@cs.cmu.edu & jcl@cs.cmu.edu)
*
* Apr 2 '98  made changes to bring the i82593 control/int handling in line
*            with offical specs...
*
**********************************************************************
*   Copyright 1995
*   Anthony D. Joseph
*   Massachusetts Institute of Technology
*
*   Permission to use, copy, modify, and distribute this program
*   for any purpose and without fee is hereby granted, provided
*   that this copyright and permission notice appear on all copies
*   and supporting documentation, the name of M.I.T. not be used
```

```
*   in advertising or publicity pertaining to distribution of the
*   program without specific prior permission, and notice be given
*
    in supporting documentation that copying and distribution is
*   by permission of M.I.T.  M.I.T. makes no representations about
*   the suitability of this software for any purpose.  It is pro-
*   vided "as is" without express or implied warranty.
************************************************************************
*
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/wavelan_cs.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wavelan_cs.c
No license file was found, but licenses were detected in source scan.

```
/* orinoco.c - (formerly known as dldwd_cs.c and orinoco_cs.c)
*
* A driver for Hermes or Prism 2 chipset based PCMCIA wireless
* adaptors, with Lucent/Agere, Intersil or Symbol firmware.
*
* Current maintainers (as of 29 September 2003) are:
*  Pavel Roskin <proski AT gnu.org>
* and David Gibson <hermes AT gibson.dropbear.id.au>
*
* (C) Copyright David Gibson, IBM Corporation 2001-2003.
* Copyright (C) 2000 David Gibson, Linuxcare Australia.
* With some help from :
* Copyright (C) 2001 Jean Tourrilhes, HP Labs
* Copyright (C) 2001 Benjamin Herrenschmidt
*
* Based on dummy_cs.c 1.27 2000/06/12 21:27:25
*
* Portions based on wvlan_cs.c 1.0.6, Copyright Andreas Neuhaus <andy
* AT fasta.fh-dortmund.de>
*      http://www.stud.fh-dortmund.de/~andy/wvlan/
*
* The contents of this file are subject to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance
 with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
```

```
*
* The initial developer of the original code is David A. Hinds
* <dahinds AT users.sourceforge.net>.  Portions created by David
* A. Hinds are Copyright (C) 1999 David A. Hinds.  All Rights
* Reserved.
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above.  If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace
 them with the notice and
* other provisions required by the GPL.  If you do not delete the
* provisions above, a recipient may use your version of this file
* under either the MPL or the GPL.  */
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/orinoco.c
No license file was found, but licenses were detected in source scan.

```
/*
* Wavelan Pcmcia driver
*
*  Jean II - HPLB '96
*
* Reorganization and extension of the driver.
* Original copyright follow. See wavelan_cs.h for details.
*
* This file contain the declarations of the Wavelan hardware. Note that
* the Pcmcia Wavelan include a i82593 controller (see definitions in
* file i82593.h).
*
* The main difference between the pcmcia hardware and the ISA one is
* the Ethernet Controller (i82593 instead of i82586). The i82593 allow
* only one send buffer. The PSA (Parameter Storage Area : EEprom for
* permanent storage of various info) is memory mapped, but not the
* MMI (Modem Management Interface).
*/

/*
* Definitions for the AT&T GIS (formerly NCR) WaveLAN PCMCIA card:
*   An Ethernet-like radio transceiver controlled by an Intel 82593
*   coprocessor.
*
*
**********************************************************************
```

```
*   Copyright 1995
*   Anthony
 D. Joseph
*   Massachusetts Institute of Technology
*
*   Permission to use, copy, modify, and distribute this program
*   for any purpose and without fee is hereby granted, provided
*   that this copyright and permission notice appear on all copies
*   and supporting documentation, the name of M.I.T. not be used
*   in advertising or publicity pertaining to distribution of the
*   program without specific prior permission, and notice be given
*   in supporting documentation that copying and distribution is
*   by permission of M.I.T.  M.I.T. makes no representations about
*   the suitability of this software for any purpose.  It is pro-
*   vided "as is" without express or implied warranty.
**********************************************************************
*
*
* Credits:
*     Special thanks to Jan Hoogendoorn of AT&T GIS Utrecht for
*       providing extremely useful information about WaveLAN PCMCIA hardware
*
*     This driver is based upon several
 other drivers, in particular:
*       David Hinds' Linux driver for the PCMCIA 3c589 ethernet adapter
*       Bruce Janson's Linux driver for the AT-bus WaveLAN adapter
*  Anders Klemets' PCMCIA WaveLAN adapter driver
*       Robert Morris' BSDI driver for the PCMCIA WaveLAN adapter
*/

#ifndef _WAVELAN_CS_H
#define _WAVELAN_CS_H

/************************** MAGIC NUMBERS ***************************/

/* The detection of the wavelan card is made by reading the MAC address
* from the card and checking it. If you have a non AT&T product (OEM,
* like DEC RoamAbout, or Digital Ocean, Epson, ...), you must modify this
* part to accommodate your hardware...
*/
static const unsigned char MAC_ADDRESSES[][3] =
{
 { 0x08, 0x00, 0x0E },  /* AT&T Wavelan (standard) & DEC RoamAbout */
 { 0x08, 0x00, 0x6A },  /* AT&T Wavelan (alternate) */
 { 0x00, 0x00, 0xE1 },  /* Hitachi Wavelan */
 { 0x00, 0x60, 0x1D }  /* Lucent Wavelan (another one) */
 /* Add your card here and send me the
```

```
 patch ! */
};

/*
* Constants used to convert channels to frequencies
*/

/* Frequency available in the 2.0 modem, in units of 250 kHz
* (as read in the offset register of the dac area).
* Used to map channel numbers used by `wfreqsel' to frequencies
*/
static const short channel_bands[] = { 0x30, 0x58, 0x64, 0x7A, 0x80, 0xA8,
       0xD0, 0xF0, 0xF8, 0x150 };

/* Frequencies of the 1.0 modem (fixed frequencies).
* Use to map the PSA `subband' to a frequency
* Note : all frequencies apart from the first one need to be multiplied by 10
*/
static const int fixed_bands[] = { 915e6, 2.425e8, 2.46e8, 2.484e8, 2.4305e8 };


/***************************** PC INTERFACE ****************************/

/* WaveLAN host interface definitions */

#define LCCR(base) (base)  /* LAN Controller Command Register */
#define LCSR(base) (base)  /* LAN Controller Status Register */
#define HACR(base) (base+0x1) /* Host Adapter Command Register */
#define HASR(base) (base+0x1) /* Host Adapter Status Register
 */
#define PIORL(base) (base+0x2) /* Program I/O Register Low */
#define RPLL(base) (base+0x2) /* Receive Pointer Latched Low */
#define PIORH(base) (base+0x3) /* Program I/O Register High */
#define RPLH(base) (base+0x3) /* Receive Pointer Latched High */
#define PIOP(base) (base+0x4) /* Program I/O Port */
#define MMR(base) (base+0x6) /* MMI Address Register */
#define MMD(base) (base+0x7) /* MMI Data Register */

/* Host Adaptor Command Register bit definitions */

#define HACR_LOF   (1 << 3) /* Lock Out Flag, toggle every 250ms */
#define HACR_PWR_STAT   (1 << 4) /* Power State, 1=active, 0=sleep */
#define HACR_TX_DMA_RESET (1 << 5) /* Reset transmit DMA ptr on high */
#define HACR_RX_DMA_RESET (1 << 6) /* Reset receive DMA ptr on high */
#define HACR_ROM_WEN   (1 << 7) /* EEPROM write enabled when true */

#define HACR_RESET             (HACR_TX_DMA_RESET | HACR_RX_DMA_RESET)
#define HACR_DEFAULT  (HACR_PWR_STAT)
```

```
/* Host Adapter Status Register bit definitions */

#define
 HASR_MMI_BUSY (1 << 2) /* MMI is busy when true */
#define HASR_LOF (1 << 3) /* Lock out flag status */
#define HASR_NO_CLK (1 << 4) /* active when modem not connected */

/* Miscellaneous bit definitions */

#define PIORH_SEL_TX (1 << 5) /* PIOR points to 0=rx/1=tx buffer */
#define MMR_MMI_WR (1 << 0) /* Next MMI cycle is 0=read, 1=write */
#define PIORH_MASK 0x1f  /* only low 5 bits are significant */
#define RPLH_MASK 0x1f  /* only low 5 bits are significant */
#define MMI_ADDR_MASK 0x7e  /* Bits 1-6 of MMR are significant */

/* Attribute Memory map */

#define CIS_ADDR 0x0000  /* Card Information Status Register */
#define PSA_ADDR 0x0e00  /* Parameter Storage Area address */
#define EEPROM_ADDR 0x1000  /* EEPROM address (unused ?) */
#define COR_ADDR 0x4000  /* Configuration Option Register */

/* Configuration Option Register bit definitions */

#define COR_CONFIG (1 << 0) /* Config Index, 0 when unconfigured */
#define COR_SW_RESET (1 << 7) /* Software Reset on true */
#define
 COR_LEVEL_IRQ (1 << 6) /* Level IRQ */

/* Local Memory map */

#define RX_BASE  0x0000  /* Receive memory, 8 kB */
#define TX_BASE  0x2000  /* Transmit memory, 2 kB */
#define UNUSED_BASE 0x2800  /* Unused, 22 kB */
#define RX_SIZE  (TX_BASE-RX_BASE) /* Size of receive area */
#define RX_SIZE_SHIFT 6  /* Bits to shift in stop register */

#define TRUE  1
#define FALSE 0

#define MOD_ENAL 1
#define MOD_PROM 2

/* Size of a MAC address */
#define WAVELAN_ADDR_SIZE 6

/* Maximum size of Wavelan packet */
```

```
#define WAVELAN_MTU 1500

#define MAXDATAZ  (6 + 6 + 2 + WAVELAN_MTU)

/********************** PARAMETER STORAGE AREA **********************/

/*
* Parameter Storage Area (PSA).
*/
typedef struct psa_t psa_t;
struct psa_t
{
 /* For the PCMCIA Adapter, locations 0x00-0x0F are unused and fixed at 00 */
 unsigned char psa_io_base_addr_1; /* [0x00] Base address 1 ??? */
 unsigned char psa_io_base_addr_2; /* [0x01] Base address 2 */
 unsigned char psa_io_base_addr_3; /* [0x02] Base address
 3 */
 unsigned char psa_io_base_addr_4; /* [0x03] Base address 4 */
 unsigned char psa_rem_boot_addr_1; /* [0x04] Remote Boot Address 1 */
 unsigned char psa_rem_boot_addr_2; /* [0x05] Remote Boot Address 2 */
 unsigned char psa_rem_boot_addr_3; /* [0x06] Remote Boot Address 3 */
 unsigned char psa_holi_params; /* [0x07] HOst Lan Interface (HOLI) Parameters */
 unsigned char psa_int_req_no;  /* [0x08] Interrupt Request Line */
 unsigned char psa_unused0[7];  /* [0x09-0x0F] unused */

 unsigned char psa_univ_mac_addr[WAVELAN_ADDR_SIZE]; /* [0x10-0x15] Universal (factory) MAC Address
*/
 unsigned char psa_local_mac_addr[WAVELAN_ADDR_SIZE]; /* [0x16-1B] Local MAC Address */
 unsigned char psa_univ_local_sel; /* [0x1C] Universal Local Selection */
#define  PSA_UNIVERSAL 0  /* Universal (factory) */
#define  PSA_LOCAL 1  /* Local */
 unsigned char psa_comp_number; /* [0x1D] Compatability Number: */
#define  PSA_COMP_PC_AT_915 0  /* PC-AT 915 MHz */
#define  PSA_COMP_PC_MC_915 1
  /* PC-MC 915 MHz */
#define  PSA_COMP_PC_AT_2400 2  /* PC-AT 2.4 GHz */
#define  PSA_COMP_PC_MC_2400 3  /* PC-MC 2.4 GHz */
#define  PSA_COMP_PCMCIA_915 4  /* PCMCIA 915 MHz or 2.0 */
 unsigned char psa_thr_pre_set; /* [0x1E] Modem Threshold Preset */
 unsigned char psa_feature_select; /* [0x1F] Call code required (1=on) */
#define  PSA_FEATURE_CALL_CODE 0x01  /* Call code required (Japan) */
 unsigned char psa_subband;  /* [0x20] Subband */
#define  PSA_SUBBAND_915  0 /* 915 MHz or 2.0 */
#define  PSA_SUBBAND_2425 1 /* 2425 MHz */
#define  PSA_SUBBAND_2460 2 /* 2460 MHz */
#define  PSA_SUBBAND_2484 3 /* 2484 MHz */
#define  PSA_SUBBAND_2430_5 4 /* 2430.5 MHz */
 unsigned char psa_quality_thr; /* [0x21] Modem Quality Threshold */
```

```
 unsigned char psa_mod_delay;  /* [0x22] Modem Delay ??? (reserved) */
 unsigned char psa_nwid[2];  /* [0x23-0x24] Network ID */
 unsigned char psa_nwid_select; /* [0x25] Network ID Select On Off */
 unsigned char psa_encryption_select; /* [0x26]
 Encryption On Off */
 unsigned char psa_encryption_key[8]; /* [0x27-0x2E] Encryption Key */
 unsigned char psa_databus_width; /* [0x2F] AT bus width select 8/16 */
 unsigned char psa_call_code[8]; /* [0x30-0x37] (Japan) Call Code */
 unsigned char psa_nwid_prefix[2]; /* [0x38-0x39] Roaming domain */
 unsigned char psa_reserved[2]; /* [0x3A-0x3B] Reserved - fixed 00 */
 unsigned char psa_conf_status; /* [0x3C] Conf Status, bit 0=1:config*/
 unsigned char psa_crc[2];  /* [0x3D] CRC-16 over PSA */
 unsigned char psa_crc_status;  /* [0x3F] CRC Valid Flag */
};

/* Size for structure checking (if padding is correct) */
#define PSA_SIZE 64

/* Calculate offset of a field in the above structure
* Warning : only even addresses are used */
#define psaoff(p,f)  ((unsigned short) ((void *)(&((psa_t *) ((void *) NULL + (p)))->f) - (void *) NULL))

/********************* MODEM MANAGEMENT INTERFACE *********************/

/*
* Modem Management Controller (MMC) write structure.
*/
typedef
 struct mmw_t mmw_t;
struct mmw_t
{
 unsigned char mmw_encr_key[8]; /* encryption key */
 unsigned char mmw_encr_enable; /* enable/disable encryption */
#define MMW_ENCR_ENABLE_MODE 0x02 /* Mode of security option */
#define MMW_ENCR_ENABLE_EN 0x01 /* Enable security option */
 unsigned char mmw_unused0[1];  /* unused */
 unsigned char mmw_des_io_invert; /* Encryption option */
#define MMW_DES_IO_INVERT_RES 0x0F /* Reserved */
#define MMW_DES_IO_INVERT_CTRL 0xF0 /* Control ??? (set to 0) */
 unsigned char mmw_unused1[5];  /* unused */
 unsigned char mmw_loopt_sel;  /* looptest selection */
#define MMW_LOOPT_SEL_DIS_NWID 0x40 /* disable NWID filtering */
#define MMW_LOOPT_SEL_INT 0x20 /* activate Attention Request */
#define MMW_LOOPT_SEL_LS 0x10 /* looptest w/o collision avoidance */
#define MMW_LOOPT_SEL_LT3A 0x08 /* looptest 3a */
#define MMW_LOOPT_SEL_LT3B 0x04 /* looptest 3b */
#define MMW_LOOPT_SEL_LT3C 0x02 /* looptest 3c */
#define MMW_LOOPT_SEL_LT3D 0x01 /* looptest 3d */
```

```
 unsigned char mmw_jabber_enable; /* jabber timer enable */
/* Abort transmissions > 200 ms */
unsigned char mmw_freeze;  /* freeze / unfreeeze signal level */
/* 0 : signal level & qual updated for every new message, 1 : frozen */
unsigned char mmw_anten_sel;  /* antenna selection */
#define MMW_ANTEN_SEL_SEL 0x01 /* direct antenna selection */
#define MMW_ANTEN_SEL_ALG_EN 0x02 /* antenna selection algo. enable */
unsigned char mmw_ifs;  /* inter frame spacing */
/* min time between transmission in bit periods (.5 us) - bit 0 ignored */
unsigned char mmw_mod_delay;   /* modem delay (synchro) */
unsigned char mmw_jam_time;  /* jamming time (after collision) */
unsigned char mmw_unused2[1];  /* unused */
unsigned char mmw_thr_pre_set; /* level threshold preset */
/* Discard all packet with signal < this value (4) */
unsigned char mmw_decay_prm;  /* decay parameters */
unsigned char mmw_decay_updat_prm; /* decay update parameterz */
unsigned char mmw_quality_thr; /*
quality (z-quotient) threshold */
/* Discard all packet with quality < this value (3) */
unsigned char mmw_netw_id_l;  /* NWID low order byte */
unsigned char mmw_netw_id_h;  /* NWID high order byte */
/* Network ID or Domain : create virtual net on the air */

/* 2.0 Hardware extension - frequency selection support */
unsigned char mmw_mode_select; /* for analog tests (set to 0) */
unsigned char mmw_unused3[1];  /* unused */
unsigned char mmw_fee_ctrl;  /* frequency eeprom control */
#define MMW_FEE_CTRL_PRE 0x10 /* Enable protected instructions */
#define MMW_FEE_CTRL_DWLD 0x08 /* Download eeprom to mmc */
#define MMW_FEE_CTRL_CMD 0x07 /* EEprom commands : */
#define MMW_FEE_CTRL_READ 0x06 /* Read */
#define MMW_FEE_CTRL_WREN 0x04 /* Write enable */
#define MMW_FEE_CTRL_WRITE 0x05 /* Write data to address */
#define MMW_FEE_CTRL_WRALL 0x04 /* Write data to all addresses */
#define MMW_FEE_CTRL_WDS 0x04 /* Write disable */
#define MMW_FEE_CTRL_PRREAD 0x16 /* Read
 addr from protect register */
#define MMW_FEE_CTRL_PREN 0x14 /* Protect register enable */
#define MMW_FEE_CTRL_PRCLEAR 0x17 /* Unprotect all registers */
#define MMW_FEE_CTRL_PRWRITE 0x15 /* Write addr in protect register */
#define MMW_FEE_CTRL_PRDS 0x14 /* Protect register disable */
 /* Never issue this command (PRDS) : it's irreversible !!! */

unsigned char mmw_fee_addr;  /* EEprom address */
#define MMW_FEE_ADDR_CHANNEL 0xF0 /* Select the channel */
#define MMW_FEE_ADDR_OFFSET 0x0F /* Offset in channel data */
#define MMW_FEE_ADDR_EN  0xC0 /* FEE_CTRL enable operations */
#define MMW_FEE_ADDR_DS  0x00 /* FEE_CTRL disable operations */
```

```
#define MMW_FEE_ADDR_ALL 0x40 /* FEE_CTRL all operations */
#define MMW_FEE_ADDR_CLEAR 0xFF /* FEE_CTRL clear operations */

 unsigned char mmw_fee_data_l;  /* Write data to EEprom */
 unsigned char mmw_fee_data_h;  /* high octet */
 unsigned char mmw_ext_ant;  /* Setting for external antenna */
#define MMW_EXT_ANT_EXTANT 0x01 /* Select external
 antenna */
#define MMW_EXT_ANT_POL  0x02 /* Polarity of the antenna */
#define MMW_EXT_ANT_INTERNAL 0x00 /* Internal antenna */
#define MMW_EXT_ANT_EXTERNAL 0x03 /* External antenna */
#define MMW_EXT_ANT_IQ_TEST 0x1C /* IQ test pattern (set to 0) */
};

/* Size for structure checking (if padding is correct) */
#define MMW_SIZE 37

/* Calculate offset of a field in the above structure */
#define mmwoff(p,f)  (unsigned short)((void *)(&((mmw_t *)((void *)0 + (p)))->f) - (void *)0)


/*
* Modem Management Controller (MMC) read structure.
*/
typedef struct mmr_t mmr_t;
struct mmr_t
{
 unsigned char mmr_unused0[8];  /* unused */
 unsigned char mmr_des_status;  /* encryption status */
 unsigned char mmr_des_avail;  /* encryption available (0x55 read) */
#define MMR_DES_AVAIL_DES 0x55  /* DES available */
#define MMR_DES_AVAIL_AES 0x33  /* AES (AT&T) available */
 unsigned char mmr_des_io_invert; /* des I/O invert register */
 unsigned char mmr_unused1[5];  /* unused */
 unsigned
 char mmr_dce_status;  /* DCE status */
#define MMR_DCE_STATUS_RX_BUSY  0x01 /* receiver busy */
#define MMR_DCE_STATUS_LOOPT_IND 0x02 /* loop test indicated */
#define MMR_DCE_STATUS_TX_BUSY  0x04 /* transmitter on */
#define MMR_DCE_STATUS_JBR_EXPIRED 0x08 /* jabber timer expired */
#define MMR_DCE_STATUS   0x0F /* mask to get the bits */
 unsigned char mmr_dsp_id;  /* DSP id (AA = Daedalus rev A) */
 unsigned char mmr_unused2[2];  /* unused */
 unsigned char mmr_correct_nwid_l; /* # of correct NWID's rxd (low) */
 unsigned char mmr_correct_nwid_h; /* # of correct NWID's rxd (high) */
 /* Warning : Read high order octet first !!! */
 unsigned char mmr_wrong_nwid_l; /* # of wrong NWID's rxd (low) */
 unsigned char mmr_wrong_nwid_h; /* # of wrong NWID's rxd (high) */
```

```
 unsigned char mmr_thr_pre_set; /* level threshold preset */
#define MMR_THR_PRE_SET  0x3F  /* level threshold preset */
#define MMR_THR_PRE_SET_CUR 0x80  /* Current signal above it */
 unsigned char mmr_signal_lvl;  /*
 signal level */
#define MMR_SIGNAL_LVL  0x3F  /* signal level */
#define MMR_SIGNAL_LVL_VALID 0x80  /* Updated since last read */
 unsigned char mmr_silence_lvl; /* silence level (noise) */
#define MMR_SILENCE_LVL  0x3F  /* silence level */
#define MMR_SILENCE_LVL_VALID 0x80  /* Updated since last read */
 unsigned char mmr_sgnl_qual;  /* signal quality */
#define MMR_SGNL_QUAL  0x0F  /* signal quality */
#define MMR_SGNL_QUAL_ANT 0x80  /* current antenna used */
 unsigned char mmr_netw_id_l;  /* NWID low order byte ??? */
 unsigned char mmr_unused3[3];  /* unused */

 /* 2.0 Hardware extension - frequency selection support */
 unsigned char mmr_fee_status;  /* Status of frequency eeprom */
#define MMR_FEE_STATUS_ID 0xF0  /* Modem revision id */
#define MMR_FEE_STATUS_DWLD 0x08  /* Download in progress */
#define MMR_FEE_STATUS_BUSY 0x04  /* EEprom busy */
 unsigned char mmr_unused4[1];  /* unused */
 unsigned char mmr_fee_data_l;  /* Read data from eeprom (low) */

 unsigned char mmr_fee_data_h;  /* Read data from eeprom (high) */
};

/* Size for structure checking (if padding is correct) */
#define MMR_SIZE 36

/* Calculate offset of a field in the above structure */
#define mmroff(p,f)  (unsigned short)((void *)(&((mmr_t *)((void *)0 + (p)))->f) - (void *)0)


/* Make the two above structures one */
typedef union mm_t
{
 struct mmw_t w; /* Write to the mmc */
 struct mmr_t r; /* Read from the mmc */
} mm_t;

#endif /* _WAVELAN_CS_H */
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/wavelan_cs.h.svn-base
No license file was found, but licenses were detected in source scan.

```
/*
* Host AP (software wireless LAN access point) driver for
* Intersil Prism2/2.5/3.
*
* Copyright (c) 2001-2002, SSH Communications Security Corp and Jouni Malinen
* <jkmaline@cc.hut.fi>
* Copyright (c) 2002-2005, Jouni Malinen <jkmaline@cc.hut.fi>
*
* 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. See README and COPYING for
* more details.
*
* FIX:
* - there is currently no way of associating TX packets to correct wds device
*   when TX Exc/OK event occurs, so all tx_packets and some
*   tx_errors/tx_dropped are added to the main netdevice; using sw_support
*   field in txdesc might be used to fix this (using Alloc event to increment
*   tx_packets would need some further info in txfid table)
*
* Buffer Access Path (BAP) usage:
*   Prism2 cards have two separate
 BAPs for accessing the card memory. These
*   should allow concurrent access to two different frames and the driver
*   previously used BAP0 for sending data and BAP1 for receiving data.
*   However, there seems to be number of issues with concurrent access and at
*   least one know hardware bug in using BAP0 and BAP1 concurrently with PCI
*   Prism2.5. Therefore, the driver now only uses BAP0 for moving data between
*   host and card memories. BAP0 accesses are protected with local->baplock
*   (spin_lock_bh) to prevent concurrent use.
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/.svn/text-base/hostap_hw.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/hostap_hw.c
No license file was found, but licenses were detected in source scan.

```
/*** -*- linux-c -*- ************************************************************

    Driver for Atmel at76c502 at76c504 and at76c506 wireless cards.

      Copyright 2000-2001 ATMEL Corporation.
      Copyright 2003-2004 Simon Kelley.

   This code was developed from version 2.1.1 of the Atmel drivers,
   released by Atmel corp. under the GPL in December 2002. It also
   includes code from the Linux aironet drivers (C) Benjamin Reed,
```

and the Linux PCMCIA package, (C) David Hinds and the Linux wireless
extensions, (C) Jean Tourrilhes.

The firmware module for reading the MAC address of the card comes from
net.russotto.AtmelMACFW, written by Matthew T. Russotto and copyright
by him. net.russotto.AtmelMACFW is used under the GPL license version 2.
This file contains the module in binary form and, under the terms
of the GPL, in source form. The source is located at the end of the file.

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 software 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 Atmel wireless lan drivers; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

For all queries about this code, please contact the current author,
Simon Kelley <simon@thekelleys.org.uk> and not Atmel Corporation.

Credit is due to HP UK and Cambridge Online Systems Ltd for supplying
hardware used during development of this
driver.

**********/
AtmelMACFW 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.
AtmelMACFW is distributed in the hope that it will be useful,

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/atmel.c
No license file was found, but licenses were detected in source scan.

/* zd_netdev.c
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_netdev.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_netdev.c
No license file was found, but licenses were detected in source scan.

and Cisco proprietary API, so both the Linux Wireless Tools and the
and Cisco proprietary API, so both the Linux Wireless Tools and the

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/Kconfig
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/Kconfig.svn-base
No license file was found, but licenses were detected in source scan.

/*
* Driver for ZyDAS zd1201 based wireless USB devices.
*
* Copyright (c) 2004, 2005 Jeroen Vreeken (pe1rxq@amsat.org)
*
* 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.
*
* Parts of this driver have been derived from a wlan-ng version
* modified by ZyDAS. They also made documentation available, thanks!
* Copyright (C) 1999 AbsoluteValue Systems, Inc. All Rights Reserved.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/zd1201.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1201.c
No license file was found, but licenses were detected in source scan.

/*

Broadcom BCM43xx wireless driver

Copyright (c) 2005 Martin Langer <martin-langer@gmx.de>,
Stefano Brivio <st3@riseup.net>
Michael Buesch <mbuesch@freenet.de>
Danny van Dyk <kugelfang@gentoo.org>
Andreas Jaggi <andreas.jaggi@waterwave.ch>

Some parts of the code in this file are derived from the ipw2200
driver Copyright(c) 2003 - 2004 Intel Corporation.

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; see the file COPYING. If not, write to
the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_power.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_radio.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_phy.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_main.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_wx.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_phy.h.svn-base
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_radio.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_main.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_wx.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_radio.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_wx.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-

base/bcm43xx_power.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_wx.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_phy.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_radio.h
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_main.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_main.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_power.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_phy.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_power.c
No license file was found, but licenses were detected in source scan.

/* By the way, for the copyright and legal stuff:
* almost everybody wrote code under the GNU or BSD license (or similar),
* and want their original copyright to remain somewhere in the
* code (for myself, I go with the GPL).
* Nobody wants to take responsibility for anything, except the fame.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wavelan.p.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/wavelan.p.h.svn-base
No license file was found, but licenses were detected in source scan.

/* zd_rf_al2230.c: Functions for the AL2230 RF controller
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_rf_al2230.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-

base/zd_rf_al2230.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/* orinoco_pci.c
*
* Driver for Prism 2.5/3 devices that have a direct PCI interface
* (i.e. these are not PCMCIA cards in a PCMCIA-to-PCI bridge).
* The card contains only one PCI region, which contains all the usual
* hermes registers, as well as the COR register.
*
* Current maintainers are:
*  Pavel Roskin <proski AT gnu.org>
* and David Gibson <hermes AT gibson.dropbear.id.au>
*
* Some of this code is borrowed from orinoco_plx.c
* Copyright (C) 2001 Daniel Barlow <dan AT telent.net>
* Some of this code is "inspired" by linux-wlan-ng-0.1.10, but nothing
* has been copied from it. linux-wlan-ng-0.1.10 is originally :
* Copyright (C) 1999 AbsoluteValue Systems, Inc.  All Rights Reserved.
* This file originally written by:
* Copyright (C) 2001 Jean Tourrilhes <jt AT hpl.hp.com>
* And is now maintained by:
* (C) Copyright David Gibson, IBM Corp. 2002-2003.
*
* The contents of this file are subject
 to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above.  If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and
* other provisions required by the GPL.  If you do not delete the
* provisions
 above, a recipient may use your version of this file
* under either the MPL or the GPL.
*/
```

```
#define DRIVER_NAME "orinoco_pci"
#define PFX DRIVER_NAME ": "

#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/init.h>
#include <linux/delay.h>
#include <linux/pci.h>

#include "orinoco.h"
#include "orinoco_pci.h"

/* Offset of the COR register of the PCI card */
#define HERMES_PCI_COR  (0x26)

/* Bitmask to reset the card */
#define HERMES_PCI_COR_MASK (0x0080)

/* Magic timeouts for doing the reset.
* Those times are straight from wlan-ng, and it is claimed that they
* are necessary. Alan will kill me. Take your time and grab a coffee. */
#define HERMES_PCI_COR_ONT (250)  /* ms */
#define HERMES_PCI_COR_OFFT (500)  /* ms */
#define HERMES_PCI_COR_BUSYT (500)  /* ms */

/*
* Do a soft reset of the card using the Configuration Option Register
* We need this to get going...
* This is the part of the code that is strongly inspired from wlan-ng
*
* Note : This
 code is done with irq enabled. This mean that many
* interrupts will occur while we are there. This is why we use the
* jiffies to regulate time instead of a straight mdelay(). Usually we
* need only around 245 iteration of the loop to do 250 ms delay.
*
* Note bis : Don't try to access HERMES_CMD during the reset phase.
* It just won't work !
*/
static int orinoco_pci_cor_reset(struct orinoco_private *priv)
{
 hermes_t *hw = &priv->hw;
 unsigned long timeout;
 u16 reg;

 /* Assert the reset until the card notices */
 hermes_write_regn(hw, PCI_COR, HERMES_PCI_COR_MASK);
 mdelay(HERMES_PCI_COR_ONT);
```

```
 /* Give time for the card to recover from this hard effort */
 hermes_write_regn(hw, PCI_COR, 0x0000);
 mdelay(HERMES_PCI_COR_OFFT);

 /* The card is ready when it's no longer busy */
 timeout = jiffies + (HERMES_PCI_COR_BUSYT * HZ / 1000);
 reg = hermes_read_regn(hw, CMD);
 while (time_before(jiffies, timeout) && (reg & HERMES_CMD_BUSY)) {
  mdelay(1);
  reg = hermes_read_regn(hw,
 CMD);
 }

 /* Still busy? */
 if (reg & HERMES_CMD_BUSY) {
  printk(KERN_ERR PFX "Busy timeout\n");
  return -ETIMEDOUT;
 }

 return 0;
}

static int orinoco_pci_init_one(struct pci_dev *pdev,
    const struct pci_device_id *ent)
{
 int err;
 struct orinoco_private *priv;
 struct orinoco_pci_card *card;
 struct net_device *dev;
 void __iomem *hermes_io;

 err = pci_enable_device(pdev);
 if (err) {
  printk(KERN_ERR PFX "Cannot enable PCI device\n");
  return err;
 }

 err = pci_request_regions(pdev, DRIVER_NAME);
 if (err) {
  printk(KERN_ERR PFX "Cannot obtain PCI resources\n");
  goto fail_resources;
 }

 hermes_io = pci_iomap(pdev, 0, 0);
 if (!hermes_io) {
  printk(KERN_ERR PFX "Cannot remap chipset registers\n");
  err = -EIO;
```

```
 goto fail_map_hermes;
}

/* Allocate network device */
dev = alloc_orinocodev(sizeof(*card), orinoco_pci_cor_reset);
if (!dev) {
 printk(KERN_ERR PFX "Cannot allocate network device\n");
 err = -ENOMEM;
 goto fail_alloc;
}

priv = netdev_priv(dev);
card
= priv->card;
SET_MODULE_OWNER(dev);
SET_NETDEV_DEV(dev, &pdev->dev);

hermes_struct_init(&priv->hw, hermes_io, HERMES_32BIT_REGSPACING);

err = request_irq(pdev->irq, orinoco_interrupt, IRQF_SHARED,
    dev->name, dev);
if (err) {
 printk(KERN_ERR PFX "Cannot allocate IRQ %d\n", pdev->irq);
 err = -EBUSY;
 goto fail_irq;
}

err = orinoco_pci_cor_reset(priv);
if (err) {
 printk(KERN_ERR PFX "Initial reset failed\n");
 goto fail;
}

err = register_netdev(dev);
if (err) {
 printk(KERN_ERR PFX "Cannot register network device\n");
 goto fail;
}

pci_set_drvdata(pdev, dev);
printk(KERN_DEBUG "%s: " DRIVER_NAME " at %s\n", dev->name,
       pci_name(pdev));

return 0;

fail:
 free_irq(pdev->irq, dev);
```

```
fail_irq:
 pci_set_drvdata(pdev, NULL);
 free_orinocodev(dev);

fail_alloc:
 pci_iounmap(pdev, hermes_io);

fail_map_hermes:
 pci_release_regions(pdev);

fail_resources:
 pci_disable_device(pdev);

 return err;
}

static void __devexit orinoco_pci_remove_one(struct
 pci_dev *pdev)
{
 struct net_device *dev = pci_get_drvdata(pdev);
 struct orinoco_private *priv = netdev_priv(dev);

 unregister_netdev(dev);
 free_irq(pdev->irq, dev);
 pci_set_drvdata(pdev, NULL);
 free_orinocodev(dev);
 pci_iounmap(pdev, priv->hw.iobase);
 pci_release_regions(pdev);
 pci_disable_device(pdev);
}

static struct pci_device_id orinoco_pci_id_table[] = {
 /* Intersil Prism 3 */
 {0x1260, 0x3872, PCI_ANY_ID, PCI_ANY_ID,},
 /* Intersil Prism 2.5 */
 {0x1260, 0x3873, PCI_ANY_ID, PCI_ANY_ID,},
 /* Samsung MagicLAN SWL-2210P */
 {0x167d, 0xa000, PCI_ANY_ID, PCI_ANY_ID,},
 {0,},
};

MODULE_DEVICE_TABLE(pci, orinoco_pci_id_table);

static struct pci_driver orinoco_pci_driver = {
 .name  = DRIVER_NAME,
 .id_table = orinoco_pci_id_table,
 .probe  = orinoco_pci_init_one,
 .remove  = __devexit_p(orinoco_pci_remove_one),
```

```
 .suspend = orinoco_pci_suspend,
 .resume  = orinoco_pci_resume,
};

static char version[] __initdata = DRIVER_NAME " " DRIVER_VERSION
 " (Pavel Roskin <proski@gnu.org>,"
 "
 David Gibson <hermes@gibson.dropbear.id.au> &"
 " Jean Tourrilhes <jt@hpl.hp.com>)";
MODULE_AUTHOR("Pavel Roskin <proski@gnu.org> & David Gibson <hermes@gibson.dropbear.id.au>");
MODULE_DESCRIPTION("Driver for wireless LAN cards using direct PCI interface");
MODULE_LICENSE("Dual MPL/GPL");

static int __init orinoco_pci_init(void)
{
 printk(KERN_DEBUG "%s\n", version);
 return pci_register_driver(&orinoco_pci_driver);
}

static void __exit orinoco_pci_exit(void)
{
 pci_unregister_driver(&orinoco_pci_driver);
}

module_init(orinoco_pci_init);
module_exit(orinoco_pci_exit);

/*
* Local variables:
*  c-indent-level: 8
*  c-basic-offset: 8
*  tab-width: 8
* End:
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/orinoco_pci.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/* By the way : for the copyright & legal stuff :
* Almost everybody wrote code under GNU or BSD license (or alike),
* and want that their original copyright remain somewhere in the
* code (for myself, I go with the GPL).
* Nobody want to take responsibility for anything, except the fame...
*/
```

Found in path(s):

* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/wavelan_cs.p.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wavelan_cs.p.h
No license file was found, but licenses were detected in source scan.

```
/*============================================================================
*
* A  PCMCIA client driver for the Raylink wireless LAN card.
* The starting point for this module was the skeleton.c in the
* PCMCIA 2.9.12 package written by David Hinds, dahinds@users.sourceforge.net
*
*
* Copyright (c) 1998  Corey Thomas (corey@world.std.com)
*
* This driver is free software; you can redistribute it and/or modify
* it under the terms of version 2 only of the GNU General Public License as
* published by the Free Software Foundation.
*
* It 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
*
* Changes:
* Arnaldo Carvalho de Melo <acme@conectiva.com.br> - 08/08/2000
* - reorganize kmallocs in ray_attach, checking all for failure
*   and releasing the previous allocations if one fails
*
* Daniele Bellucci <bellucda@tiscali.it> - 07/10/2003
* - Audit copy_to_user in ioctl(SIOCGIWESSID)
*
============================================================================*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/ray_cs.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/ray_cs.c
No license file was found, but licenses were detected in source scan.

```
/* zd_rf_al7230b.c: Functions for the AL7230B RF controller
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_rf_al7230b.c.svn-base
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_rf_al7230b.c
No license file was found, but licenses were detected in source scan.

/* zd_ieee80211.c
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_ieee80211.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_ieee80211.c
No license file was found, but licenses were detected in source scan.

MODULE_LICENSE("GPL");
* according to the terms of the GNU General Public License.

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-

base/wavelan.c.svn-base
No license file was found, but licenses were detected in source scan.

/* zd_rf.c
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_rf.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_rf.c
No license file was found, but licenses were detected in source scan.

/*
*
* Copyright (C) 2002 Intersil Americas Inc.
*   (C) 2003 Aurelien Alleaume <slts@free.fr>
*   (C) 2003 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
*
* 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
*
* 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
*
*/

Found in path(s):

* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/isl_ioctl.h.svn-base
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/isl_ioctl.h
No license file was found, but licenses were detected in source scan.

```
/* zd_util.c
*
* 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
*
* Utility program
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_util.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_util.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/*
*  Copyright (C) 2003 Aurelien Alleaume <slts@free.fr>
*
*  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
*
*  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
*
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/oid_mgt.h.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/oid_mgt.h
No license file was found, but licenses were detected in source scan.

MODULE_LICENSE("GPL");

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/hostap_plx.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/.svn/text-base/hostap_cs.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/.svn/text-base/hostap_plx.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/hostap_pci.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/.svn/text-base/hostap_pci.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/wl3501_cs.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/netwave_cs.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/wl3501_cs.c.svn-base
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/netwave_cs.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/hostap/hostap_cs.c
No license file was found, but licenses were detected in source scan.

/* zd_types.h
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_types.h.svn-base

* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_types.h
No license file was found, but licenses were detected in source scan.

```
/*
*
*
*  Copyright (C) 2003 Herbert Valerio Riedel <hvr@gnu.org>
*  Copyright (C) 2004 Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
*  Copyright (C) 2004 Aurelien Alleaume <slts@free.fr>
*
*  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
*
*  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
*
*/
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/isl_oid.h.svn-base
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/isl_oid.h
No license file was found, but licenses were detected in source scan.

```
/*

 Broadcom BCM43xx wireless driver

 debugfs driver debugging code

 Copyright (c) 2005 Michael Buesch <mbuesch@freenet.de>

 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; see the file COPYING. If not, write to
the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_debugfs.c
*
/opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_debugfs.c.svn-base
No license file was found, but licenses were detected in source scan.

/* zd_rf_rfmd.c: Functions for the RFMD RF controller
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_rf_rf2959.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_rf_rf2959.c
No license file was found, but licenses were detected in source scan.

/*

Broadcom BCM43xx wireless driver

DMA ringbuffer and descriptor allocation/management

Copyright (c) 2005, 2006 Michael Buesch <mbuesch@freenet.de>

Some code in this file is derived from the b44.c driver
Copyright (C) 2002 David S. Miller
Copyright (C) Pekka Pietikainen

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; see the file COPYING. If not, write to
the Free Software
Foundation, Inc., 51 Franklin Steet, Fifth Floor,
Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_dma.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_dma.c.svn-base
No license file was found, but licenses were detected in source scan.

/*
* Copyright (c) 2004, 2005 Jeroen Vreeken (pe1rxq@amsat.org)
*
* 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.
*
* Parts of this driver have been derived from a wlan-ng version
* modified by ZyDAS.
* Copyright (C) 1999 AbsoluteValue Systems, Inc. All Rights Reserved.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1201.h
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/zd1201.h.svn-base
No license file was found, but licenses were detected in source scan.

/*
* Copyright 1996 The Board of Trustees of The Leland Stanford

```
* Junior University. All Rights Reserved.
*
* Permission to use, copy, modify, and distribute this
* software and its documentation for any purpose and without
* fee is hereby granted, provided that the above copyright
* notice appear in all copies.  Stanford University
* makes no representations about the suitability of this
* software for any purpose.  It is provided "as is" without
* express or implied warranty.
*
* strip.c This module implements Starmode Radio IP (STRIP)
*  for kernel-based devices like TTY.  It interfaces between a
*  raw TTY, and the kernel's INET protocol layers (via DDI).
*
* Version: @(#)strip.c 1.3 July 1997
*
* Author: Stuart Cheshire <cheshire@cs.stanford.edu>
*
* Fixes: v0.9 12th Feb 1996 (SC)
*  New byte stuffing (2+6 run-length encoding)
*  New watchdog timer task
*  New Protocol key (SIP0)
*
*  v0.9.1
 3rd March 1996 (SC)
*  Changed to dynamic device allocation -- no more compile
*  time (or boot time) limit on the number of STRIP devices.
*
*  v0.9.2 13th March 1996 (SC)
*  Uses arp cache lookups (but doesn't send arp packets yet)
*
*  v0.9.3 17th April 1996 (SC)
*  Fixed bug where STR_ERROR flag was getting set unneccessarily
*  (causing otherwise good packets to be unneccessarily dropped)
*
*  v0.9.4 27th April 1996 (SC)
*  First attempt at using "&COMMAND" Starmode AT commands
*
*  v0.9.5 29th May 1996 (SC)
*  First attempt at sending (unicast) ARP packets
*
*  v0.9.6 5th June 1996 (Elliot)
*  Put "message level" tags in every "printk" statement
*
*  v0.9.7 13th June 1996 (laik)
*  Added support for the /proc fs
*
*            v0.9.8 July 1996 (Mema)
```

```
*           Added packet logging
*
*           v1.0 November 1996 (SC)
*           Fixed (severe) memory leaks in the /proc fs code
*           Fixed race conditions in the
 logging code
*
*           v1.1 January 1997 (SC)
*           Deleted packet logging (use tcpdump instead)
*           Added support for Metricom Firmware v204 features
*           (like message checksums)
*
*           v1.2 January 1997 (SC)
*           Put portables list back in
*
*           v1.3 July 1997 (SC)
*           Made STRIP driver set the radio's baud rate automatically.
*           It is no longer necessarily to manually set the radio's
*           rate permanently to 115200 -- the driver handles setting
*           the rate automatically.
*/

#ifdef MODULE
static const char StripVersion[] = "1.3A-STUART.CHESHIRE-MODULAR";
#else
static const char StripVersion[] = "1.3A-STUART.CHESHIRE";
#endif

#define TICKLE_TIMERS 0
#define EXT_COUNTERS 1


/************************************************************************/
/* Header files       */

#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/init.h>
#include
 <linux/bitops.h>
#include <asm/system.h>
#include <asm/uaccess.h>

# include <linux/ctype.h>
#include <linux/string.h>
#include <linux/mm.h>
#include <linux/interrupt.h>
#include <linux/in.h>
```

```
#include <linux/tty.h>
#include <linux/errno.h>
#include <linux/netdevice.h>
#include <linux/inetdevice.h>
#include <linux/etherdevice.h>
#include <linux/skbuff.h>
#include <linux/if_arp.h>
#include <linux/if_strip.h>
#include <linux/proc_fs.h>
#include <linux/seq_file.h>
#include <linux/serial.h>
#include <linux/serialP.h>
#include <linux/rcupdate.h>
#include <net/arp.h>

#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/time.h>
#include <linux/jiffies.h>

/************************************************************************/
/* Useful structures and definitions     */

/*
* A MetricomKey identifies the protocol being carried inside a Metricom
* Starmode packet.
*/

typedef union {
 __u8 c[4];
 __u32 l;
} MetricomKey;

/*
* An IP address can be viewed as four bytes in memory
 (which is what it is) or as
* a single 32-bit long (which is convenient for assignment, equality testing etc.)
*/

typedef union {
 __u8 b[4];
 __u32 l;
} IPaddr;

/*
* A MetricomAddressString is used to hold a printable representation of
* a Metricom address.
*/
```

```
typedef struct {
 __u8 c[24];
} MetricomAddressString;

/* Encapsulation can expand packet of size x to 65/64x + 1
* Sent packet looks like "<CR>*<address>*<key><encaps payload><CR>"
*                       1 1   1-18  1  4        ?        1
* eg.                   <CR>*0000-1234*SIP0<encaps payload><CR>
* We allow 31 bytes for the stars, the key, the address and the <CR>s
*/
#define STRIP_ENCAP_SIZE(X) (32 + (X)*65L/64L)

/*
* A STRIP_Header is never really sent over the radio, but making a dummy
* header for internal use within the kernel that looks like an Ethernet
* header makes certain other software happier. For example, tcpdump
* already understands Ethernet headers.
*/

typedef struct {
 MetricomAddress
 dst_addr; /* Destination address, e.g. "0000-1234"   */
 MetricomAddress src_addr; /* Source address, e.g. "0000-5678"        */
 unsigned short protocol; /* The protocol type, using Ethernet codes */
} STRIP_Header;

typedef struct {
 char c[60];
} MetricomNode;

#define NODE_TABLE_SIZE 32
typedef struct {
 struct timeval timestamp;
 int num_nodes;
 MetricomNode node[NODE_TABLE_SIZE];
} MetricomNodeTable;

enum { FALSE = 0, TRUE = 1 };

/*
* Holds the radio's firmware version.
*/
typedef struct {
 char c[50];
} FirmwareVersion;

/*
```

```
* Holds the radio's serial number.
*/
typedef struct {
 char c[18];
} SerialNumber;

/*
* Holds the radio's battery voltage.
*/
typedef struct {
 char c[11];
} BatteryVoltage;

typedef struct {
 char c[8];
} char8;

enum {
 NoStructure = 0, /* Really old firmware */
 StructuredMessages = 1, /* Parsable AT response msgs */
 ChecksummedMessages = 2 /* Parsable AT response msgs with checksums */
};

struct strip {
 int magic;
 /*
  * These are
 pointers to the malloc()ed frame buffers.
  */

 unsigned char *rx_buff; /* buffer for received IP packet */
 unsigned char *sx_buff; /* buffer for received serial data */
 int sx_count;  /* received serial data counter */
 int sx_size;  /* Serial buffer size           */
 unsigned char *tx_buff; /* transmitter buffer           */
 unsigned char *tx_head; /* pointer to next byte to XMIT */
 int tx_left;  /* bytes left in XMIT queue     */
 int tx_size;  /* Serial buffer size           */

 /*
  * STRIP interface statistics.
  */

 unsigned long rx_packets; /* inbound frames counter       */
 unsigned long tx_packets; /* outbound frames counter      */
 unsigned long rx_errors; /* Parity, etc. errors          */
 unsigned long tx_errors; /* Planned stuff                */
 unsigned long rx_dropped; /* No memory for skb           */
```

```
  unsigned long tx_dropped; /* When MTU change             */
  unsigned long rx_over_errors; /* Frame bigger then STRIP buf. */

  unsigned long pps_timer; /*
  Timer to determine pps       */
  unsigned long rx_pps_count; /* Counter to determine pps     */
  unsigned long tx_pps_count; /* Counter to determine pps     */
  unsigned long sx_pps_count; /* Counter to determine pps     */
  unsigned long rx_average_pps; /* rx packets per second * 8    */
  unsigned long tx_average_pps; /* tx packets per second * 8    */
  unsigned long sx_average_pps; /* sent packets per second * 8  */

 #ifdef EXT_COUNTERS
  unsigned long rx_bytes;  /* total received bytes */
  unsigned long tx_bytes;  /* total received bytes */
  unsigned long rx_rbytes; /* bytes thru radio i/f */
  unsigned long tx_rbytes; /* bytes thru radio i/f */
  unsigned long rx_sbytes; /* tot bytes thru serial i/f */
  unsigned long tx_sbytes; /* tot bytes thru serial i/f */
  unsigned long rx_ebytes; /* tot stat/err bytes */
  unsigned long tx_ebytes; /* tot stat/err bytes */
 #endif

  /*
   * Internal variables.
   */

  struct list_head  list;  /* Linked list of devices */

  int discard;   /* Set if
  serial error          */
  int working;   /* Is radio working correctly?  */
  int firmware_level;  /* Message structuring level    */
  int next_command;  /* Next periodic command        */
  unsigned int user_baud;  /* The user-selected baud rate  */
  int mtu;   /* Our mtu (to spot changes!)   */
  long watchdog_doprobe;  /* Next time to test the radio  */
  long watchdog_doreset;  /* Time to do next reset        */
  long gratuitous_arp;  /* Time to send next ARP refresh */
  long arp_interval;  /* Next ARP interval            */
  struct timer_list idle_timer; /* For periodic wakeup calls    */
  MetricomAddress true_dev_addr; /* True address of radio        */
  int manual_dev_addr;  /* Hack: See note below         */

  FirmwareVersion firmware_version; /* The radio's firmware version */
  SerialNumber serial_number; /* The radio's serial number    */
  BatteryVoltage battery_voltage; /* The radio's battery voltage  */
```

```
/*
 * Other useful structures.
 */

struct tty_struct *tty;  /* ptr to
TTY structure        */
struct net_device *dev;  /* Our device structure         */

/*
 * Neighbour radio records
 */

MetricomNodeTable portables;
MetricomNodeTable poletops;
};

/*
* Note: manual_dev_addr hack
*
* It is not possible to change the hardware address of a Metricom radio,
* or to send packets with a user-specified hardware source address, thus
* trying to manually set a hardware source address is a questionable
* thing to do.  However, if the user *does* manually set the hardware
* source address of a STRIP interface, then the kernel will believe it,
* and use it in certain places. For example, the hardware address listed
* by ifconfig will be the manual address, not the true one.
* (Both addresses are listed in /proc/net/strip.)
* Also, ARP packets will be sent out giving the user-specified address as
* the source address, not the real address. This is dangerous, because
* it means you won't receive any replies -- the ARP replies will go to
* the
 specified address, which will be some other radio. The case where
* this is useful is when that other radio is also connected to the same
* machine. This allows you to connect a pair of radios to one machine,
* and to use one exclusively for inbound traffic, and the other
* exclusively for outbound traffic. Pretty neat, huh?
*
* Here's the full procedure to set this up:
*
* 1. "slattach" two interfaces, e.g. st0 for outgoing packets,
*    and st1 for incoming packets
*
* 2. "ifconfig" st0 (outbound radio) to have the hardware address
*    which is the real hardware address of st1 (inbound radio).
*    Now when it sends out packets, it will masquerade as st1, and
*    replies will be sent to that radio, which is exactly what we want.
*
* 3. Set the route table entry ("route add default ..." or
```

```
*    "route add -net ...", as appropriate) to send packets via the st0
*    interface (outbound radio). Do not add any route which sends packets
*    out via the st1 interface
 -- that radio is for inbound traffic only.
*
* 4. "ifconfig" st1 (inbound radio) to have hardware address zero.
*    This tells the STRIP driver to "shut down" that interface and not
*    send any packets through it. In particular, it stops sending the
*    periodic gratuitous ARP packets that a STRIP interface normally sends.
*    Also, when packets arrive on that interface, it will search the
*    interface list to see if there is another interface who's manual
*    hardware address matches its own real address (i.e. st0 in this
*    example) and if so it will transfer ownership of the skbuff to
*    that interface, so that it looks to the kernel as if the packet
*    arrived on that interface. This is necessary because when the
*    kernel sends an ARP packet on st0, it expects to get a reply on
*    st0, and if it sees the reply come from st1 then it will ignore
*    it (to be accurate, it puts the entry in the ARP table, but
*    labelled in such a way that st0
 can't use it).
*
* Thanks to Petros Maniatis for coming up with the idea of splitting
* inbound and outbound traffic between two interfaces, which turned
* out to be really easy to implement, even if it is a bit of a hack.
*
* Having set a manual address on an interface, you can restore it
* to automatic operation (where the address is automatically kept
* consistent with the real address of the radio) by setting a manual
* address of all ones, e.g. "ifconfig st0 hw strip FFFFFFFFFFFF"
* This 'turns off' manual override mode for the device address.
*
* Note: The IEEE 802 headers reported in tcpdump will show the *real*
* radio addresses the packets were sent and received from, so that you
* can see what is really going on with packets, and which interfaces
* they are really going through.
*/


/************************************************************************/
/* Constants       */

/*
* CommandString1 works on all radios
* Other CommandStrings are only used
 with firmware that provides structured responses.
*
* ats319=1 Enables Info message for node additions and deletions
* ats319=2 Enables Info message for a new best node
```

```
* ats319=4 Enables checksums
* ats319=8 Enables ACK messages
*/

static const int MaxCommandStringLength = 32;
static const int CompatibilityCommand = 1;

static const char CommandString0[] = "*&COMMAND*ATS319=7"; /* Turn on checksums & info messages */
static const char CommandString1[] = "*&COMMAND*ATS305?"; /* Query radio name */
static const char CommandString2[] = "*&COMMAND*ATS325?"; /* Query battery voltage */
static const char CommandString3[] = "*&COMMAND*ATS300?"; /* Query version information */
static const char CommandString4[] = "*&COMMAND*ATS311?"; /* Query poletop list */
static const char CommandString5[] = "*&COMMAND*AT~LA";  /* Query portables list */
typedef struct {
 const char *string;
 long length;
} StringDescriptor;

static const StringDescriptor CommandString[] = {
 {CommandString0, sizeof(CommandString0)
 - 1},
 {CommandString1, sizeof(CommandString1) - 1},
 {CommandString2, sizeof(CommandString2) - 1},
 {CommandString3, sizeof(CommandString3) - 1},
 {CommandString4, sizeof(CommandString4) - 1},
 {CommandString5, sizeof(CommandString5) - 1}
};

#define GOT_ALL_RADIO_INFO(S)      \
   ((S)->firmware_version.c[0] && \
    (S)->battery_voltage.c[0]  && \
    memcmp(&(S)->true_dev_addr, zero_address.c, sizeof(zero_address)))

static const char hextable[16] = "0123456789ABCDEF";

static const MetricomAddress zero_address;
static const MetricomAddress broadcast_address =
   { {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF} };

static const MetricomKey SIP0Key = { "SIP0" };
static const MetricomKey ARP0Key = { "ARP0" };
static const MetricomKey ATR_Key = { "ATR " };
static const MetricomKey ACK_Key = { "ACK_" };
static const MetricomKey INF_Key = { "INF_" };
static const MetricomKey ERR_Key = { "ERR_" };

static const long MaxARPInterval = 60 * HZ; /* One minute */
```

```
/*
* Maximum
 Starmode packet length is 1183 bytes. Allowing 4 bytes for
* protocol key, 4 bytes for checksum, one byte for CR, and 65/64 expansion
* for STRIP encoding, that translates to a maximum payload MTU of 1155.
* Note: A standard NFS 1K data packet is a total of 0x480 (1152) bytes
* long, including IP header, UDP header, and NFS header. Setting the STRIP
* MTU to 1152 allows us to send default sized NFS packets without fragmentation.
*/
static const unsigned short MAX_SEND_MTU = 1152;
static const unsigned short MAX_RECV_MTU = 1500; /* Hoping for Ethernet sized packets in the future! */
static const unsigned short DEFAULT_STRIP_MTU = 1152;
static const int STRIP_MAGIC = 0x5303;
static const long LongTime = 0x7FFFFFFF;

/************************************************************************/
/* Global variables       */

static LIST_HEAD(strip_list);
static DEFINE_SPINLOCK(strip_lock);

/************************************************************************/
/* Macros       */

/*
 Returns TRUE if text T begins with prefix P */
#define has_prefix(T,L,P) (((L) >= sizeof(P)-1) && !strncmp((T), (P), sizeof(P)-1))

/* Returns TRUE if text T of length L is equal to string S */
#define text_equal(T,L,S) (((L) == sizeof(S)-1) && !strncmp((T), (S), sizeof(S)-1))

#define READHEX(X) ((X)>='0' && (X)<='9' ? (X)-'0' :      \
                (X)>='a' && (X)<='f' ? (X)-'a'+10 :   \
                (X)>='A' && (X)<='F' ? (X)-'A'+10 : 0 )

#define READHEX16(X) ((__u16)(READHEX(X)))

#define READDEC(X) ((X)>='0' && (X)<='9' ? (X)-'0' : 0)

#define ARRAY_END(X) (&((X)[ARRAY_SIZE(X)]))

#define JIFFIE_TO_SEC(X) ((X) / HZ)


/************************************************************************/
/* Utility routines       */

static int arp_query(unsigned char *haddr, u32 paddr,
```

```
      struct net_device *dev)
{
 struct neighbour *neighbor_entry;
 int ret = 0;

 neighbor_entry = neigh_lookup(&arp_tbl, &paddr, dev);

 if (neighbor_entry != NULL) {
  neighbor_entry->used =
 jiffies;
  if (neighbor_entry->nud_state & NUD_VALID) {
   memcpy(haddr, neighbor_entry->ha, dev->addr_len);
   ret = 1;
  }
  neigh_release(neighbor_entry);
 }
 return ret;
}

static void DumpData(char *msg, struct strip *strip_info, __u8 * ptr,
      __u8 * end)
{
 static const int MAX_DumpData = 80;
 __u8 pkt_text[MAX_DumpData], *p = pkt_text;

 *p++ = '\"';

 while (ptr < end && p < &pkt_text[MAX_DumpData - 4]) {
  if (*ptr == '\\') {
   *p++ = '\\';
   *p++ = '\\';
  } else {
   if (*ptr >= 32 && *ptr <= 126) {
    *p++ = *ptr;
   } else {
    sprintf(p, "\\%02X", *ptr);
    p += 3;
   }
  }
  ptr++;
 }

 if (ptr == end)
  *p++ = '\"';
 *p++ = 0;

 printk(KERN_INFO "%s: %-13s%s\n", strip_info->dev->name, msg, pkt_text);
}
```

```
/*************************************************************************/
/* Byte stuffing/unstuffing routines     */

/* Stuffing scheme:
* 00    Unused (reserved character)
* 01-3F Run of 2-64 different characters
* 40-7F Run of 1-64 different characters
 plus a single zero at the end
* 80-BF Run of 1-64 of the same character
* C0-FF Run of 1-64 zeroes (ASCII 0)
*/

typedef enum {
 Stuff_Diff = 0x00,
 Stuff_DiffZero = 0x40,
 Stuff_Same = 0x80,
 Stuff_Zero = 0xC0,
 Stuff_NoCode = 0xFF, /* Special code, meaning no code selected */

 Stuff_CodeMask = 0xC0,
 Stuff_CountMask = 0x3F,
 Stuff_MaxCount = 0x3F,
 Stuff_Magic = 0x0D /* The value we are eliminating */
} StuffingCode;

/* StuffData encodes the data starting at "src" for "length" bytes.
* It writes it to the buffer pointed to by "dst" (which must be at least
* as long as 1 + 65/64 of the input length). The output may be up to 1.6%
* larger than the input for pathological input, but will usually be smaller.
* StuffData returns the new value of the dst pointer as its result.
* "code_ptr_ptr" points to a "__u8 *" which is used to hold encoding state
* between calls, allowing an encoded packet to be incrementally built up
* from small parts. On the first call, the "__u8 *" pointed
 to should be
* initialized to NULL; between subsequent calls the calling routine should
* leave the value alone and simply pass it back unchanged so that the
* encoder can recover its current state.
*/

#define StuffData_FinishBlock(X) \
(*code_ptr = (X) ^ Stuff_Magic, code = Stuff_NoCode)

static __u8 *StuffData(__u8 * src, __u32 length, __u8 * dst,
        __u8 ** code_ptr_ptr)
{
 __u8 *end = src + length;
```

```
__u8 *code_ptr = *code_ptr_ptr;
__u8 code = Stuff_NoCode, count = 0;

if (!length)
 return (dst);

if (code_ptr) {
 /*
  * Recover state from last call, if applicable
  */
 code = (*code_ptr ^ Stuff_Magic) & Stuff_CodeMask;
 count = (*code_ptr ^ Stuff_Magic) & Stuff_CountMask;
}

while (src < end) {
 switch (code) {
  /* Stuff_NoCode: If no current code, select one */
 case Stuff_NoCode:
  /* Record where we're going to put this code */
  code_ptr = dst++;
  count = 0; /* Reset the count (zero means one instance) */
  /* Tentatively start a new block */
  if
(*src == 0) {
   code = Stuff_Zero;
   src++;
  } else {
   code = Stuff_Same;
   *dst++ = *src++ ^ Stuff_Magic;
  }
  /* Note: We optimistically assume run of same -- */
  /* which will be fixed later in Stuff_Same */
  /* if it turns out not to be true. */
  break;

  /* Stuff_Zero: We already have at least one zero encoded */
 case Stuff_Zero:
  /* If another zero, count it, else finish this code block */
  if (*src == 0) {
   count++;
   src++;
  } else {
   StuffData_FinishBlock(Stuff_Zero + count);
  }
  break;

  /* Stuff_Same: We already have at least one byte encoded */
 case Stuff_Same:
```

```
  /* If another one the same, count it */
  if ((*src ^ Stuff_Magic) == code_ptr[1]) {
   count++;
   src++;
   break;
  }
  /* else, this byte does not match this block. */
  /* If we already have two or more bytes encoded, finish this code block */
  if (count) {
   StuffData_FinishBlock(Stuff_Same + count);
   break;
  }
  /* else, we only have one so
far, so switch to Stuff_Diff code */
  code = Stuff_Diff;
  /* and fall through to Stuff_Diff case below
   * Note cunning cleverness here: case Stuff_Diff compares
   * the current character with the previous two to see if it
   * has a run of three the same. Won't this be an error if
   * there aren't two previous characters stored to compare with?
   * No. Because we know the current character is *not* the same
   * as the previous one, the first test below will necessarily
   * fail and the send half of the "if" won't be executed.
   */

  /* Stuff_Diff: We have at least two *different* bytes encoded */
 case Stuff_Diff:
  /* If this is a zero, must encode a Stuff_DiffZero, and begin a new block */
  if (*src == 0) {
   StuffData_FinishBlock(Stuff_DiffZero +
          count);
  }
  /* else, if we have three in a row, it is worth starting a Stuff_Same block */
  else if ((*src ^ Stuff_Magic) == dst[-1]
    && dst[-1] == dst[-2]) {
   /* Back off the last two
characters we encoded */
   code += count - 2;
   /* Note: "Stuff_Diff + 0" is an illegal code */
   if (code == Stuff_Diff + 0) {
    code = Stuff_Same + 0;
   }
   StuffData_FinishBlock(code);
   code_ptr = dst - 2;
   /* dst[-1] already holds the correct value */
   count = 2; /* 2 means three bytes encoded */
   code = Stuff_Same;
  }
```

```
  /* else, another different byte, so add it to the block */
  else {
   *dst++ = *src ^ Stuff_Magic;
   count++;
  }
  src++; /* Consume the byte */
  break;
 }
 if (count == Stuff_MaxCount) {
  StuffData_FinishBlock(code + count);
 }
}
if (code == Stuff_NoCode) {
 *code_ptr_ptr = NULL;
} else {
 *code_ptr_ptr = code_ptr;
 StuffData_FinishBlock(code + count);
}
return (dst);
}

/*
* UnStuffData decodes the data at "src", up to (but not including) "end".
* It writes the decoded data into the buffer pointed to by "dst", up to a
* maximum of "dst_length", and returns the new value of "src" so that a
* follow-on call can
 read more data, continuing from where the first left off.
*
* There are three types of results:
* 1. The source data runs out before extracting "dst_length" bytes:
*    UnStuffData returns NULL to indicate failure.
* 2. The source data produces exactly "dst_length" bytes:
*    UnStuffData returns new_src = end to indicate that all bytes were consumed.
* 3. "dst_length" bytes are extracted, with more remaining.
*    UnStuffData returns new_src < end to indicate that there are more bytes
*    to be read.
*
* Note: The decoding may be destructive, in that it may alter the source
* data in the process of decoding it (this is necessary to allow a follow-on
* call to resume correctly).
*/

static __u8 *UnStuffData(__u8 * src, __u8 * end, __u8 * dst,
    __u32 dst_length)
{
 __u8 *dst_end = dst + dst_length;
 /* Sanity check */
 if (!src || !end || !dst || !dst_length)
```

```
 return (NULL);
while (src < end && dst < dst_end) {
 int count = (*src ^ Stuff_Magic) & Stuff_CountMask;
 switch
((*src ^ Stuff_Magic) & Stuff_CodeMask) {
 case Stuff_Diff:
  if (src + 1 + count >= end)
   return (NULL);
  do {
   *dst++ = *++src ^ Stuff_Magic;
  }
  while (--count >= 0 && dst < dst_end);
  if (count < 0)
   src += 1;
  else {
   if (count == 0)
    *src = Stuff_Same ^ Stuff_Magic;
   else
    *src =
       (Stuff_Diff +
        count) ^ Stuff_Magic;
  }
  break;
 case Stuff_DiffZero:
  if (src + 1 + count >= end)
   return (NULL);
  do {
   *dst++ = *++src ^ Stuff_Magic;
  }
  while (--count >= 0 && dst < dst_end);
  if (count < 0)
   *src = Stuff_Zero ^ Stuff_Magic;
  else
   *src =
      (Stuff_DiffZero + count) ^ Stuff_Magic;
  break;
 case Stuff_Same:
  if (src + 1 >= end)
   return (NULL);
  do {
   *dst++ = src[1] ^ Stuff_Magic;
  }
  while (--count >= 0 && dst < dst_end);
  if (count < 0)
   src += 2;
  else
   *src = (Stuff_Same + count) ^ Stuff_Magic;
  break;
```

```
 case Stuff_Zero:
  do {
   *dst++ = 0;
  }
  while
(--count >= 0 && dst < dst_end);
  if (count < 0)
   src += 1;
  else
   *src = (Stuff_Zero + count) ^ Stuff_Magic;
  break;
 }
}
if (dst < dst_end)
 return (NULL);
else
 return (src);
}


/************************************************************************/
/* General routines for STRIP      */

/*
* get_baud returns the current baud rate, as one of the constants defined in
* termbits.h
* If the user has issued a baud rate override using the 'setserial' command
* and the logical current rate is set to 38.4, then the true baud rate
* currently in effect (57.6 or 115.2) is returned.
*/
static unsigned int get_baud(struct tty_struct *tty)
{
 if (!tty || !tty->termios)
  return (0);
 if ((tty->termios->c_cflag & CBAUD) == B38400 && tty->driver_data) {
  struct async_struct *info =
      (struct async_struct *) tty->driver_data;
  if ((info->flags & ASYNC_SPD_MASK) == ASYNC_SPD_HI)
   return (B57600);
  if ((info->flags & ASYNC_SPD_MASK) == ASYNC_SPD_VHI)
   return
(B115200);
 }
 return (tty->termios->c_cflag & CBAUD);
}

/*
* set_baud sets the baud rate to the rate defined by baudcode
```

```
* Note: The rate B38400 should be avoided, because the user may have
* issued a 'setserial' speed override to map that to a different speed.
* We could achieve a true rate of 38400 if we needed to by cancelling
* any user speed override that is in place, but that might annoy the
* user, so it is simplest to just avoid using 38400.
*/
static void set_baud(struct tty_struct *tty, unsigned int baudcode)
{
 struct termios old_termios = *(tty->termios);
 tty->termios->c_cflag &= ~CBAUD; /* Clear the old baud setting */
 tty->termios->c_cflag |= baudcode; /* Set the new baud setting */
 tty->driver->set_termios(tty, &old_termios);
}

/*
* Convert a string to a Metricom Address.
*/

#define IS_RADIO_ADDRESS(p) (                                   \
 isdigit((p)[0]) && isdigit((p)[1]) && isdigit((p)[2]) && isdigit((p)[3]) && \
 (p)[4] == '-' &&
                                        \
 isdigit((p)[5]) && isdigit((p)[6]) && isdigit((p)[7]) && isdigit((p)[8])    )

static int string_to_radio_address(MetricomAddress * addr, __u8 * p)
{
 if (!IS_RADIO_ADDRESS(p))
  return (1);
 addr->c[0] = 0;
 addr->c[1] = 0;
 addr->c[2] = READHEX(p[0]) << 4 | READHEX(p[1]);
 addr->c[3] = READHEX(p[2]) << 4 | READHEX(p[3]);
 addr->c[4] = READHEX(p[5]) << 4 | READHEX(p[6]);
 addr->c[5] = READHEX(p[7]) << 4 | READHEX(p[8]);
 return (0);
}

/*
* Convert a Metricom Address to a string.
*/

static __u8 *radio_address_to_string(const MetricomAddress * addr,
        MetricomAddressString * p)
{
 sprintf(p->c, "%02X%02X-%02X%02X", addr->c[2], addr->c[3],
  addr->c[4], addr->c[5]);
 return (p->c);
}
```

```
/*
* Note: Must make sure sx_size is big enough to receive a stuffed
* MAX_RECV_MTU packet. Additionally, we also want to ensure that it's
* big enough to receive a large radio neighbour list (currently 4K).
*/

static int allocate_buffers(struct
 strip *strip_info, int mtu)
{
 struct net_device *dev = strip_info->dev;
 int sx_size = max_t(int, STRIP_ENCAP_SIZE(MAX_RECV_MTU), 4096);
 int tx_size = STRIP_ENCAP_SIZE(mtu) + MaxCommandStringLength;
 __u8 *r = kmalloc(MAX_RECV_MTU, GFP_ATOMIC);
 __u8 *s = kmalloc(sx_size, GFP_ATOMIC);
 __u8 *t = kmalloc(tx_size, GFP_ATOMIC);
 if (r && s && t) {
  strip_info->rx_buff = r;
  strip_info->sx_buff = s;
  strip_info->tx_buff = t;
  strip_info->sx_size = sx_size;
  strip_info->tx_size = tx_size;
  strip_info->mtu = dev->mtu = mtu;
  return (1);
 }
 kfree(r);
 kfree(s);
 kfree(t);
 return (0);
}

/*
* MTU has been changed by the IP layer.
* We could be in
* an upcall from the tty driver, or in an ip packet queue.
*/
static int strip_change_mtu(struct net_device *dev, int new_mtu)
{
 struct strip *strip_info = netdev_priv(dev);
 int old_mtu = strip_info->mtu;
 unsigned char *orbuff = strip_info->rx_buff;
 unsigned char *osbuff = strip_info->sx_buff;
 unsigned char *otbuff =
 strip_info->tx_buff;

 if (new_mtu > MAX_SEND_MTU) {
  printk(KERN_ERR
         "%s: MTU exceeds maximum allowable (%d), MTU change cancelled.\n",
```

```
        strip_info->dev->name, MAX_SEND_MTU);
 return -EINVAL;
 }

 spin_lock_bh(&strip_lock);
 if (!allocate_buffers(strip_info, new_mtu)) {
  printk(KERN_ERR "%s: unable to grow strip buffers, MTU change cancelled.\n",
        strip_info->dev->name);
  spin_unlock_bh(&strip_lock);
  return -ENOMEM;
 }

 if (strip_info->sx_count) {
  if (strip_info->sx_count <= strip_info->sx_size)
   memcpy(strip_info->sx_buff, osbuff,
         strip_info->sx_count);
  else {
   strip_info->discard = strip_info->sx_count;
   strip_info->rx_over_errors++;
  }
 }

 if (strip_info->tx_left) {
  if (strip_info->tx_left <= strip_info->tx_size)
   memcpy(strip_info->tx_buff, strip_info->tx_head,
         strip_info->tx_left);
  else {
   strip_info->tx_left = 0;
   strip_info->tx_dropped++;
  }
 }
 strip_info->tx_head = strip_info->tx_buff;
 spin_unlock_bh(&strip_lock);

 printk(KERN_NOTICE
 "%s: strip MTU changed fom %d to %d.\n",
        strip_info->dev->name, old_mtu, strip_info->mtu);

 kfree(orbuff);
 kfree(osbuff);
 kfree(otbuff);
 return 0;
}

static void strip_unlock(struct strip *strip_info)
{
 /*
  * Set the timer to go off in one second.
```

```
 */
 strip_info->idle_timer.expires = jiffies + 1 * HZ;
 add_timer(&strip_info->idle_timer);
 netif_wake_queue(strip_info->dev);
}



/*
* If the time is in the near future, time_delta prints the number of
* seconds to go into the buffer and returns the address of the buffer.
* If the time is not in the near future, it returns the address of the
* string "Not scheduled" The buffer must be long enough to contain the
* ascii representation of the number plus 9 charactes for the " seconds"
* and the null character.
*/
#ifdef CONFIG_PROC_FS
static char *time_delta(char buffer[], long time)
{
 time -= jiffies;
 if (time > LongTime / 2)
  return ("Not scheduled");
 if (time < 0)
  time = 0; /*
 Don't print negative times */
 sprintf(buffer, "%ld seconds", time / HZ);
 return (buffer);
}

/* get Nth element of the linked list */
static struct strip *strip_get_idx(loff_t pos)
{
 struct list_head *l;
 int i = 0;

 list_for_each_rcu(l, &strip_list) {
  if (pos == i)
   return list_entry(l, struct strip, list);
  ++i;
 }
 return NULL;
}

static void *strip_seq_start(struct seq_file *seq, loff_t *pos)
{
 rcu_read_lock();
 return *pos ? strip_get_idx(*pos - 1) : SEQ_START_TOKEN;
}
```

```
static void *strip_seq_next(struct seq_file *seq, void *v, loff_t *pos)
{
 struct list_head *l;
 struct strip *s;

 ++*pos;
 if (v == SEQ_START_TOKEN)
  return strip_get_idx(1);

 s = v;
 l = &s->list;
 list_for_each_continue_rcu(l, &strip_list) {
  return list_entry(l, struct strip, list);
 }
 return NULL;
}

static void strip_seq_stop(struct seq_file *seq, void *v)
{
 rcu_read_unlock();
}

static void strip_seq_neighbours(struct seq_file *seq,
      const MetricomNodeTable * table,
      const char
 *title)
{
 /* We wrap this in a do/while loop, so if the table changes */
 /* while we're reading it, we just go around and try again. */
 struct timeval t;

 do {
  int i;
  t = table->timestamp;
  if (table->num_nodes)
   seq_printf(seq, "\n %s\n", title);
  for (i = 0; i < table->num_nodes; i++) {
   MetricomNode node;

   spin_lock_bh(&strip_lock);
   node = table->node[i];
   spin_unlock_bh(&strip_lock);
   seq_printf(seq, "  %s\n", node.c);
  }
 } while (table->timestamp.tv_sec != t.tv_sec
   || table->timestamp.tv_usec != t.tv_usec);
}
```

```
/*
* This function prints radio status information via the seq_file
* interface.  The interface takes care of buffer size and over
* run issues.
*
* The buffer in seq_file is PAGESIZE (4K)
* so this routine should never print more or it will get truncated.
* With the maximum of 32 portables and 32 poletops
* reported, the routine outputs 3107 bytes into the buffer.
*/
static void strip_seq_status_info(struct seq_file *seq,
      const
 struct strip *strip_info)
{
 char temp[32];
 MetricomAddressString addr_string;

 /* First, we must copy all of our data to a safe place, */
 /* in case a serial interrupt comes in and changes it.  */
 int tx_left = strip_info->tx_left;
 unsigned long rx_average_pps = strip_info->rx_average_pps;
 unsigned long tx_average_pps = strip_info->tx_average_pps;
 unsigned long sx_average_pps = strip_info->sx_average_pps;
 int working = strip_info->working;
 int firmware_level = strip_info->firmware_level;
 long watchdog_doprobe = strip_info->watchdog_doprobe;
 long watchdog_doreset = strip_info->watchdog_doreset;
 long gratuitous_arp = strip_info->gratuitous_arp;
 long arp_interval = strip_info->arp_interval;
 FirmwareVersion firmware_version = strip_info->firmware_version;
 SerialNumber serial_number = strip_info->serial_number;
 BatteryVoltage battery_voltage = strip_info->battery_voltage;
 char *if_name = strip_info->dev->name;
 MetricomAddress true_dev_addr = strip_info->true_dev_addr;
 MetricomAddress
 dev_dev_addr =
    *(MetricomAddress *) strip_info->dev->dev_addr;
 int manual_dev_addr = strip_info->manual_dev_addr;
#ifdef EXT_COUNTERS
 unsigned long rx_bytes = strip_info->rx_bytes;
 unsigned long tx_bytes = strip_info->tx_bytes;
 unsigned long rx_rbytes = strip_info->rx_rbytes;
 unsigned long tx_rbytes = strip_info->tx_rbytes;
 unsigned long rx_sbytes = strip_info->rx_sbytes;
 unsigned long tx_sbytes = strip_info->tx_sbytes;
 unsigned long rx_ebytes = strip_info->rx_ebytes;
 unsigned long tx_ebytes = strip_info->tx_ebytes;
```

```
#endif

 seq_printf(seq, "\nInterface name\t\t%s\n", if_name);
 seq_printf(seq, " Radio working:\t\t%s\n", working ? "Yes" : "No");
 radio_address_to_string(&true_dev_addr, &addr_string);
 seq_printf(seq, " Radio address:\t\t%s\n", addr_string.c);
 if (manual_dev_addr) {
  radio_address_to_string(&dev_dev_addr, &addr_string);
  seq_printf(seq, " Device address:\t%s\n", addr_string.c);
 }
 seq_printf(seq, " Firmware version:\t%s", !working ? "Unknown"
 :
       !firmware_level ? "Should be upgraded" :
       firmware_version.c);
 if (firmware_level >= ChecksummedMessages)
  seq_printf(seq, " (Checksums Enabled)");
 seq_printf(seq, "\n");
 seq_printf(seq, " Serial number:\t\t%s\n", serial_number.c);
 seq_printf(seq, " Battery voltage:\t%s\n", battery_voltage.c);
 seq_printf(seq, " Transmit queue (bytes):%d\n", tx_left);
 seq_printf(seq, " Receive packet rate:   %ld packets per second\n",
       rx_average_pps / 8);
 seq_printf(seq, " Transmit packet rate:  %ld packets per second\n",
       tx_average_pps / 8);
 seq_printf(seq, " Sent packet rate:      %ld packets per second\n",
       sx_average_pps / 8);
 seq_printf(seq, " Next watchdog probe:\t%s\n",
       time_delta(temp, watchdog_doprobe));
 seq_printf(seq, " Next watchdog reset:\t%s\n",
       time_delta(temp, watchdog_doreset));
 seq_printf(seq, " Next gratuitous ARP:\t");

 if (!memcmp
     (strip_info->dev->dev_addr, zero_address.c,
      sizeof(zero_address)))
  seq_printf(seq,
 "Disabled\n");
 else {
  seq_printf(seq, "%s\n", time_delta(temp, gratuitous_arp));
  seq_printf(seq, " Next ARP interval:\t%ld seconds\n",
        JIFFIE_TO_SEC(arp_interval));
 }

 if (working) {
#ifdef EXT_COUNTERS
  seq_printf(seq, "\n");
  seq_printf(seq,
        " Total bytes:         \trx:\t%lu\ttx:\t%lu\n",
```

```
        rx_bytes, tx_bytes);
  seq_printf(seq,
        "  thru radio:         \trx:\t%lu\ttx:\t%lu\n",
        rx_rbytes, tx_rbytes);
  seq_printf(seq,
        "  thru serial port:   \trx:\t%lu\ttx:\t%lu\n",
        rx_sbytes, tx_sbytes);
  seq_printf(seq,
        " Total stat/err bytes:\trx:\t%lu\ttx:\t%lu\n",
        rx_ebytes, tx_ebytes);
#endif
  strip_seq_neighbours(seq, &strip_info->poletops,
     "Poletops:");
  strip_seq_neighbours(seq, &strip_info->portables,
     "Portables:");
 }
}

/*
* This function is exports status information from the STRIP driver through
* the /proc file system.
*/
static int strip_seq_show(struct seq_file *seq, void *v)
{
 if (v
 == SEQ_START_TOKEN)
  seq_printf(seq, "strip_version: %s\n", StripVersion);
 else
  strip_seq_status_info(seq, (const struct strip *)v);
 return 0;
}


static struct seq_operations strip_seq_ops = {
 .start = strip_seq_start,
 .next  = strip_seq_next,
 .stop  = strip_seq_stop,
 .show  = strip_seq_show,
};

static int strip_seq_open(struct inode *inode, struct file *file)
{
 return seq_open(file, &strip_seq_ops);
}

static struct file_operations strip_seq_fops = {
 .owner  = THIS_MODULE,
 .open    = strip_seq_open,
```

```
 .read    = seq_read,
 .llseek  = seq_lseek,
 .release = seq_release,
};
#endif



/*****************************************************************************/
/* Sending routines       */

static void ResetRadio(struct strip *strip_info)
{
 struct tty_struct *tty = strip_info->tty;
 static const char init[] = "ate0q1dt**starmode\r**";
 StringDescriptor s = { init, sizeof(init) - 1 };

 /*
  * If the radio isn't working anymore,
  * we should clear the old status information.

 */
 if (strip_info->working) {
  printk(KERN_INFO "%s: No response: Resetting radio.\n",
         strip_info->dev->name);
  strip_info->firmware_version.c[0] = '\0';
  strip_info->serial_number.c[0] = '\0';
  strip_info->battery_voltage.c[0] = '\0';
  strip_info->portables.num_nodes = 0;
  do_gettimeofday(&strip_info->portables.timestamp);
  strip_info->poletops.num_nodes = 0;
  do_gettimeofday(&strip_info->poletops.timestamp);
 }

 strip_info->pps_timer = jiffies;
 strip_info->rx_pps_count = 0;
 strip_info->tx_pps_count = 0;
 strip_info->sx_pps_count = 0;
 strip_info->rx_average_pps = 0;
 strip_info->tx_average_pps = 0;
 strip_info->sx_average_pps = 0;

 /* Mark radio address as unknown */
 *(MetricomAddress *) & strip_info->true_dev_addr = zero_address;
 if (!strip_info->manual_dev_addr)
  *(MetricomAddress *) strip_info->dev->dev_addr =
      zero_address;
 strip_info->working = FALSE;
```

```
 strip_info->firmware_level = NoStructure;
 strip_info->next_command = CompatibilityCommand;
 strip_info->watchdog_doprobe
 = jiffies + 10 * HZ;
 strip_info->watchdog_doreset = jiffies + 1 * HZ;

 /* If the user has selected a baud rate above 38.4 see what magic we have to do */
 if (strip_info->user_baud > B38400) {
  /*
   * Subtle stuff: Pay attention :-)
   * If the serial port is currently at the user's selected (>38.4) rate,
   * then we temporarily switch to 19.2 and issue the ATS304 command
   * to tell the radio to switch to the user's selected rate.
   * If the serial port is not currently at that rate, that means we just
   * issued the ATS304 command last time through, so this time we restore
   * the user's selected rate and issue the normal starmode reset string.
   */
  if (strip_info->user_baud == get_baud(tty)) {
   static const char b0[] = "ate0q1s304=57600\r";
   static const char b1[] = "ate0q1s304=115200\r";
   static const StringDescriptor baudstring[2] =
       { {b0, sizeof(b0) - 1}
   , {b1, sizeof(b1) - 1}
   };
   set_baud(tty, B19200);
   if (strip_info->user_baud
 == B57600)
    s = baudstring[0];
   else if (strip_info->user_baud == B115200)
    s = baudstring[1];
   else
    s = baudstring[1]; /* For now */
  } else
   set_baud(tty, strip_info->user_baud);
 }

 tty->driver->write(tty, s.string, s.length);
#ifdef EXT_COUNTERS
 strip_info->tx_ebytes += s.length;
#endif
}

/*
* Called by the driver when there's room for more data.  If we have
* more packets to send, we send them here.
*/

static void strip_write_some_more(struct tty_struct *tty)
```

```
{
 struct strip *strip_info = (struct strip *) tty->disc_data;

 /* First make sure we're connected. */
 if (!strip_info || strip_info->magic != STRIP_MAGIC ||
     !netif_running(strip_info->dev))
  return;

 if (strip_info->tx_left > 0) {
  int num_written =
      tty->driver->write(tty, strip_info->tx_head,
          strip_info->tx_left);
  strip_info->tx_left -= num_written;
  strip_info->tx_head += num_written;
#ifdef EXT_COUNTERS
  strip_info->tx_sbytes += num_written;
#endif
 } else
 {  /* Else start transmission of another packet */

  tty->flags &= ~(1 << TTY_DO_WRITE_WAKEUP);
  strip_unlock(strip_info);
 }
}

static __u8 *add_checksum(__u8 * buffer, __u8 * end)
{
 __u16 sum = 0;
 __u8 *p = buffer;
 while (p < end)
  sum += *p++;
 end[3] = hextable[sum & 0xF];
 sum >>= 4;
 end[2] = hextable[sum & 0xF];
 sum >>= 4;
 end[1] = hextable[sum & 0xF];
 sum >>= 4;
 end[0] = hextable[sum & 0xF];
 return (end + 4);
}

static unsigned char *strip_make_packet(unsigned char *buffer,
     struct strip *strip_info,
     struct sk_buff *skb)
{
 __u8 *ptr = buffer;
 __u8 *stuffstate = NULL;
 STRIP_Header *header = (STRIP_Header *) skb->data;
```

```
MetricomAddress haddr = header->dst_addr;
int len = skb->len - sizeof(STRIP_Header);
MetricomKey key;

/*HexDump("strip_make_packet", strip_info, skb->data, skb->data + skb->len); */

if (header->protocol == htons(ETH_P_IP))
 key = SIP0Key;
else if (header->protocol == htons(ETH_P_ARP))
 key = ARP0Key;
else {
 printk(KERN_ERR

    "%s: strip_make_packet: Unknown packet type 0x%04X\n",
       strip_info->dev->name, ntohs(header->protocol));
 return (NULL);
}

if (len > strip_info->mtu) {
 printk(KERN_ERR
       "%s: Dropping oversized transmit packet: %d bytes\n",
       strip_info->dev->name, len);
 return (NULL);
}

/*
 * If we're sending to ourselves, discard the packet.
 * (Metricom radios choke if they try to send a packet to their own address.)
 */
if (!memcmp(haddr.c, strip_info->true_dev_addr.c, sizeof(haddr))) {
 printk(KERN_ERR "%s: Dropping packet addressed to self\n",
       strip_info->dev->name);
 return (NULL);
}

/*
 * If this is a broadcast packet, send it to our designated Metricom
 * 'broadcast hub' radio (First byte of address being 0xFF means broadcast)
 */
if (haddr.c[0] == 0xFF) {
 __be32 brd = 0;
 struct in_device *in_dev;

 rcu_read_lock();
 in_dev = __in_dev_get_rcu(strip_info->dev);
 if (in_dev == NULL) {
  rcu_read_unlock();
  return NULL;
```

```
 }
 if
(in_dev->ifa_list)
  brd = in_dev->ifa_list->ifa_broadcast;
 rcu_read_unlock();

 /* arp_query returns 1 if it succeeds in looking up the address, 0 if it fails */
 if (!arp_query(haddr.c, brd, strip_info->dev)) {
  printk(KERN_ERR
         "%s: Unable to send packet (no broadcast hub configured)\n",
         strip_info->dev->name);
  return (NULL);
 }
 /*
  * If we are the broadcast hub, don't bother sending to ourselves.
  * (Metricom radios choke if they try to send a packet to their own address.)
  */
 if (!memcmp
     (haddr.c, strip_info->true_dev_addr.c, sizeof(haddr)))
  return (NULL);
}

*ptr++ = 0x0D;
*ptr++ = '*';
*ptr++ = hextable[haddr.c[2] >> 4];
*ptr++ = hextable[haddr.c[2] & 0xF];
*ptr++ = hextable[haddr.c[3] >> 4];
*ptr++ = hextable[haddr.c[3] & 0xF];
*ptr++ = '-';
*ptr++ = hextable[haddr.c[4] >> 4];
*ptr++ = hextable[haddr.c[4] & 0xF];
*ptr++ = hextable[haddr.c[5] >> 4];
*ptr++ = hextable[haddr.c[5] & 0xF];
*ptr++ = '*';
*ptr++ = key.c[0];
*ptr++
= key.c[1];
*ptr++ = key.c[2];
*ptr++ = key.c[3];

ptr =
    StuffData(skb->data + sizeof(STRIP_Header), len, ptr,
       &stuffstate);

if (strip_info->firmware_level >= ChecksummedMessages)
 ptr = add_checksum(buffer + 1, ptr);

*ptr++ = 0x0D;
```

```
 return (ptr);
}

static void strip_send(struct strip *strip_info, struct sk_buff *skb)
{
 MetricomAddress haddr;
 unsigned char *ptr = strip_info->tx_buff;
 int doreset = (long) jiffies - strip_info->watchdog_doreset >= 0;
 int doprobe = (long) jiffies - strip_info->watchdog_doprobe >= 0
    && !doreset;
 __be32 addr, brd;

 /*
  * 1. If we have a packet, encapsulate it and put it in the buffer
  */
 if (skb) {
  char *newptr = strip_make_packet(ptr, strip_info, skb);
  strip_info->tx_pps_count++;
  if (!newptr)
   strip_info->tx_dropped++;
  else {
   ptr = newptr;
   strip_info->sx_pps_count++;
   strip_info->tx_packets++; /* Count another successful packet */
#ifdef EXT_COUNTERS
   strip_info->tx_bytes += skb->len;
   strip_info->tx_rbytes
 += ptr - strip_info->tx_buff;
#endif
   /*DumpData("Sending:", strip_info, strip_info->tx_buff, ptr); */
   /*HexDump("Sending", strip_info, strip_info->tx_buff, ptr); */
  }
 }

 /*
  * 2. If it is time for another tickle, tack it on, after the packet
  */
 if (doprobe) {
  StringDescriptor ts = CommandString[strip_info->next_command];
#if TICKLE_TIMERS
  {
   struct timeval tv;
   do_gettimeofday(&tv);
   printk(KERN_INFO "**** Sending tickle string %d      at %02d.%06d\n",
          strip_info->next_command, tv.tv_sec % 100,
          tv.tv_usec);
  }
#endif
```

```
 if (ptr == strip_info->tx_buff)
  *ptr++ = 0x0D;

 *ptr++ = '*'; /* First send "**" to provoke an error message */
 *ptr++ = '*';

 /* Then add the command */
 memcpy(ptr, ts.string, ts.length);

 /* Add a checksum ? */
 if (strip_info->firmware_level < ChecksummedMessages)
  ptr += ts.length;
 else
  ptr = add_checksum(ptr, ptr + ts.length);

 *ptr++ = 0x0D; /* Terminate the command with a <CR>
*/

 /* Cycle to next periodic command? */
 if (strip_info->firmware_level >= StructuredMessages)
  if (++strip_info->next_command >=
      ARRAY_SIZE(CommandString))
   strip_info->next_command = 0;
#ifdef EXT_COUNTERS
 strip_info->tx_ebytes += ts.length;
#endif
 strip_info->watchdog_doprobe = jiffies + 10 * HZ;
 strip_info->watchdog_doreset = jiffies + 1 * HZ;
 /*printk(KERN_INFO "%s: Routine radio test.\n", strip_info->dev->name); */
 }

 /*
  * 3. Set up the strip_info ready to send the data (if any).
  */
 strip_info->tx_head = strip_info->tx_buff;
 strip_info->tx_left = ptr - strip_info->tx_buff;
 strip_info->tty->flags |= (1 << TTY_DO_WRITE_WAKEUP);

 /*
  * 4. Debugging check to make sure we're not overflowing the buffer.
  */
 if (strip_info->tx_size - strip_info->tx_left < 20)
  printk(KERN_ERR "%s: Sending%5d bytes;%5d bytes free.\n",
         strip_info->dev->name, strip_info->tx_left,
         strip_info->tx_size - strip_info->tx_left);

 /*
  * 5. If watchdog has
```

```
expired, reset the radio. Note: if there's data waiting in
 * the buffer, strip_write_some_more will send it after the reset has finished
 */
if (doreset) {
 ResetRadio(strip_info);
 return;
}

if (1) {
 struct in_device *in_dev;

 brd = addr = 0;
 rcu_read_lock();
 in_dev = __in_dev_get_rcu(strip_info->dev);
 if (in_dev) {
  if (in_dev->ifa_list) {
   brd = in_dev->ifa_list->ifa_broadcast;
   addr = in_dev->ifa_list->ifa_local;
  }
 }
 rcu_read_unlock();
}


/*
 * 6. If it is time for a periodic ARP, queue one up to be sent.
 * We only do this if:
 *  1. The radio is working
 *  2. It's time to send another periodic ARP
 *  3. We really know what our address is (and it is not manually set to zero)
 *  4. We have a designated broadcast address configured
 * If we queue up an ARP packet when we don't have a designated broadcast
 * address configured, then the packet will just have to be discarded in
 * strip_make_packet. This is not fatal, but it causes misleading
information
 * to be displayed in tcpdump. tcpdump will report that periodic APRs are
 * being sent, when in fact they are not, because they are all being dropped
 * in the strip_make_packet routine.
 */
if (strip_info->working
    && (long) jiffies - strip_info->gratuitous_arp >= 0
    && memcmp(strip_info->dev->dev_addr, zero_address.c,
       sizeof(zero_address))
    && arp_query(haddr.c, brd, strip_info->dev)) {
 /*printk(KERN_INFO "%s: Sending gratuitous ARP with interval %ld\n",
    strip_info->dev->name, strip_info->arp_interval / HZ); */
 strip_info->gratuitous_arp =
     jiffies + strip_info->arp_interval;
```

```
 strip_info->arp_interval *= 2;
 if (strip_info->arp_interval > MaxARPInterval)
  strip_info->arp_interval = MaxARPInterval;
 if (addr)
  arp_send(ARPOP_REPLY, ETH_P_ARP, addr, /* Target address of ARP packet is our address */
    strip_info->dev, /* Device to send packet on */
    addr, /* Source IP address this ARP packet comes from */
    NULL, /*
 Destination HW address is NULL (broadcast it) */
    strip_info->dev->dev_addr, /* Source HW address is our HW address */
    strip_info->dev->dev_addr); /* Target HW address is our HW address (redundant) */
 }

 /*
  * 7. All ready. Start the transmission
  */
 strip_write_some_more(strip_info->tty);
}

/* Encapsulate a datagram and kick it into a TTY queue. */
static int strip_xmit(struct sk_buff *skb, struct net_device *dev)
{
 struct strip *strip_info = netdev_priv(dev);

 if (!netif_running(dev)) {
  printk(KERN_ERR "%s: xmit call when iface is down\n",
        dev->name);
  return (1);
 }

 netif_stop_queue(dev);

 del_timer(&strip_info->idle_timer);


 if (time_after(jiffies, strip_info->pps_timer + HZ)) {
  unsigned long t = jiffies - strip_info->pps_timer;
  unsigned long rx_pps_count = (strip_info->rx_pps_count * HZ * 8 + t / 2) / t;
  unsigned long tx_pps_count = (strip_info->tx_pps_count * HZ * 8 + t / 2) / t;
  unsigned long sx_pps_count = (strip_info->sx_pps_count
 * HZ * 8 + t / 2) / t;

  strip_info->pps_timer = jiffies;
  strip_info->rx_pps_count = 0;
  strip_info->tx_pps_count = 0;
  strip_info->sx_pps_count = 0;

  strip_info->rx_average_pps = (strip_info->rx_average_pps + rx_pps_count + 1) / 2;
```

```
 strip_info->tx_average_pps = (strip_info->tx_average_pps + tx_pps_count + 1) / 2;
 strip_info->sx_average_pps = (strip_info->sx_average_pps + sx_pps_count + 1) / 2;

 if (rx_pps_count / 8 >= 10)
  printk(KERN_INFO "%s: WARNING: Receiving %ld packets per second.\n",
        strip_info->dev->name, rx_pps_count / 8);
 if (tx_pps_count / 8 >= 10)
  printk(KERN_INFO "%s: WARNING: Tx        %ld packets per second.\n",
        strip_info->dev->name, tx_pps_count / 8);
 if (sx_pps_count / 8 >= 10)
  printk(KERN_INFO "%s: WARNING: Sending   %ld packets per second.\n",
        strip_info->dev->name, sx_pps_count / 8);
}

spin_lock_bh(&strip_lock);

strip_send(strip_info, skb);

spin_unlock_bh(&strip_lock);

if (skb)
 dev_kfree_skb(skb);
return
0;
}

/*
* IdleTask periodically calls strip_xmit, so even when we have no IP packets
* to send for an extended period of time, the watchdog processing still gets
* done to ensure that the radio stays in Starmode
*/

static void strip_IdleTask(unsigned long parameter)
{
 strip_xmit(NULL, (struct net_device *) parameter);
}

/*
* Create the MAC header for an arbitrary protocol layer
*
* saddr!=NULL       means use this specific address (n/a for Metricom)
* saddr==NULL       means use default device source address
* daddr!=NULL       means use this destination address
* daddr==NULL       means leave destination address alone
*               (e.g. unresolved arp -- kernel will call
*               rebuild_header later to fill in the address)
*/
```

```
static int strip_header(struct sk_buff *skb, struct net_device *dev,
  unsigned short type, void *daddr, void *saddr,
  unsigned len)
{
 struct strip *strip_info = netdev_priv(dev);
 STRIP_Header *header = (STRIP_Header
 *) skb_push(skb, sizeof(STRIP_Header));

 /*printk(KERN_INFO "%s: strip_header 0x%04X %s\n", dev->name, type,
   type == ETH_P_IP ? "IP" : type == ETH_P_ARP ? "ARP" : ""); */

 header->src_addr = strip_info->true_dev_addr;
 header->protocol = htons(type);

 /*HexDump("strip_header", netdev_priv(dev), skb->data, skb->data + skb->len); */

 if (!daddr)
  return (-dev->hard_header_len);

 header->dst_addr = *(MetricomAddress *) daddr;
 return (dev->hard_header_len);
}

/*
* Rebuild the MAC header. This is called after an ARP
* (or in future other address resolution) has completed on this
* sk_buff. We now let ARP fill in the other fields.
* I think this should return zero if packet is ready to send,
* or non-zero if it needs more time to do an address lookup
*/

static int strip_rebuild_header(struct sk_buff *skb)
{
#ifdef CONFIG_INET
 STRIP_Header *header = (STRIP_Header *) skb->data;

 /* Arp find returns zero if if knows the address, */
 /* or if it doesn't know the address
 it sends an ARP packet and returns non-zero */
 return arp_find(header->dst_addr.c, skb) ? 1 : 0;
#else
 return 0;
#endif
}


/*****************************************************************************/
/* Receiving routines       */
```

```
/*
* This function parses the response to the ATS300? command,
* extracting the radio version and serial number.
*/
static void get_radio_version(struct strip *strip_info, __u8 * ptr, __u8 * end)
{
 __u8 *p, *value_begin, *value_end;
 int len;

 /* Determine the beginning of the second line of the payload */
 p = ptr;
 while (p < end && *p != 10)
  p++;
 if (p >= end)
  return;
 p++;
 value_begin = p;

 /* Determine the end of line */
 while (p < end && *p != 10)
  p++;
 if (p >= end)
  return;
 value_end = p;
 p++;

 len = value_end - value_begin;
 len = min_t(int, len, sizeof(FirmwareVersion) - 1);
 if (strip_info->firmware_version.c[0] == 0)
  printk(KERN_INFO "%s: Radio Firmware: %.*s\n",
        strip_info->dev->name, len, value_begin);
 sprintf(strip_info->firmware_version.c,
 "%.*s", len, value_begin);

 /* Look for the first colon */
 while (p < end && *p != ':')
  p++;
 if (p >= end)
  return;
 /* Skip over the space */
 p += 2;
 len = sizeof(SerialNumber) - 1;
 if (p + len <= end) {
  sprintf(strip_info->serial_number.c, "%.*s", len, p);
 } else {
  printk(KERN_DEBUG
        "STRIP: radio serial number shorter (%zd) than expected (%d)\n",
```

```
        end - p, len);
 }
}

/*
* This function parses the response to the ATS325? command,
* extracting the radio battery voltage.
*/
static void get_radio_voltage(struct strip *strip_info, __u8 * ptr, __u8 * end)
{
 int len;

 len = sizeof(BatteryVoltage) - 1;
 if (ptr + len <= end) {
  sprintf(strip_info->battery_voltage.c, "%.*s", len, ptr);
 } else {
  printk(KERN_DEBUG
        "STRIP: radio voltage string shorter (%zd) than expected (%d)\n",
        end - ptr, len);
 }
}

/*
* This function parses the responses to the AT~LA and ATS311 commands,
* which list the radio's neighbours.
 */
static void get_radio_neighbours(MetricomNodeTable * table, __u8 * ptr, __u8 * end)
{
 table->num_nodes = 0;
 while (ptr < end && table->num_nodes < NODE_TABLE_SIZE) {
  MetricomNode *node = &table->node[table->num_nodes++];
  char *dst = node->c, *limit = dst + sizeof(*node) - 1;
  while (ptr < end && *ptr <= 32)
   ptr++;
  while (ptr < end && dst < limit && *ptr != 10)
   *dst++ = *ptr++;
  *dst++ = 0;
  while (ptr < end && ptr[-1] != 10)
   ptr++;
 }
 do_gettimeofday(&table->timestamp);
}

static int get_radio_address(struct strip *strip_info, __u8 * p)
{
 MetricomAddress addr;

 if (string_to_radio_address(&addr, p))
```

```
 return (1);

 /* See if our radio address has changed */
 if (memcmp(strip_info->true_dev_addr.c, addr.c, sizeof(addr))) {
  MetricomAddressString addr_string;
  radio_address_to_string(&addr, &addr_string);
  printk(KERN_INFO "%s: Radio address = %s\n",
         strip_info->dev->name, addr_string.c);
  strip_info->true_dev_addr = addr;
  if (!strip_info->manual_dev_addr)
   *(MetricomAddress
 *) strip_info->dev->dev_addr =
       addr;
  /* Give the radio a few seconds to get its head straight, then send an arp */
  strip_info->gratuitous_arp = jiffies + 15 * HZ;
  strip_info->arp_interval = 1 * HZ;
 }
 return (0);
}

static int verify_checksum(struct strip *strip_info)
{
 __u8 *p = strip_info->sx_buff;
 __u8 *end = strip_info->sx_buff + strip_info->sx_count - 4;
 u_short sum =
     (READHEX16(end[0]) << 12) | (READHEX16(end[1]) << 8) |
     (READHEX16(end[2]) << 4) | (READHEX16(end[3]));
 while (p < end)
  sum -= *p++;
 if (sum == 0 && strip_info->firmware_level == StructuredMessages) {
  strip_info->firmware_level = ChecksummedMessages;
  printk(KERN_INFO "%s: Radio provides message checksums\n",
         strip_info->dev->name);
 }
 return (sum == 0);
}

static void RecvErr(char *msg, struct strip *strip_info)
{
 __u8 *ptr = strip_info->sx_buff;
 __u8 *end = strip_info->sx_buff + strip_info->sx_count;
 DumpData(msg, strip_info, ptr,
 end);
 strip_info->rx_errors++;
}

static void RecvErr_Message(struct strip *strip_info, __u8 * sendername,
        const __u8 * msg, u_long len)
```

```
{
 if (has_prefix(msg, len, "001")) { /* Not in StarMode! */
  RecvErr("Error Msg:", strip_info);
  printk(KERN_INFO "%s: Radio %s is not in StarMode\n",
        strip_info->dev->name, sendername);
 }

 else if (has_prefix(msg, len, "002")) { /* Remap handle */
  /* We ignore "Remap handle" messages for now */
 }

 else if (has_prefix(msg, len, "003")) { /* Can't resolve name */
  RecvErr("Error Msg:", strip_info);
  printk(KERN_INFO "%s: Destination radio name is unknown\n",
        strip_info->dev->name);
 }

 else if (has_prefix(msg, len, "004")) { /* Name too small or missing */
  strip_info->watchdog_doreset = jiffies + LongTime;
#if TICKLE_TIMERS
  {
   struct timeval tv;
   do_gettimeofday(&tv);
   printk(KERN_INFO
        "**** Got ERR_004 response         at %02d.%06d\n",
        tv.tv_sec % 100, tv.tv_usec);
  }
#endif
  if (!strip_info->working)
  {
   strip_info->working = TRUE;
   printk(KERN_INFO "%s: Radio now in starmode\n",
        strip_info->dev->name);
   /*
    * If the radio has just entered a working state, we should do our first
    * probe ASAP, so that we find out our radio address etc. without delay.
    */
   strip_info->watchdog_doprobe = jiffies;
  }
  if (strip_info->firmware_level == NoStructure && sendername) {
   strip_info->firmware_level = StructuredMessages;
   strip_info->next_command = 0; /* Try to enable checksums ASAP */
   printk(KERN_INFO
        "%s: Radio provides structured messages\n",
        strip_info->dev->name);
  }
  if (strip_info->firmware_level >= StructuredMessages) {
   /*
```

```
   * If this message has a valid checksum on the end, then the call to verify_checksum
   * will elevate the firmware_level to ChecksummedMessages for us. (The actual return
   * code from verify_checksum is ignored here.)
   */
  verify_checksum(strip_info);
  /*
   * If the radio
has structured messages but we don't yet have all our information about it,
   * we should do probes without delay, until we have gathered all the information
   */
  if (!GOT_ALL_RADIO_INFO(strip_info))
   strip_info->watchdog_doprobe = jiffies;
 }
}

else if (has_prefix(msg, len, "005")) /* Bad count specification */
 RecvErr("Error Msg:", strip_info);

else if (has_prefix(msg, len, "006")) /* Header too big */
 RecvErr("Error Msg:", strip_info);

else if (has_prefix(msg, len, "007")) { /* Body too big */
 RecvErr("Error Msg:", strip_info);
 printk(KERN_ERR
       "%s: Error! Packet size too big for radio.\n",
       strip_info->dev->name);
}

else if (has_prefix(msg, len, "008")) { /* Bad character in name */
 RecvErr("Error Msg:", strip_info);
 printk(KERN_ERR
       "%s: Radio name contains illegal character\n",
       strip_info->dev->name);
}

else if (has_prefix(msg, len, "009")) /* No count or line terminator */
 RecvErr("Error Msg:", strip_info);

else
if (has_prefix(msg, len, "010")) /* Invalid checksum */
 RecvErr("Error Msg:", strip_info);

else if (has_prefix(msg, len, "011")) /* Checksum didn't match */
 RecvErr("Error Msg:", strip_info);

else if (has_prefix(msg, len, "012")) /* Failed to transmit packet */
 RecvErr("Error Msg:", strip_info);
```

```
 else
  RecvErr("Error Msg:", strip_info);
}

static void process_AT_response(struct strip *strip_info, __u8 * ptr,
    __u8 * end)
{
 u_long len;
 __u8 *p = ptr;
 while (p < end && p[-1] != 10)
  p++;  /* Skip past first newline character */
 /* Now ptr points to the AT command, and p points to the text of the response. */
 len = p - ptr;

#if TICKLE_TIMERS
 {
  struct timeval tv;
  do_gettimeofday(&tv);
  printk(KERN_INFO "**** Got AT response %.7s      at %02d.%06d\n",
         ptr, tv.tv_sec % 100, tv.tv_usec);
 }
#endif

 if (has_prefix(ptr, len, "ATS300?"))
  get_radio_version(strip_info, p, end);
 else if (has_prefix(ptr, len, "ATS305?"))
  get_radio_address(strip_info,
 p);
 else if (has_prefix(ptr, len, "ATS311?"))
  get_radio_neighbours(&strip_info->poletops, p, end);
 else if (has_prefix(ptr, len, "ATS319=7"))
  verify_checksum(strip_info);
 else if (has_prefix(ptr, len, "ATS325?"))
  get_radio_voltage(strip_info, p, end);
 else if (has_prefix(ptr, len, "AT~LA"))
  get_radio_neighbours(&strip_info->portables, p, end);
 else
  RecvErr("Unknown AT Response:", strip_info);
}

static void process_ACK(struct strip *strip_info, __u8 * ptr, __u8 * end)
{
 /* Currently we don't do anything with ACKs from the radio */
}

static void process_Info(struct strip *strip_info, __u8 * ptr, __u8 * end)
{
 if (ptr + 16 > end)
```

```
  RecvErr("Bad Info Msg:", strip_info);
}

static struct net_device *get_strip_dev(struct strip *strip_info)
{
 /* If our hardware address is *manually set* to zero, and we know our */
 /* real radio hardware address, try to find another strip device that has been */
 /* manually set to that address that we can 'transfer ownership' of this packet
 to  */
 if (strip_info->manual_dev_addr &&
     !memcmp(strip_info->dev->dev_addr, zero_address.c,
      sizeof(zero_address))
     && memcmp(&strip_info->true_dev_addr, zero_address.c,
        sizeof(zero_address))) {
  struct net_device *dev;
  read_lock_bh(&dev_base_lock);
  dev = dev_base;
  while (dev) {
   if (dev->type == strip_info->dev->type &&
       !memcmp(dev->dev_addr,
        &strip_info->true_dev_addr,
        sizeof(MetricomAddress))) {
    printk(KERN_INFO
           "%s: Transferred packet ownership to %s.\n",
           strip_info->dev->name, dev->name);
    read_unlock_bh(&dev_base_lock);
    return (dev);
   }
   dev = dev->next;
  }
  read_unlock_bh(&dev_base_lock);
 }
 return (strip_info->dev);
}

/*
* Send one completely decapsulated datagram to the next layer.
*/

static void deliver_packet(struct strip *strip_info, STRIP_Header * header,
       __u16 packetlen)
{
 struct sk_buff *skb = dev_alloc_skb(sizeof(STRIP_Header) + packetlen);
 if (!skb) {
  printk(KERN_ERR
  "%s: memory squeeze, dropping packet.\n",
         strip_info->dev->name);
  strip_info->rx_dropped++;
```

```
 } else {
  memcpy(skb_put(skb, sizeof(STRIP_Header)), header,
         sizeof(STRIP_Header));
  memcpy(skb_put(skb, packetlen), strip_info->rx_buff,
         packetlen);
  skb->dev = get_strip_dev(strip_info);
  skb->protocol = header->protocol;
  skb->mac.raw = skb->data;

  /* Having put a fake header on the front of the sk_buff for the */
  /* benefit of tools like tcpdump, skb_pull now 'consumes' that  */
  /* fake header before we hand the packet up to the next layer.  */
  skb_pull(skb, sizeof(STRIP_Header));

  /* Finally, hand the packet up to the next layer (e.g. IP or ARP, etc.) */
  strip_info->rx_packets++;
  strip_info->rx_pps_count++;
#ifdef EXT_COUNTERS
  strip_info->rx_bytes += packetlen;
#endif
  skb->dev->last_rx = jiffies;
  netif_rx(skb);
 }
}

static void process_IP_packet(struct strip *strip_info,
         STRIP_Header * header, __u8 * ptr,
         __u8
 * end)
{
 __u16 packetlen;

 /* Decode start of the IP packet header */
 ptr = UnStuffData(ptr, end, strip_info->rx_buff, 4);
 if (!ptr) {
  RecvErr("IP Packet too short", strip_info);
  return;
 }

 packetlen = ((__u16) strip_info->rx_buff[2] << 8) | strip_info->rx_buff[3];

 if (packetlen > MAX_RECV_MTU) {
  printk(KERN_INFO "%s: Dropping oversized received IP packet: %d bytes\n",
         strip_info->dev->name, packetlen);
  strip_info->rx_dropped++;
  return;
 }
```

```
/*printk(KERN_INFO "%s: Got %d byte IP packet\n", strip_info->dev->name, packetlen); */

/* Decode remainder of the IP packet */
ptr =
    UnStuffData(ptr, end, strip_info->rx_buff + 4, packetlen - 4);
if (!ptr) {
 RecvErr("IP Packet too short", strip_info);
 return;
}

if (ptr < end) {
 RecvErr("IP Packet too long", strip_info);
 return;
}

header->protocol = htons(ETH_P_IP);

deliver_packet(strip_info, header, packetlen);
}

static void process_ARP_packet(struct strip *strip_info,
        STRIP_Header
 * header, __u8 * ptr,
        __u8 * end)
{
 __u16 packetlen;
 struct arphdr *arphdr = (struct arphdr *) strip_info->rx_buff;

 /* Decode start of the ARP packet */
 ptr = UnStuffData(ptr, end, strip_info->rx_buff, 8);
 if (!ptr) {
  RecvErr("ARP Packet too short", strip_info);
  return;
 }

 packetlen = 8 + (arphdr->ar_hln + arphdr->ar_pln) * 2;

 if (packetlen > MAX_RECV_MTU) {
  printk(KERN_INFO
        "%s: Dropping oversized received ARP packet: %d bytes\n",
        strip_info->dev->name, packetlen);
  strip_info->rx_dropped++;
  return;
 }

 /*printk(KERN_INFO "%s: Got %d byte ARP %s\n",
    strip_info->dev->name, packetlen,
    ntohs(arphdr->ar_op) == ARPOP_REQUEST ? "request" : "reply"); */
```

```
/* Decode remainder of the ARP packet */
ptr =
    UnStuffData(ptr, end, strip_info->rx_buff + 8, packetlen - 8);
if (!ptr) {
 RecvErr("ARP Packet too short", strip_info);
 return;
}

if (ptr < end) {
 RecvErr("ARP Packet too long", strip_info);
 return;
}

header->protocol
= htons(ETH_P_ARP);

 deliver_packet(strip_info, header, packetlen);
}

/*
* process_text_message processes a <CR>-terminated block of data received
* from the radio that doesn't begin with a '*' character. All normal
* Starmode communication messages with the radio begin with a '*',
* so any text that does not indicates a serial port error, a radio that
* is in Hayes command mode instead of Starmode, or a radio with really
* old firmware that doesn't frame its Starmode responses properly.
*/
static void process_text_message(struct strip *strip_info)
{
 __u8 *msg = strip_info->sx_buff;
 int len = strip_info->sx_count;

 /* Check for anything that looks like it might be our radio name */
 /* (This is here for backwards compatibility with old firmware)  */
 if (len == 9 && get_radio_address(strip_info, msg) == 0)
  return;

 if (text_equal(msg, len, "OK"))
  return;  /* Ignore 'OK' responses from prior commands */
 if (text_equal(msg, len, "ERROR"))
  return;  /* Ignore 'ERROR' messages
 */
 if (has_prefix(msg, len, "ate0q1"))
  return;  /* Ignore character echo back from the radio */

 /* Catch other error messages */
 /* (This is here for backwards compatibility with old firmware) */
```

```
 if (has_prefix(msg, len, "ERR_")) {
  RecvErr_Message(strip_info, NULL, &msg[4], len - 4);
  return;
 }

 RecvErr("No initial *", strip_info);
}

/*
* process_message processes a <CR>-terminated block of data received
* from the radio. If the radio is not in Starmode or has old firmware,
* it may be a line of text in response to an AT command. Ideally, with
* a current radio that's properly in Starmode, all data received should
* be properly framed and checksummed radio message blocks, containing
* either a starmode packet, or a other communication from the radio
* firmware, like "INF_" Info messages and &COMMAND responses.
*/
static void process_message(struct strip *strip_info)
{
 STRIP_Header header = { zero_address, zero_address, 0 };
 __u8 *ptr = strip_info->sx_buff;
 __u8
 *end = strip_info->sx_buff + strip_info->sx_count;
 __u8 sendername[32], *sptr = sendername;
 MetricomKey key;

 /*HexDump("Receiving", strip_info, ptr, end); */

 /* Check for start of address marker, and then skip over it */
 if (*ptr == '*')
  ptr++;
 else {
  process_text_message(strip_info);
  return;
 }

 /* Copy out the return address */
 while (ptr < end && *ptr != '*'
       && sptr < ARRAY_END(sendername) - 1)
  *sptr++ = *ptr++;
 *sptr = 0;  /* Null terminate the sender name */

 /* Check for end of address marker, and skip over it */
 if (ptr >= end || *ptr != '*') {
  RecvErr("No second *", strip_info);
  return;
 }
 ptr++;   /* Skip the second '*' */
```

```
/* If the sender name is "&COMMAND", ignore this 'packet'       */
/* (This is here for backwards compatibility with old firmware) */
if (!strcmp(sendername, "&COMMAND")) {
 strip_info->firmware_level = NoStructure;
 strip_info->next_command = CompatibilityCommand;
 return;
}

if (ptr + 4 > end) {
 RecvErr("No
proto key", strip_info);
 return;
}

/* Get the protocol key out of the buffer */
key.c[0] = *ptr++;
key.c[1] = *ptr++;
key.c[2] = *ptr++;
key.c[3] = *ptr++;

/* If we're using checksums, verify the checksum at the end of the packet */
if (strip_info->firmware_level >= ChecksummedMessages) {
 end -= 4; /* Chop the last four bytes off the packet (they're the checksum) */
 if (ptr > end) {
  RecvErr("Missing Checksum", strip_info);
  return;
 }
 if (!verify_checksum(strip_info)) {
  RecvErr("Bad Checksum", strip_info);
  return;
 }
}

/*printk(KERN_INFO "%s: Got packet from \"%s\".\n", strip_info->dev->name, sendername); */

/*
 * Fill in (pseudo) source and destination addresses in the packet.
 * We assume that the destination address was our address (the radio does not
 * tell us this). If the radio supplies a source address, then we use it.
 */
header.dst_addr = strip_info->true_dev_addr;
string_to_radio_address(&header.src_addr, sendername);

#ifdef EXT_COUNTERS
 if
 (key.l == SIP0Key.l) {
  strip_info->rx_rbytes += (end - ptr);
```

```
  process_IP_packet(strip_info, &header, ptr, end);
 } else if (key.l == ARP0Key.l) {
  strip_info->rx_rbytes += (end - ptr);
  process_ARP_packet(strip_info, &header, ptr, end);
 } else if (key.l == ATR_Key.l) {
  strip_info->rx_ebytes += (end - ptr);
  process_AT_response(strip_info, ptr, end);
 } else if (key.l == ACK_Key.l) {
  strip_info->rx_ebytes += (end - ptr);
  process_ACK(strip_info, ptr, end);
 } else if (key.l == INF_Key.l) {
  strip_info->rx_ebytes += (end - ptr);
  process_Info(strip_info, ptr, end);
 } else if (key.l == ERR_Key.l) {
  strip_info->rx_ebytes += (end - ptr);
  RecvErr_Message(strip_info, sendername, ptr, end - ptr);
 } else
  RecvErr("Unrecognized protocol key", strip_info);
#else
 if (key.l == SIP0Key.l)
  process_IP_packet(strip_info, &header, ptr, end);
 else if (key.l == ARP0Key.l)
  process_ARP_packet(strip_info, &header, ptr, end);
 else if (key.l == ATR_Key.l)
  process_AT_response(strip_info,
 ptr, end);
 else if (key.l == ACK_Key.l)
  process_ACK(strip_info, ptr, end);
 else if (key.l == INF_Key.l)
  process_Info(strip_info, ptr, end);
 else if (key.l == ERR_Key.l)
  RecvErr_Message(strip_info, sendername, ptr, end - ptr);
 else
  RecvErr("Unrecognized protocol key", strip_info);
#endif
}

#define TTYERROR(X) ((X) == TTY_BREAK   ? "Break"           : \
                     (X) == TTY_FRAME   ? "Framing Error"   : \
                     (X) == TTY_PARITY  ? "Parity Error"    : \
                     (X) == TTY_OVERRUN ? "Hardware Overrun" : "Unknown Error")

/*
* Handle the 'receiver data ready' interrupt.
* This function is called by the 'tty_io' module in the kernel when
* a block of STRIP data has been received, which can now be decapsulated
* and sent on to some IP layer for further processing.
*/
```

```
static void strip_receive_buf(struct tty_struct *tty, const unsigned char *cp,
    char *fp, int count)
{
 struct strip *strip_info = (struct strip
 *) tty->disc_data;
 const unsigned char *end = cp + count;

 if (!strip_info || strip_info->magic != STRIP_MAGIC
     || !netif_running(strip_info->dev))
  return;

 spin_lock_bh(&strip_lock);
#if 0
 {
  struct timeval tv;
  do_gettimeofday(&tv);
  printk(KERN_INFO
         "**** strip_receive_buf: %3d bytes at %02d.%06d\n",
         count, tv.tv_sec % 100, tv.tv_usec);
 }
#endif

#ifdef EXT_COUNTERS
 strip_info->rx_sbytes += count;
#endif

 /* Read the characters out of the buffer */
 while (cp < end) {
  if (fp && *fp)
   printk(KERN_INFO "%s: %s on serial port\n",
          strip_info->dev->name, TTYERROR(*fp));
  if (fp && *fp++ && !strip_info->discard) { /* If there's a serial error, record it */
   /* If we have some characters in the buffer, discard them */
   strip_info->discard = strip_info->sx_count;
   strip_info->rx_errors++;
  }

  /* Leading control characters (CR, NL, Tab, etc.) are ignored */
  if (strip_info->sx_count > 0 || *cp >= ' ') {
   if (*cp == 0x0D) { /* If
 end of packet, decide what to do with it */
    if (strip_info->sx_count > 3000)
     printk(KERN_INFO
            "%s: Cut a %d byte packet (%zd bytes remaining)%s\n",
            strip_info->dev->name,
            strip_info->sx_count,
            end - cp - 1,
```

```
            strip_info->
            discard ? " (discarded)" :
            "");
    if (strip_info->sx_count >
        strip_info->sx_size) {
     strip_info->rx_over_errors++;
     printk(KERN_INFO
            "%s: sx_buff overflow (%d bytes total)\n",
            strip_info->dev->name,
            strip_info->sx_count);
    } else if (strip_info->discard)
     printk(KERN_INFO
            "%s: Discarding bad packet (%d/%d)\n",
            strip_info->dev->name,
            strip_info->discard,
            strip_info->sx_count);
    else
     process_message(strip_info);
    strip_info->discard = 0;
    strip_info->sx_count = 0;
   } else {
    /* Make sure we have space in the buffer */
    if (strip_info->sx_count <

   strip_info->sx_size)
     strip_info->sx_buff[strip_info->
           sx_count] =
         *cp;
    strip_info->sx_count++;
   }
  }
  cp++;
 }
 spin_unlock_bh(&strip_lock);
}


/************************************************************************/
/* General control routines      */

static int set_mac_address(struct strip *strip_info,
      MetricomAddress * addr)
{
 /*
  * We're using a manually specified address if the address is set
  * to anything other than all ones. Setting the address to all ones
  * disables manual mode and goes back to automatic address determination
  * (tracking the true address that the radio has).
```

```
 */
 strip_info->manual_dev_addr =
     memcmp(addr->c, broadcast_address.c,
     sizeof(broadcast_address));
 if (strip_info->manual_dev_addr)
  *(MetricomAddress *) strip_info->dev->dev_addr = *addr;
 else
  *(MetricomAddress *) strip_info->dev->dev_addr =
      strip_info->true_dev_addr;
 return 0;
}

static int strip_set_mac_address(struct net_device
 *dev, void *addr)
{
 struct strip *strip_info = netdev_priv(dev);
 struct sockaddr *sa = addr;
 printk(KERN_INFO "%s: strip_set_dev_mac_address called\n", dev->name);
 set_mac_address(strip_info, (MetricomAddress *) sa->sa_data);
 return 0;
}

static struct net_device_stats *strip_get_stats(struct net_device *dev)
{
 struct strip *strip_info = netdev_priv(dev);
 static struct net_device_stats stats;

 memset(&stats, 0, sizeof(struct net_device_stats));

 stats.rx_packets = strip_info->rx_packets;
 stats.tx_packets = strip_info->tx_packets;
 stats.rx_dropped = strip_info->rx_dropped;
 stats.tx_dropped = strip_info->tx_dropped;
 stats.tx_errors = strip_info->tx_errors;
 stats.rx_errors = strip_info->rx_errors;
 stats.rx_over_errors = strip_info->rx_over_errors;
 return (&stats);
}


/************************************************************************/
/* Opening and closing       */

/*
* Here's the order things happen:
* When the user runs "slattach -p strip ..."
*  1. The TTY
```

```
 module calls strip_open;;
*  2. strip_open calls strip_alloc
*  3.                strip_alloc calls register_netdev
*  4.                register_netdev calls strip_dev_init
*  5. then strip_open finishes setting up the strip_info
*
* When the user runs "ifconfig st<x> up address netmask ..."
*  6. strip_open_low gets called
*
* When the user runs "ifconfig st<x> down"
*  7. strip_close_low gets called
*
* When the user kills the slattach process
*  8. strip_close gets called
*  9. strip_close calls dev_close
* 10. if the device is still up, then dev_close calls strip_close_low
* 11. strip_close calls strip_free
*/

/* Open the low-level part of the STRIP channel. Easy! */

static int strip_open_low(struct net_device *dev)
{
 struct strip *strip_info = netdev_priv(dev);

 if (strip_info->tty == NULL)
  return (-ENODEV);

 if (!allocate_buffers(strip_info, dev->mtu))
  return (-ENOMEM);

 strip_info->sx_count = 0;
 strip_info->tx_left = 0;

 strip_info->discard = 0;
 strip_info->working
 = FALSE;
 strip_info->firmware_level = NoStructure;
 strip_info->next_command = CompatibilityCommand;
 strip_info->user_baud = get_baud(strip_info->tty);

 printk(KERN_INFO "%s: Initializing Radio.\n",
       strip_info->dev->name);
 ResetRadio(strip_info);
 strip_info->idle_timer.expires = jiffies + 1 * HZ;
 add_timer(&strip_info->idle_timer);
 netif_wake_queue(dev);
 return (0);
```

```
}


/*
* Close the low-level part of the STRIP channel. Easy!
*/

static int strip_close_low(struct net_device *dev)
{
 struct strip *strip_info = netdev_priv(dev);

 if (strip_info->tty == NULL)
  return -EBUSY;
 strip_info->tty->flags &= ~(1 << TTY_DO_WRITE_WAKEUP);

 netif_stop_queue(dev);

 /*
  * Free all STRIP frame buffers.
  */
 kfree(strip_info->rx_buff);
 strip_info->rx_buff = NULL;
 kfree(strip_info->sx_buff);
 strip_info->sx_buff = NULL;
 kfree(strip_info->tx_buff);
 strip_info->tx_buff = NULL;

 del_timer(&strip_info->idle_timer);
 return 0;
}

/*
* This routine is called
 by DDI when the
* (dynamically assigned) device is registered
*/

static void strip_dev_setup(struct net_device *dev)
{
 /*
  * Finish setting up the DEVICE info.
  */

 SET_MODULE_OWNER(dev);

 dev->trans_start = 0;
 dev->last_rx = 0;
 dev->tx_queue_len = 30; /* Drop after 30 frames queued */
```

```
 dev->flags = 0;
 dev->mtu = DEFAULT_STRIP_MTU;
 dev->type = ARPHRD_METRICOM; /* dtang */
 dev->hard_header_len = sizeof(STRIP_Header);
 /*
  *  dev->priv            Already holds a pointer to our struct strip
  */

 *(MetricomAddress *) & dev->broadcast = broadcast_address;
 dev->dev_addr[0] = 0;
 dev->addr_len = sizeof(MetricomAddress);

 /*
  * Pointers to interface service routines.
  */

 dev->open = strip_open_low;
 dev->stop = strip_close_low;
 dev->hard_start_xmit = strip_xmit;
 dev->hard_header = strip_header;
 dev->rebuild_header = strip_rebuild_header;
 dev->set_mac_address = strip_set_mac_address;
 dev->get_stats = strip_get_stats;
 dev->change_mtu = strip_change_mtu;
}

/*
* Free
 a STRIP channel.
*/

static void strip_free(struct strip *strip_info)
{
 spin_lock_bh(&strip_lock);
 list_del_rcu(&strip_info->list);
 spin_unlock_bh(&strip_lock);

 strip_info->magic = 0;

 free_netdev(strip_info->dev);
}


/*
* Allocate a new free STRIP channel
*/
static struct strip *strip_alloc(void)
```

```
{
 struct list_head *n;
 struct net_device *dev;
 struct strip *strip_info;

 dev = alloc_netdev(sizeof(struct strip), "st%d",
     strip_dev_setup);

 if (!dev)
  return NULL; /* If no more memory, return */


 strip_info = dev->priv;
 strip_info->dev = dev;

 strip_info->magic = STRIP_MAGIC;
 strip_info->tty = NULL;

 strip_info->gratuitous_arp = jiffies + LongTime;
 strip_info->arp_interval = 0;
 init_timer(&strip_info->idle_timer);
 strip_info->idle_timer.data = (long) dev;
 strip_info->idle_timer.function = strip_IdleTask;


 spin_lock_bh(&strip_lock);
rescan:
 /*
  * Search the list to find where to put our new entry
  * (and in the process decide what channel number it
 is
  * going to be)
  */
 list_for_each(n, &strip_list) {
  struct strip *s = hlist_entry(n, struct strip, list);

  if (s->dev->base_addr == dev->base_addr) {
   ++dev->base_addr;
   goto rescan;
  }
 }

 sprintf(dev->name, "st%ld", dev->base_addr);

 list_add_tail_rcu(&strip_info->list, &strip_list);
 spin_unlock_bh(&strip_lock);

 return strip_info;
```

```
}

/*
* Open the high-level part of the STRIP channel.
* This function is called by the TTY module when the
* STRIP line discipline is called for.  Because we are
* sure the tty line exists, we only have to link it to
* a free STRIP channel...
*/

static int strip_open(struct tty_struct *tty)
{
 struct strip *strip_info = (struct strip *) tty->disc_data;

 /*
  * First make sure we're not already connected.
  */

 if (strip_info && strip_info->magic == STRIP_MAGIC)
  return -EEXIST;

 /*
  * OK.  Find a free STRIP channel to use.
  */
 if ((strip_info = strip_alloc()) == NULL)
  return -ENFILE;

 /*
  * Register our newly created device
 so it can be ifconfig'd
  * strip_dev_init() will be called as a side-effect
  */

 if (register_netdev(strip_info->dev) != 0) {
  printk(KERN_ERR "strip: register_netdev() failed.\n");
  strip_free(strip_info);
  return -ENFILE;
 }

 strip_info->tty = tty;
 tty->disc_data = strip_info;
 tty->receive_room = 65536;

 if (tty->driver->flush_buffer)
  tty->driver->flush_buffer(tty);

 /*
  * Restore default settings
```

```
 */

strip_info->dev->type = ARPHRD_METRICOM; /* dtang */

/*
 * Set tty options
 */

tty->termios->c_iflag |= IGNBRK | IGNPAR; /* Ignore breaks and parity errors. */
tty->termios->c_cflag |= CLOCAL; /* Ignore modem control signals. */
tty->termios->c_cflag &= ~HUPCL; /* Don't close on hup */

printk(KERN_INFO "STRIP: device \"%s\" activated\n",
       strip_info->dev->name);

/*
 * Done.  We have linked the TTY line to a channel.
 */
return (strip_info->dev->base_addr);
}

/*
* Close down a STRIP channel.
* This means flushing out any pending queues, and
 then restoring the
* TTY line discipline to what it was before it got hooked to STRIP
* (which usually is TTY again).
*/

static void strip_close(struct tty_struct *tty)
{
 struct strip *strip_info = (struct strip *) tty->disc_data;

 /*
  * First make sure we're connected.
  */

 if (!strip_info || strip_info->magic != STRIP_MAGIC)
  return;

 unregister_netdev(strip_info->dev);

 tty->disc_data = NULL;
 strip_info->tty = NULL;
 printk(KERN_INFO "STRIP: device \"%s\" closed down\n",
       strip_info->dev->name);
 strip_free(strip_info);
 tty->disc_data = NULL;
```

```
}


/************************************************************************/
/* Perform I/O control calls on an active STRIP channel.  */

static int strip_ioctl(struct tty_struct *tty, struct file *file,
        unsigned int cmd, unsigned long arg)
{
 struct strip *strip_info = (struct strip *) tty->disc_data;

 /*
  * First make sure we're connected.
  */

 if (!strip_info || strip_info->magic != STRIP_MAGIC)
  return
 -EINVAL;

 switch (cmd) {
 case SIOCGIFNAME:
  if(copy_to_user((void __user *) arg, strip_info->dev->name, strlen(strip_info->dev->name) + 1))
   return -EFAULT;
  break;
 case SIOCSIFHWADDR:
 {
  MetricomAddress addr;
  //printk(KERN_INFO "%s: SIOCSIFHWADDR\n", strip_info->dev->name);
  if(copy_from_user(&addr, (void __user *) arg, sizeof(MetricomAddress)))
   return -EFAULT;
  return set_mac_address(strip_info, &addr);
 }
 /*
  * Allow stty to read, but not set, the serial port
  */

 case TCGETS:
 case TCGETA:
  return n_tty_ioctl(tty, file, cmd, arg);
  break;
 default:
  return -ENOIOCTLCMD;
  break;
 }
 return 0;
}
```

```
/************************************************************************/
/* Initialization       */

static struct tty_ldisc strip_ldisc = {
 .magic = TTY_LDISC_MAGIC,
 .name = "strip",
 .owner = THIS_MODULE,
 .open = strip_open,
 .close = strip_close,
 .ioctl = strip_ioctl,
 .receive_buf = strip_receive_buf,
 .write_wakeup = strip_write_some_more,
};

/*
* Initialize
 the STRIP driver.
* This routine is called at boot time, to bootstrap the multi-channel
* STRIP driver
*/

static char signon[] __initdata =
   KERN_INFO "STRIP: Version %s (unlimited channels)\n";

static int __init strip_init_driver(void)
{
 int status;

 printk(signon, StripVersion);


 /*
  * Fill in our line protocol discipline, and register it
  */
 if ((status = tty_register_ldisc(N_STRIP, &strip_ldisc)))
  printk(KERN_ERR "STRIP: can't register line discipline (err = %d)\n",
        status);

 /*
  * Register the status file with /proc
  */
 proc_net_fops_create("strip", S_IFREG | S_IRUGO, &strip_seq_fops);

 return status;
}

module_init(strip_init_driver);
```

```
static const char signoff[] __exitdata =
  KERN_INFO "STRIP: Module Unloaded\n";

static void __exit strip_exit_driver(void)
{
 int i;
 struct list_head *p,*n;

 /* module ref count rules assure that all entries are unregistered */
 list_for_each_safe(p, n, &strip_list) {
  struct strip *s = list_entry(p, struct
 strip, list);
  strip_free(s);
 }

 /* Unregister with the /proc/net file here. */
 proc_net_remove("strip");

 if ((i = tty_unregister_ldisc(N_STRIP)))
  printk(KERN_ERR "STRIP: can't unregister line discipline (err = %d)\n", i);

 printk(signoff);
}

module_exit(strip_exit_driver);

MODULE_AUTHOR("Stuart Cheshire <cheshire@cs.stanford.edu>");
MODULE_DESCRIPTION("Starmode Radio IP (STRIP) Device Driver");
MODULE_LICENSE("Dual BSD/GPL");

MODULE_SUPPORTED_DEVICE("Starmode Radio IP (STRIP) modem");
```

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/.svn/text-base/strip.c.svn-base
No license file was found, but licenses were detected in source scan.

```
/*
*  Copyright (C) 2003,2004 Aurelien Alleaume <slts@free.fr>
*
*  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
*
*  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
*
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/oid_mgt.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/prism54/.svn/text-base/oid_mgt.c.svn-base
No license file was found, but licenses were detected in source scan.

/*

 Broadcom BCM43xx wireless driver

 SYSFS support routines

 Copyright (c) 2006 Michael Buesch <mbuesch@freenet.de>

 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; see the file COPYING. If not, write to
 the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
 Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_sysfs.c
*
 /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_sysfs.c.svn-base
No license file was found, but licenses were detected in source scan.

/*

 Broadcom BCM43xx wireless driver

Copyright (c) 2005 Martin Langer <martin-langer@gmx.de>,
Stefano Brivio <st3@riseup.net>
Michael Buesch <mbuesch@freenet.de>
Danny van Dyk <kugelfang@gentoo.org>
Andreas Jaggi <andreas.jaggi@waterwave.ch>

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; see the file
COPYING. If not, write to
the Free Software Foundation, Inc., 51 Franklin Steet, Fifth Floor,
Boston, MA 02110-1301, USA.

*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_leds.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_leds.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/.svn/text-base/bcm43xx_ilt.c.svn-base
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/bcm43xx/bcm43xx_ilt.c
No license file was found, but licenses were detected in source scan.

/* zd_usb.c
*
* 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
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/zd_usb.c
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/zd1211rw/.svn/text-base/zd_usb.c.svn-base
No license file was found, but licenses were detected in source scan.

/* orinoco_pci.c
*
* Driver for Prism 2.5/3 devices that have a direct PCI interface
* (i.e. these are not PCMCIA cards in a PCMCIA-to-PCI bridge).
* The card contains only one PCI region, which contains all the usual
* hermes registers, as well as the COR register.
*
* Current maintainers are:
* Pavel Roskin <proski AT gnu.org>
* and David Gibson <hermes AT gibson.dropbear.id.au>
*
* Some of this code is borrowed from orinoco_plx.c
* Copyright (C) 2001 Daniel Barlow <dan AT telent.net>
* Some of this code is "inspired" by linux-wlan-ng-0.1.10, but nothing
* has been copied from it. linux-wlan-ng-0.1.10 is originally :
* Copyright (C) 1999 AbsoluteValue Systems, Inc. All Rights Reserved.
* This file originally written by:
* Copyright (C) 2001 Jean Tourrilhes <jt AT hpl.hp.com>
* And is now maintained by:
* (C) Copyright David Gibson, IBM Corp. 2002-2003.
*
* The contents of this file are subject
 to the Mozilla Public License
* Version 1.1 (the "License"); you may not use this file except in
* compliance with the License. You may obtain a copy of the License
* at http://www.mozilla.org/MPL/
*
* Software distributed under the License is distributed on an "AS IS"
* basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See
* the License for the specific language governing rights and
* limitations under the License.
*
* Alternatively, the contents of this file may be used under the
* terms of the GNU General Public License version 2 (the "GPL"), in
* which case the provisions of the GPL are applicable instead of the
* above. If you wish to allow the use of your version of this file
* only under the terms of the GPL and not to allow others to use your
* version of this file under the MPL, indicate your decision by
* deleting the provisions above and replace them with the notice and

* other provisions required by the GPL. If you do not delete the
* provisions
 above, a recipient may use your version of this file
* under either the MPL or the GPL.
*/

Found in path(s):
* /opt/cola/permits/2005174788_1715687491.5437253/0/wireless-zip/wireless/wireless/orinoco_pci.c