Open Source Used In Cisco Identity Services Engine 2.2

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: 78EE117C99-139725897

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 contact us at external-opensource-requests@cisco.com.

In your requests please include the following reference number 78EE117C99-139725897

Contents

1.1 alpine-baselayout 3.0.3 :r0

1.1.1 Available under license :

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

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.

1.2 alpine-conf 3.4.1 :r2

1.2.1 Available under license :

Copyright (c) <year> <copyright holders>

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.

1.3 alpine-keys 1.1 :r0

1.3.1 Available under license :

Copyright (c) <year> <copyright holders>

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 apk-tools 2.6.7 :r0

1.4.1 Available under license :

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.

1.5 busybox 1.24.2

1.5.1 Available under license :

bzip2 applet in busybox is based on lightly-modified source
of bzip2 version 1.0.4. bzip2 source is distributed
under the following conditions (copied verbatim from LICENSE file)
===

This program, "bzip2", the associated library "libbzip2", and all documentation, are copyright (C) 1996-2006 Julian R Seward. 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 origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

3. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

4. 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.

Julian Seward, Cambridge, UK.
jseward@bzip.org
bzip2/libbzip2 version 1.0.4 of 20 December 2006
--- A note on GPL versions

BusyBox is distributed under version 2 of the General Public License (included in its entirety, below). Version 2 is the only version of this license which this version of BusyBox (or modified versions derived from this one) may be distributed under.

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.

1.6 busybox-initscripts 3.0 :r3

1.6.1 Available under license :

--- A note on GPL versions

BusyBox is distributed under version 2 of the General Public License (included
in its entirety, below). Version 2 is the only version of this license which
this version of BusyBox (or modified versions derived from this one) may be
distributed under.

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.

1.7 libc-utils 0.7 :r0

1.7.1 Available under license :

```
/* $NetBSD: queue.h,v 1.68 2014/11/19 08:10:01 uebayasi Exp $ */

/*
* Copyright (c) 1991, 1993
* The Regents of the University of California.  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 University 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 REGENTS 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 REGENTS 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.
*
* @(#)queue.h 8.5 (Berkeley) 8/20/94
*/

#ifndef _SYS_QUEUE_H_
#define _SYS_QUEUE_H_

/*
* This file defines five types of data structures: singly-linked lists,
* lists, simple queues, tail queues, and circular queues.
*
* A singly-linked list is headed by a single forward pointer. The
* elements are singly linked for minimum space and pointer manipulation
* overhead at the expense of O(n) removal for arbitrary elements. New
* elements can be added to the list after an existing element or at the
```

```
* head of the list.  Elements being removed from the head of the list
* should use the explicit macro for this purpose for optimum
* efficiency. A singly-linked list may only be traversed in the forward
* direction.  Singly-linked lists are ideal for applications with large
* datasets and few or no removals or for implementing a LIFO queue.
*
* A list is headed by a single forward pointer (or an array of forward
* pointers for a hash table header). The elements are doubly linked
* so that an arbitrary element can be removed without a need to
* traverse the list. New elements can be added to the list before
* or after an existing element or at the head of the list. A list
* may only be traversed in the forward direction.
*
* A simple queue is headed by a pair of pointers, one the head of the
* list and the other to the tail of the list. The elements are singly
* linked to save space, so elements can only be removed from the
* head of the list. New elements can be added to the list after
* an existing element, at the head of the list, or at the end of the
* list. A simple queue may only be traversed in the forward direction.
*
* A tail queue is headed by a pair of pointers, one to the head of the
* list and the other to the tail of the list. The elements are doubly
* linked so that an arbitrary element can be removed without a need to
* traverse the list. New elements can be added to the list before or
* after an existing element, at the head of the list, or at the end of
* the list. A tail queue may be traversed in either direction.
*
* A circle queue is headed by a pair of pointers, one to the head of the
* list and the other to the tail of the list. The elements are doubly
* linked so that an arbitrary element can be removed without a need to
* traverse the list. New elements can be added to the list before or after
* an existing element, at the head of the list, or at the end of the list.
* A circle queue may be traversed in either direction, but has a more
* complex end of list detection.
*
* For details on the use of these macros, see the queue(3) manual page.
*/

/*
* Include the definition of NULL only on NetBSD because sys/null.h
* is not available elsewhere.  This conditional makes the header
* portable and it can simply be dropped verbatim into any system.
* The caveat is that on other systems some other header
* must provide NULL before the macros can be used.
*/
#ifdef __NetBSD__
#include <sys/null.h>
#endif
```

```
#if defined(QUEUEDEBUG)
# if defined(_KERNEL)
#  define QUEUEDEBUG_ABORT(...) panic(__VA_ARGS__)
# else
#  include <err.h>
#  define QUEUEDEBUG_ABORT(...) err(1, __VA_ARGS__)
# endif
#endif

/*
* Singly-linked List definitions.
*/
#define SLIST_HEAD(name, type)      \
struct name {        \
 struct type *slh_first; /* first element */   \
}

#define SLIST_HEAD_INITIALIZER(head)     \
 { NULL }

#define SLIST_ENTRY(type)      \
struct {        \
 struct type *sle_next; /* next element */   \
}

/*
* Singly-linked List access methods.
*/
#define SLIST_FIRST(head) ((head)->slh_first)
#define SLIST_END(head)  NULL
#define SLIST_EMPTY(head) ((head)->slh_first == NULL)
#define SLIST_NEXT(elm, field) ((elm)->field.sle_next)

#define SLIST_FOREACH(var, head, field)     \
 for((var) = (head)->slh_first;     \
     (var) != SLIST_END(head);     \
     (var) = (var)->field.sle_next)

#define SLIST_FOREACH_SAFE(var, head, field, tvar)   \
 for ((var) = SLIST_FIRST((head));    \
     (var) != SLIST_END(head) &&     \
     ((tvar) = SLIST_NEXT((var), field), 1);   \
     (var) = (tvar))

/*
* Singly-linked List functions.
*/
```

```
#define SLIST_INIT(head) do {      \
 (head)->slh_first = SLIST_END(head);    \
} while (/*CONSTCOND*/0)

#define SLIST_INSERT_AFTER(slistelm, elm, field) do {   \
 (elm)->field.sle_next = (slistelm)->field.sle_next;  \
 (slistelm)->field.sle_next = (elm);    \
} while (/*CONSTCOND*/0)

#define SLIST_INSERT_HEAD(head, elm, field) do {   \
 (elm)->field.sle_next = (head)->slh_first;   \
 (head)->slh_first = (elm);     \
} while (/*CONSTCOND*/0)

#define SLIST_REMOVE_AFTER(slistelm, field) do {   \
 (slistelm)->field.sle_next =     \
     SLIST_NEXT(SLIST_NEXT((slistelm), field), field);  \
} while (/*CONSTCOND*/0)

#define SLIST_REMOVE_HEAD(head, field) do {    \
 (head)->slh_first = (head)->slh_first->field.sle_next;  \
} while (/*CONSTCOND*/0)

#define SLIST_REMOVE(head, elm, type, field) do {   \
 if ((head)->slh_first == (elm)) {    \
  SLIST_REMOVE_HEAD((head), field);   \
 }        \
 else {        \
  struct type *curelm = (head)->slh_first;  \
  while(curelm->field.sle_next != (elm))   \
   curelm = curelm->field.sle_next;  \
  curelm->field.sle_next =    \
      curelm->field.sle_next->field.sle_next;  \
 }        \
} while (/*CONSTCOND*/0)


/*
* List definitions.
*/
#define LIST_HEAD(name, type)      \
struct name {        \
 struct type *lh_first; /* first element */   \
}

#define LIST_HEAD_INITIALIZER(head)     \
 { NULL }
```

```
#define LIST_ENTRY(type)      \
struct {        \
 struct type *le_next; /* next element */   \
 struct type **le_prev; /* address of previous next element */ \
}

/*
* List access methods.
*/
#define LIST_FIRST(head)  ((head)->lh_first)
#define LIST_END(head)   NULL
#define LIST_EMPTY(head)  ((head)->lh_first == LIST_END(head))
#define LIST_NEXT(elm, field)  ((elm)->field.le_next)

#define LIST_FOREACH(var, head, field)     \
 for ((var) = ((head)->lh_first);    \
     (var) != LIST_END(head);     \
     (var) = ((var)->field.le_next))

#define LIST_FOREACH_SAFE(var, head, field, tvar)   \
 for ((var) = LIST_FIRST((head));    \
     (var) != LIST_END(head) &&     \
     ((tvar) = LIST_NEXT((var), field), 1);   \
     (var) = (tvar))

#define LIST_MOVE(head1, head2) do {     \
 LIST_INIT((head2));      \
 if (!LIST_EMPTY((head1))) {     \
  (head2)->lh_first = (head1)->lh_first;   \
  LIST_INIT((head1));     \
 }        \
} while (/*CONSTCOND*/0)

/*
* List functions.
*/
#if defined(QUEUEDEBUG)
#define QUEUEDEBUG_LIST_INSERT_HEAD(head, elm, field)   \
 if ((head)->lh_first &&      \
     (head)->lh_first->field.le_prev != &(head)->lh_first) \
  QUEUEDEBUG_ABORT("LIST_INSERT_HEAD %p %s:%d", (head), \
      __FILE__, __LINE__);
#define QUEUEDEBUG_LIST_OP(elm, field)     \
 if ((elm)->field.le_next &&     \
     (elm)->field.le_next->field.le_prev !=   \
     &(elm)->field.le_next)     \
  QUEUEDEBUG_ABORT("LIST_* forw %p %s:%d", (elm),  \
      __FILE__, __LINE__);    \
```

```
 if (*(elm)->field.le_prev != (elm))    \
  QUEUEDEBUG_ABORT("LIST_* back %p %s:%d", (elm),  \
      __FILE__, __LINE__);
#define QUEUEDEBUG_LIST_POSTREMOVE(elm, field)    \
 (elm)->field.le_next = (void *)1L;    \
 (elm)->field.le_prev = (void *)1L;
#else
#define QUEUEDEBUG_LIST_INSERT_HEAD(head, elm, field)
#define QUEUEDEBUG_LIST_OP(elm, field)
#define QUEUEDEBUG_LIST_POSTREMOVE(elm, field)
#endif

#define LIST_INIT(head) do {      \
 (head)->lh_first = LIST_END(head);    \
} while (/*CONSTCOND*/0)

#define LIST_INSERT_AFTER(listelm, elm, field) do {   \
 QUEUEDEBUG_LIST_OP((listelm), field)    \
 if (((elm)->field.le_next = (listelm)->field.le_next) !=  \
     LIST_END(head))      \
  (listelm)->field.le_next->field.le_prev =  \
      &(elm)->field.le_next;    \
 (listelm)->field.le_next = (elm);    \
 (elm)->field.le_prev = &(listelm)->field.le_next;  \
} while (/*CONSTCOND*/0)

#define LIST_INSERT_BEFORE(listelm, elm, field) do {   \
 QUEUEDEBUG_LIST_OP((listelm), field)    \
 (elm)->field.le_prev = (listelm)->field.le_prev;  \
 (elm)->field.le_next = (listelm);    \
 *(listelm)->field.le_prev = (elm);    \
 (listelm)->field.le_prev = &(elm)->field.le_next;  \
} while (/*CONSTCOND*/0)

#define LIST_INSERT_HEAD(head, elm, field) do {    \
 QUEUEDEBUG_LIST_INSERT_HEAD((head), (elm), field)  \
 if (((elm)->field.le_next = (head)->lh_first) != LIST_END(head))\
  (head)->lh_first->field.le_prev = &(elm)->field.le_next;\
 (head)->lh_first = (elm);     \
 (elm)->field.le_prev = &(head)->lh_first;   \
} while (/*CONSTCOND*/0)

#define LIST_REMOVE(elm, field) do {     \
 QUEUEDEBUG_LIST_OP((elm), field)    \
 if ((elm)->field.le_next != NULL)    \
  (elm)->field.le_next->field.le_prev =    \
      (elm)->field.le_prev;    \
 *(elm)->field.le_prev = (elm)->field.le_next;   \
```

```
 QUEUEDEBUG_LIST_POSTREMOVE((elm), field)   \
} while (/*CONSTCOND*/0)

#define LIST_REPLACE(elm, elm2, field) do {    \
 if (((elm2)->field.le_next = (elm)->field.le_next) != NULL) \
  (elm2)->field.le_next->field.le_prev =   \
      &(elm2)->field.le_next;    \
 (elm2)->field.le_prev = (elm)->field.le_prev;   \
 *(elm2)->field.le_prev = (elm2);    \
 QUEUEDEBUG_LIST_POSTREMOVE((elm), field)   \
} while (/*CONSTCOND*/0)

/*
* Simple queue definitions.
*/
#define SIMPLEQ_HEAD(name, type)     \
struct name {        \
 struct type *sqh_first; /* first element */   \
 struct type **sqh_last; /* addr of last next element */  \
}

#define SIMPLEQ_HEAD_INITIALIZER(head)     \
 { NULL, &(head).sqh_first }

#define SIMPLEQ_ENTRY(type)      \
struct {        \
 struct type *sqe_next; /* next element */   \
}

/*
* Simple queue access methods.
*/
#define SIMPLEQ_FIRST(head)  ((head)->sqh_first)
#define SIMPLEQ_END(head)  NULL
#define SIMPLEQ_EMPTY(head)  ((head)->sqh_first == SIMPLEQ_END(head))
#define SIMPLEQ_NEXT(elm, field) ((elm)->field.sqe_next)

#define SIMPLEQ_FOREACH(var, head, field)    \
 for ((var) = ((head)->sqh_first);    \
     (var) != SIMPLEQ_END(head);     \
     (var) = ((var)->field.sqe_next))

#define SIMPLEQ_FOREACH_SAFE(var, head, field, next)   \
 for ((var) = ((head)->sqh_first);    \
     (var) != SIMPLEQ_END(head) &&    \
     ((next = ((var)->field.sqe_next)), 1);   \
     (var) = (next))
```

```
/*
* Simple queue functions.
*/
#define SIMPLEQ_INIT(head) do {      \
 (head)->sqh_first = NULL;     \
 (head)->sqh_last = &(head)->sqh_first;    \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_INSERT_HEAD(head, elm, field) do {   \
 if (((elm)->field.sqe_next = (head)->sqh_first) == NULL) \
  (head)->sqh_last = &(elm)->field.sqe_next;  \
 (head)->sqh_first = (elm);     \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_INSERT_TAIL(head, elm, field) do {   \
 (elm)->field.sqe_next = NULL;     \
 *(head)->sqh_last = (elm);     \
 (head)->sqh_last = &(elm)->field.sqe_next;   \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_INSERT_AFTER(head, listelm, elm, field) do {  \
 if (((elm)->field.sqe_next = (listelm)->field.sqe_next) == NULL)\
  (head)->sqh_last = &(elm)->field.sqe_next;  \
 (listelm)->field.sqe_next = (elm);    \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_REMOVE_HEAD(head, field) do {    \
 if (((head)->sqh_first = (head)->sqh_first->field.sqe_next) == NULL) \
  (head)->sqh_last = &(head)->sqh_first;   \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_REMOVE_AFTER(head, elm, field) do {   \
 if (((elm)->field.sqe_next = (elm)->field.sqe_next->field.sqe_next) \
     == NULL)       \
  (head)->sqh_last = &(elm)->field.sqe_next;  \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_REMOVE(head, elm, type, field) do {   \
 if ((head)->sqh_first == (elm)) {    \
  SIMPLEQ_REMOVE_HEAD((head), field);   \
 } else {       \
  struct type *curelm = (head)->sqh_first;  \
  while (curelm->field.sqe_next != (elm))   \
   curelm = curelm->field.sqe_next;  \
  if ((curelm->field.sqe_next =    \
   curelm->field.sqe_next->field.sqe_next) == NULL) \
      (head)->sqh_last = &(curelm)->field.sqe_next; \
 }       \
```

```
} while (/*CONSTCOND*/0)

#define SIMPLEQ_CONCAT(head1, head2) do {    \
 if (!SIMPLEQ_EMPTY((head2))) {     \
  *(head1)->sqh_last = (head2)->sqh_first;  \
  (head1)->sqh_last = (head2)->sqh_last;  \
  SIMPLEQ_INIT((head2));     \
 }        \
} while (/*CONSTCOND*/0)

#define SIMPLEQ_LAST(head, type, field)     \
 (SIMPLEQ_EMPTY((head)) ?      \
  NULL :       \
         ((struct type *)(void *)    \
  ((char *)((head)->sqh_last) - offsetof(struct type, field))))

/*
* Tail queue definitions.
*/
#define _TAILQ_HEAD(name, type, qual)     \
struct name {        \
 qual type *tqh_first;  /* first element */  \
 qual type *qual *tqh_last; /* addr of last next element */ \
}
#define TAILQ_HEAD(name, type) _TAILQ_HEAD(name, struct type,)

#define TAILQ_HEAD_INITIALIZER(head)     \
 { TAILQ_END(head), &(head).tqh_first }

#define _TAILQ_ENTRY(type, qual)     \
struct {        \
 qual type *tqe_next;  /* next element */  \
 qual type *qual *tqe_prev; /* address of previous next element */\
}
#define TAILQ_ENTRY(type) _TAILQ_ENTRY(struct type,)

/*
* Tail queue access methods.
*/
#define TAILQ_FIRST(head)  ((head)->tqh_first)
#define TAILQ_END(head)   (NULL)
#define TAILQ_NEXT(elm, field)  ((elm)->field.tqe_next)
#define TAILQ_LAST(head, headname) \
 (*(((struct headname *)((head)->tqh_last))->tqh_last))
#define TAILQ_PREV(elm, headname, field) \
 (*(((struct headname *)((elm)->field.tqe_prev))->tqh_last))
#define TAILQ_EMPTY(head)  (TAILQ_FIRST(head) == TAILQ_END(head))
```

```
#define TAILQ_FOREACH(var, head, field)     \
 for ((var) = ((head)->tqh_first);    \
     (var) != TAILQ_END(head);     \
     (var) = ((var)->field.tqe_next))

#define TAILQ_FOREACH_SAFE(var, head, field, next)   \
 for ((var) = ((head)->tqh_first);    \
     (var) != TAILQ_END(head) &&     \
     ((next) = TAILQ_NEXT(var, field), 1); (var) = (next))

#define TAILQ_FOREACH_REVERSE(var, head, headname, field)  \
 for ((var) = (*(((struct headname *)((head)->tqh_last))->tqh_last));\
     (var) != TAILQ_END(head);     \
     (var) = (*(((struct headname *)((var)->field.tqe_prev))->tqh_last)))

#define TAILQ_FOREACH_REVERSE_SAFE(var, head, headname, field, prev) \
 for ((var) = TAILQ_LAST((head), headname);   \
     (var) != TAILQ_END(head) &&     \
     ((prev) = TAILQ_PREV((var), headname, field), 1); (var) = (prev))

/*
* Tail queue functions.
*/
#if defined(QUEUEDEBUG)
#define QUEUEDEBUG_TAILQ_INSERT_HEAD(head, elm, field)   \
 if ((head)->tqh_first &&     \
     (head)->tqh_first->field.tqe_prev != &(head)->tqh_first) \
  QUEUEDEBUG_ABORT("TAILQ_INSERT_HEAD %p %s:%d", (head), \
      __FILE__, __LINE__);
#define QUEUEDEBUG_TAILQ_INSERT_TAIL(head, elm, field)   \
 if (*(head)->tqh_last != NULL)     \
  QUEUEDEBUG_ABORT("TAILQ_INSERT_TAIL %p %s:%d", (head), \
      __FILE__, __LINE__);
#define QUEUEDEBUG_TAILQ_OP(elm, field)     \
 if ((elm)->field.tqe_next &&     \
     (elm)->field.tqe_next->field.tqe_prev !=   \
     &(elm)->field.tqe_next)     \
  QUEUEDEBUG_ABORT("TAILQ_* forw %p %s:%d", (elm), \
      __FILE__, __LINE__);    \
 if (*(elm)->field.tqe_prev != (elm))    \
  QUEUEDEBUG_ABORT("TAILQ_* back %p %s:%d", (elm), \
      __FILE__, __LINE__);
#define QUEUEDEBUG_TAILQ_PREREMOVE(head, elm, field)   \
 if ((elm)->field.tqe_next == NULL &&    \
     (head)->tqh_last != &(elm)->field.tqe_next)   \
  QUEUEDEBUG_ABORT("TAILQ_PREREMOVE head %p elm %p %s:%d",\
      (head), (elm), __FILE__, __LINE__);
```

```
#define QUEUEDEBUG_TAILQ_POSTREMOVE(elm, field)    \
 (elm)->field.tqe_next = (void *)1L;    \
 (elm)->field.tqe_prev = (void *)1L;
#else
#define QUEUEDEBUG_TAILQ_INSERT_HEAD(head, elm, field)
#define QUEUEDEBUG_TAILQ_INSERT_TAIL(head, elm, field)
#define QUEUEDEBUG_TAILQ_OP(elm, field)
#define QUEUEDEBUG_TAILQ_PREREMOVE(head, elm, field)
#define QUEUEDEBUG_TAILQ_POSTREMOVE(elm, field)
#endif

#define TAILQ_INIT(head) do {      \
 (head)->tqh_first = TAILQ_END(head);    \
 (head)->tqh_last = &(head)->tqh_first;    \
} while (/*CONSTCOND*/0)

#define TAILQ_INSERT_HEAD(head, elm, field) do {   \
 QUEUEDEBUG_TAILQ_INSERT_HEAD((head), (elm), field)  \
 if (((elm)->field.tqe_next = (head)->tqh_first) != TAILQ_END(head))\
  (head)->tqh_first->field.tqe_prev =   \
      &(elm)->field.tqe_next;    \
 else        \
  (head)->tqh_last = &(elm)->field.tqe_next;  \
 (head)->tqh_first = (elm);     \
 (elm)->field.tqe_prev = &(head)->tqh_first;   \
} while (/*CONSTCOND*/0)

#define TAILQ_INSERT_TAIL(head, elm, field) do {   \
 QUEUEDEBUG_TAILQ_INSERT_TAIL((head), (elm), field)  \
 (elm)->field.tqe_next = TAILQ_END(head);   \
 (elm)->field.tqe_prev = (head)->tqh_last;   \
 *(head)->tqh_last = (elm);     \
 (head)->tqh_last = &(elm)->field.tqe_next;   \
} while (/*CONSTCOND*/0)

#define TAILQ_INSERT_AFTER(head, listelm, elm, field) do {  \
 QUEUEDEBUG_TAILQ_OP((listelm), field)    \
 if (((elm)->field.tqe_next = (listelm)->field.tqe_next) !=  \
     TAILQ_END(head))      \
  (elm)->field.tqe_next->field.tqe_prev =   \
      &(elm)->field.tqe_next;    \
 else        \
  (head)->tqh_last = &(elm)->field.tqe_next;  \
 (listelm)->field.tqe_next = (elm);    \
 (elm)->field.tqe_prev = &(listelm)->field.tqe_next;  \
} while (/*CONSTCOND*/0)

#define TAILQ_INSERT_BEFORE(listelm, elm, field) do {   \
```

```
 QUEUEDEBUG_TAILQ_OP((listelm), field)    \
 (elm)->field.tqe_prev = (listelm)->field.tqe_prev;  \
 (elm)->field.tqe_next = (listelm);    \
 *(listelm)->field.tqe_prev = (elm);    \
 (listelm)->field.tqe_prev = &(elm)->field.tqe_next;  \
} while (/*CONSTCOND*/0)

#define TAILQ_REMOVE(head, elm, field) do {    \
 QUEUEDEBUG_TAILQ_PREREMOVE((head), (elm), field)  \
 QUEUEDEBUG_TAILQ_OP((elm), field)    \
 if (((elm)->field.tqe_next) != TAILQ_END(head))   \
  (elm)->field.tqe_next->field.tqe_prev =   \
      (elm)->field.tqe_prev;    \
 else        \
  (head)->tqh_last = (elm)->field.tqe_prev;  \
 *(elm)->field.tqe_prev = (elm)->field.tqe_next;   \
 QUEUEDEBUG_TAILQ_POSTREMOVE((elm), field);   \
} while (/*CONSTCOND*/0)

#define TAILQ_REPLACE(head, elm, elm2, field) do {   \
      if (((elm2)->field.tqe_next = (elm)->field.tqe_next) !=  \
     TAILQ_END(head))        \
             (elm2)->field.tqe_next->field.tqe_prev =  \
                 &(elm2)->field.tqe_next;    \
      else        \
             (head)->tqh_last = &(elm2)->field.tqe_next;  \
      (elm2)->field.tqe_prev = (elm)->field.tqe_prev;   \
      *(elm2)->field.tqe_prev = (elm2);    \
 QUEUEDEBUG_TAILQ_POSTREMOVE((elm), field);   \
} while (/*CONSTCOND*/0)

#define TAILQ_CONCAT(head1, head2, field) do {    \
 if (!TAILQ_EMPTY(head2)) {     \
  *(head1)->tqh_last = (head2)->tqh_first;  \
  (head2)->tqh_first->field.tqe_prev = (head1)->tqh_last; \
  (head1)->tqh_last = (head2)->tqh_last;   \
  TAILQ_INIT((head2));     \
 }        \
} while (/*CONSTCOND*/0)

/*
* Singly-linked Tail queue declarations.
*/
#define STAILQ_HEAD(name, type)      \
struct name {        \
 struct type *stqh_first; /* first element */  \
 struct type **stqh_last; /* addr of last next element */ \
}
```

```
#define STAILQ_HEAD_INITIALIZER(head)     \
 { NULL, &(head).stqh_first }

#define STAILQ_ENTRY(type)      \
struct {        \
 struct type *stqe_next; /* next element */   \
}

/*
* Singly-linked Tail queue access methods.
*/
#define STAILQ_FIRST(head) ((head)->stqh_first)
#define STAILQ_END(head) NULL
#define STAILQ_NEXT(elm, field) ((elm)->field.stqe_next)
#define STAILQ_EMPTY(head) (STAILQ_FIRST(head) == STAILQ_END(head))

/*
* Singly-linked Tail queue functions.
*/
#define STAILQ_INIT(head) do {      \
 (head)->stqh_first = NULL;     \
 (head)->stqh_last = &(head)->stqh_first;    \
} while (/*CONSTCOND*/0)

#define STAILQ_INSERT_HEAD(head, elm, field) do {   \
 if (((elm)->field.stqe_next = (head)->stqh_first) == NULL) \
  (head)->stqh_last = &(elm)->field.stqe_next;  \
 (head)->stqh_first = (elm);     \
} while (/*CONSTCOND*/0)

#define STAILQ_INSERT_TAIL(head, elm, field) do {   \
 (elm)->field.stqe_next = NULL;     \
 *(head)->stqh_last = (elm);     \
 (head)->stqh_last = &(elm)->field.stqe_next;   \
} while (/*CONSTCOND*/0)

#define STAILQ_INSERT_AFTER(head, listelm, elm, field) do {  \
 if (((elm)->field.stqe_next = (listelm)->field.stqe_next) == NULL)\
  (head)->stqh_last = &(elm)->field.stqe_next;  \
 (listelm)->field.stqe_next = (elm);    \
} while (/*CONSTCOND*/0)

#define STAILQ_REMOVE_HEAD(head, field) do {    \
 if (((head)->stqh_first = (head)->stqh_first->field.stqe_next) == NULL) \
  (head)->stqh_last = &(head)->stqh_first;   \
} while (/*CONSTCOND*/0)
```

```
#define STAILQ_REMOVE(head, elm, type, field) do {   \
 if ((head)->stqh_first == (elm)) {    \
  STAILQ_REMOVE_HEAD((head), field);   \
 } else {       \
  struct type *curelm = (head)->stqh_first;  \
  while (curelm->field.stqe_next != (elm))   \
   curelm = curelm->field.stqe_next;  \
  if ((curelm->field.stqe_next =    \
   curelm->field.stqe_next->field.stqe_next) == NULL) \
       (head)->stqh_last = &(curelm)->field.stqe_next; \
 }       \
} while (/*CONSTCOND*/0)

#define STAILQ_FOREACH(var, head, field)    \
 for ((var) = ((head)->stqh_first);    \
  (var);       \
  (var) = ((var)->field.stqe_next))

#define STAILQ_FOREACH_SAFE(var, head, field, tvar)   \
 for ((var) = STAILQ_FIRST((head));    \
     (var) && ((tvar) = STAILQ_NEXT((var), field), 1);  \
     (var) = (tvar))

#define STAILQ_CONCAT(head1, head2) do {    \
 if (!STAILQ_EMPTY((head2))) {     \
  *(head1)->stqh_last = (head2)->stqh_first;  \
  (head1)->stqh_last = (head2)->stqh_last;  \
  STAILQ_INIT((head2));     \
 }       \
} while (/*CONSTCOND*/0)

#define STAILQ_LAST(head, type, field)     \
 (STAILQ_EMPTY((head)) ?      \
  NULL :       \
        ((struct type *)(void *)    \
  ((char *)((head)->stqh_last) - offsetof(struct type, field))))


#ifndef _KERNEL
/*
* Circular queue definitions. Do not use. We still keep the macros
* for compatibility but because of pointer aliasing issues their use
* is discouraged!
*/

/*
* __launder_type():  We use this ugly hack to work around the the compiler
* noticing that two types may not alias each other and elide tests in code.
```

```
* We hit this in the CIRCLEQ macros when comparing 'struct name *' and
* 'struct type *' (see CIRCLEQ_HEAD()).  Modern compilers (such as GCC
* 4.8) declare these comparisons as always false, causing the code to
* not run as designed.
*
* This hack is only to be used for comparisons and thus can be fully const.
* Do not use for assignment.
*
* If we ever choose to change the ABI of the CIRCLEQ macros, we could fix
* this by changing the head/tail sentinal values, but see the note above
* this one.
*/
static __inline const void * __launder_type(const void *);
static __inline const void *
__launder_type(const void *__x)
{
 __asm __volatile("" : "+r" (__x));
 return __x;
}

#if defined(QUEUEDEBUG)
#define QUEUEDEBUG_CIRCLEQ_HEAD(head, field)    \
 if ((head)->cqh_first != CIRCLEQ_ENDC(head) &&   \
     (head)->cqh_first->field.cqe_prev != CIRCLEQ_ENDC(head)) \
  QUEUEDEBUG_ABORT("CIRCLEQ head forw %p %s:%d", (head), \
        __FILE__, __LINE__);    \
 if ((head)->cqh_last != CIRCLEQ_ENDC(head) &&   \
     (head)->cqh_last->field.cqe_next != CIRCLEQ_ENDC(head)) \
  QUEUEDEBUG_ABORT("CIRCLEQ head back %p %s:%d", (head), \
        __FILE__, __LINE__);
#define QUEUEDEBUG_CIRCLEQ_ELM(head, elm, field)   \
 if ((elm)->field.cqe_next == CIRCLEQ_ENDC(head)) {  \
  if ((head)->cqh_last != (elm))    \
   QUEUEDEBUG_ABORT("CIRCLEQ elm last %p %s:%d", \
       (elm), __FILE__, __LINE__);   \
 } else {       \
  if ((elm)->field.cqe_next->field.cqe_prev != (elm)) \
   QUEUEDEBUG_ABORT("CIRCLEQ elm forw %p %s:%d", \
       (elm), __FILE__, __LINE__);   \
 }       \
 if ((elm)->field.cqe_prev == CIRCLEQ_ENDC(head)) {  \
  if ((head)->cqh_first != (elm))    \
   QUEUEDEBUG_ABORT("CIRCLEQ elm first %p %s:%d", \
       (elm), __FILE__, __LINE__);   \
 } else {       \
  if ((elm)->field.cqe_prev->field.cqe_next != (elm)) \
   QUEUEDEBUG_ABORT("CIRCLEQ elm prev %p %s:%d", \
       (elm), __FILE__, __LINE__);   \
```

```
 }
#define QUEUEDEBUG_CIRCLEQ_POSTREMOVE(elm, field)   \
 (elm)->field.cqe_next = (void *)1L;    \
 (elm)->field.cqe_prev = (void *)1L;
#else
#define QUEUEDEBUG_CIRCLEQ_HEAD(head, field)
#define QUEUEDEBUG_CIRCLEQ_ELM(head, elm, field)
#define QUEUEDEBUG_CIRCLEQ_POSTREMOVE(elm, field)
#endif

#define CIRCLEQ_HEAD(name, type)     \
struct name {        \
 struct type *cqh_first;  /* first element */  \
 struct type *cqh_last;  /* last element */  \
}

#define CIRCLEQ_HEAD_INITIALIZER(head)     \
 { CIRCLEQ_END(&head), CIRCLEQ_END(&head) }

#define CIRCLEQ_ENTRY(type)      \
struct {        \
 struct type *cqe_next;  /* next element */  \
 struct type *cqe_prev;  /* previous element */  \
}

/*
* Circular queue functions.
*/
#define CIRCLEQ_INIT(head) do {      \
 (head)->cqh_first = CIRCLEQ_END(head);    \
 (head)->cqh_last = CIRCLEQ_END(head);    \
} while (/*CONSTCOND*/0)

#define CIRCLEQ_INSERT_AFTER(head, listelm, elm, field) do {  \
 QUEUEDEBUG_CIRCLEQ_HEAD((head), field)    \
 QUEUEDEBUG_CIRCLEQ_ELM((head), (listelm), field)  \
 (elm)->field.cqe_next = (listelm)->field.cqe_next;  \
 (elm)->field.cqe_prev = (listelm);    \
 if ((listelm)->field.cqe_next == CIRCLEQ_ENDC(head))  \
  (head)->cqh_last = (elm);    \
 else        \
  (listelm)->field.cqe_next->field.cqe_prev = (elm); \
 (listelm)->field.cqe_next = (elm);    \
} while (/*CONSTCOND*/0)

#define CIRCLEQ_INSERT_BEFORE(head, listelm, elm, field) do {  \
 QUEUEDEBUG_CIRCLEQ_HEAD((head), field)    \
 QUEUEDEBUG_CIRCLEQ_ELM((head), (listelm), field)  \
```

```
 (elm)->field.cqe_next = (listelm);    \
 (elm)->field.cqe_prev = (listelm)->field.cqe_prev;  \
 if ((listelm)->field.cqe_prev == CIRCLEQ_ENDC(head))  \
  (head)->cqh_first = (elm);    \
 else        \
  (listelm)->field.cqe_prev->field.cqe_next = (elm); \
 (listelm)->field.cqe_prev = (elm);    \
} while (/*CONSTCOND*/0)

#define CIRCLEQ_INSERT_HEAD(head, elm, field) do {   \
 QUEUEDEBUG_CIRCLEQ_HEAD((head), field)    \
 (elm)->field.cqe_next = (head)->cqh_first;   \
 (elm)->field.cqe_prev = CIRCLEQ_END(head);   \
 if ((head)->cqh_last == CIRCLEQ_ENDC(head))   \
  (head)->cqh_last = (elm);    \
 else        \
  (head)->cqh_first->field.cqe_prev = (elm);  \
 (head)->cqh_first = (elm);     \
} while (/*CONSTCOND*/0)

#define CIRCLEQ_INSERT_TAIL(head, elm, field) do {   \
 QUEUEDEBUG_CIRCLEQ_HEAD((head), field)    \
 (elm)->field.cqe_next = CIRCLEQ_END(head);   \
 (elm)->field.cqe_prev = (head)->cqh_last;   \
 if ((head)->cqh_first == CIRCLEQ_ENDC(head))   \
  (head)->cqh_first = (elm);    \
 else        \
  (head)->cqh_last->field.cqe_next = (elm);  \
 (head)->cqh_last = (elm);     \
} while (/*CONSTCOND*/0)

#define CIRCLEQ_REMOVE(head, elm, field) do {    \
 QUEUEDEBUG_CIRCLEQ_HEAD((head), field)    \
 QUEUEDEBUG_CIRCLEQ_ELM((head), (elm), field)   \
 if ((elm)->field.cqe_next == CIRCLEQ_ENDC(head))  \
  (head)->cqh_last = (elm)->field.cqe_prev;  \
 else        \
  (elm)->field.cqe_next->field.cqe_prev =   \
      (elm)->field.cqe_prev;    \
 if ((elm)->field.cqe_prev == CIRCLEQ_ENDC(head))  \
  (head)->cqh_first = (elm)->field.cqe_next;  \
 else        \
  (elm)->field.cqe_prev->field.cqe_next =   \
      (elm)->field.cqe_next;    \
 QUEUEDEBUG_CIRCLEQ_POSTREMOVE((elm), field)   \
} while (/*CONSTCOND*/0)

#define CIRCLEQ_FOREACH(var, head, field)    \
```

```
 for ((var) = ((head)->cqh_first);    \
  (var) != CIRCLEQ_ENDC(head);    \
  (var) = ((var)->field.cqe_next))

#define CIRCLEQ_FOREACH_REVERSE(var, head, field)   \
 for ((var) = ((head)->cqh_last);    \
  (var) != CIRCLEQ_ENDC(head);    \
  (var) = ((var)->field.cqe_prev))

/*
* Circular queue access methods.
*/
#define CIRCLEQ_FIRST(head)  ((head)->cqh_first)
#define CIRCLEQ_LAST(head)  ((head)->cqh_last)
/* For comparisons */
#define CIRCLEQ_ENDC(head)  (__launder_type(head))
/* For assignments */
#define CIRCLEQ_END(head)  ((void *)(head))
#define CIRCLEQ_NEXT(elm, field) ((elm)->field.cqe_next)
#define CIRCLEQ_PREV(elm, field) ((elm)->field.cqe_prev)
#define CIRCLEQ_EMPTY(head)      \
   (CIRCLEQ_FIRST(head) == CIRCLEQ_ENDC(head))

#define CIRCLEQ_LOOP_NEXT(head, elm, field)    \
 (((elm)->field.cqe_next == CIRCLEQ_ENDC(head))   \
     ? ((head)->cqh_first)     \
     : (elm->field.cqe_next))
#define CIRCLEQ_LOOP_PREV(head, elm, field)    \
 (((elm)->field.cqe_prev == CIRCLEQ_ENDC(head))   \
     ? ((head)->cqh_last)     \
     : (elm->field.cqe_prev))
#endif /* !_KERNEL */

#endif /* !_SYS_QUEUE_H_ */
/* $NetBSD: tree.h,v 1.20 2013/09/14 13:20:45 joerg Exp $ */
/* $OpenBSD: tree.h,v 1.13 2011/07/09 00:19:45 pirofti Exp $ */
/*
* Copyright 2002 Niels Provos <provos@citi.umich.edu>
* 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.
```

```
*
* 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.
*/

#ifndef _SYS_TREE_H_
#define _SYS_TREE_H_

/*
* This file defines data structures for different types of trees:
* splay trees and red-black trees.
*
* A splay tree is a self-organizing data structure.  Every operation
* on the tree causes a splay to happen.  The splay moves the requested
* node to the root of the tree and partly rebalances it.
*
* This has the benefit that request locality causes faster lookups as
* the requested nodes move to the top of the tree.  On the other hand,
* every lookup causes memory writes.
*
* The Balance Theorem bounds the total access time for m operations
* and n inserts on an initially empty tree as O((m + n)lg n).  The
* amortized cost for a sequence of m accesses to a splay tree is O(lg n);
*
* A red-black tree is a binary search tree with the node color as an
* extra attribute.  It fulfills a set of conditions:
* - every search path from the root to a leaf consists of the
*   same number of black nodes,
* - each red node (except for the root) has a black parent,
* - each leaf node is black.
*
* Every operation on a red-black tree is bounded as O(lg n).
* The maximum height of a red-black tree is 2lg (n+1).
*/

#define SPLAY_HEAD(name, type)      \
struct name {        \
 struct type *sph_root; /* root of the tree */   \
}
```

```
#define SPLAY_INITIALIZER(root)      \
 { NULL }

#define SPLAY_INIT(root) do {      \
 (root)->sph_root = NULL;     \
} while (/*CONSTCOND*/ 0)

#define SPLAY_ENTRY(type)      \
struct {        \
 struct type *spe_left; /* left element */   \
 struct type *spe_right; /* right element */   \
}

#define SPLAY_LEFT(elm, field)  (elm)->field.spe_left
#define SPLAY_RIGHT(elm, field)  (elm)->field.spe_right
#define SPLAY_ROOT(head)  (head)->sph_root
#define SPLAY_EMPTY(head)  (SPLAY_ROOT(head) == NULL)

/* SPLAY_ROTATE_{LEFT,RIGHT} expect that tmp hold SPLAY_{RIGHT,LEFT} */
#define SPLAY_ROTATE_RIGHT(head, tmp, field) do {   \
 SPLAY_LEFT((head)->sph_root, field) = SPLAY_RIGHT(tmp, field); \
 SPLAY_RIGHT(tmp, field) = (head)->sph_root;   \
 (head)->sph_root = tmp;      \
} while (/*CONSTCOND*/ 0)

#define SPLAY_ROTATE_LEFT(head, tmp, field) do {   \
 SPLAY_RIGHT((head)->sph_root, field) = SPLAY_LEFT(tmp, field); \
 SPLAY_LEFT(tmp, field) = (head)->sph_root;   \
 (head)->sph_root = tmp;      \
} while (/*CONSTCOND*/ 0)

#define SPLAY_LINKLEFT(head, tmp, field) do {    \
 SPLAY_LEFT(tmp, field) = (head)->sph_root;   \
 tmp = (head)->sph_root;      \
 (head)->sph_root = SPLAY_LEFT((head)->sph_root, field);  \
} while (/*CONSTCOND*/ 0)

#define SPLAY_LINKRIGHT(head, tmp, field) do {    \
 SPLAY_RIGHT(tmp, field) = (head)->sph_root;   \
 tmp = (head)->sph_root;      \
 (head)->sph_root = SPLAY_RIGHT((head)->sph_root, field); \
} while (/*CONSTCOND*/ 0)

#define SPLAY_ASSEMBLE(head, node, left, right, field) do {  \
 SPLAY_RIGHT(left, field) = SPLAY_LEFT((head)->sph_root, field); \
 SPLAY_LEFT(right, field) = SPLAY_RIGHT((head)->sph_root, field);\
 SPLAY_LEFT((head)->sph_root, field) = SPLAY_RIGHT(node, field); \
 SPLAY_RIGHT((head)->sph_root, field) = SPLAY_LEFT(node, field); \
```

```
} while (/*CONSTCOND*/ 0)

/* Generates prototypes and inline functions */

#define SPLAY_PROTOTYPE(name, type, field, cmp)    \
void name##_SPLAY(struct name *, struct type *);   \
void name##_SPLAY_MINMAX(struct name *, int);    \
struct type *name##_SPLAY_INSERT(struct name *, struct type *);  \
struct type *name##_SPLAY_REMOVE(struct name *, struct type *);  \
         \
/* Finds the node with the same key as elm */    \
static __inline struct type *      \
name##_SPLAY_FIND(struct name *head, struct type *elm)   \
{        \
 if (SPLAY_EMPTY(head))      \
  return(NULL);      \
 name##_SPLAY(head, elm);     \
 if ((cmp)(elm, (head)->sph_root) == 0)    \
  return (head->sph_root);    \
 return (NULL);       \
}        \
         \
static __inline __unused struct type *     \
name##_SPLAY_NEXT(struct name *head, struct type *elm)   \
{        \
 name##_SPLAY(head, elm);     \
 if (SPLAY_RIGHT(elm, field) != NULL) {    \
  elm = SPLAY_RIGHT(elm, field);    \
  while (SPLAY_LEFT(elm, field) != NULL) {  \
   elm = SPLAY_LEFT(elm, field);   \
  }      \
 } else       \
  elm = NULL;      \
 return (elm);       \
}        \
         \
static __unused __inline struct type *     \
name##_SPLAY_MIN_MAX(struct name *head, int val)   \
{        \
 name##_SPLAY_MINMAX(head, val);     \
       return (SPLAY_ROOT(head));     \
}

/* Main splay operation.
* Moves node close to the key of elm to top
*/
#define SPLAY_GENERATE(name, type, field, cmp)    \
struct type *       \
```

```
name##_SPLAY_INSERT(struct name *head, struct type *elm)  \
{          \
   if (SPLAY_EMPTY(head)) {       \
     SPLAY_LEFT(elm, field) = SPLAY_RIGHT(elm, field) = NULL; \
   } else {          \
     int __comp;        \
     name##_SPLAY(head, elm);      \
     __comp = (cmp)(elm, (head)->sph_root);   \
     if(__comp < 0) {       \
      SPLAY_LEFT(elm, field) = SPLAY_LEFT((head)->sph_root, field);\
      SPLAY_RIGHT(elm, field) = (head)->sph_root;  \
      SPLAY_LEFT((head)->sph_root, field) = NULL;  \
     } else if (__comp > 0) {      \
      SPLAY_RIGHT(elm, field) = SPLAY_RIGHT((head)->sph_root, field);\
      SPLAY_LEFT(elm, field) = (head)->sph_root;  \
      SPLAY_RIGHT((head)->sph_root, field) = NULL; \
     } else        \
      return ((head)->sph_root);    \
   }          \
   (head)->sph_root = (elm);       \
   return (NULL);        \
}          \
         \
struct type *         \
name##_SPLAY_REMOVE(struct name *head, struct type *elm)  \
{          \
 struct type *__tmp;       \
 if (SPLAY_EMPTY(head))       \
  return (NULL);       \
 name##_SPLAY(head, elm);      \
 if ((cmp)(elm, (head)->sph_root) == 0) {    \
  if (SPLAY_LEFT((head)->sph_root, field) == NULL) { \
   (head)->sph_root = SPLAY_RIGHT((head)->sph_root, field);\
  } else {       \
   __tmp = SPLAY_RIGHT((head)->sph_root, field); \
   (head)->sph_root = SPLAY_LEFT((head)->sph_root, field);\
   name##_SPLAY(head, elm);    \
   SPLAY_RIGHT((head)->sph_root, field) = __tmp; \
  }        \
  return (elm);       \
 }         \
 return (NULL);        \
}          \
         \
void          \
name##_SPLAY(struct name *head, struct type *elm)   \
{          \
 struct type __node, *__left, *__right, *__tmp;   \
```

```
 int __comp;       \
\
 SPLAY_LEFT(&__node, field) = SPLAY_RIGHT(&__node, field) = NULL;\
 __left = __right = &__node;     \
\
 while ((__comp = (cmp)(elm, (head)->sph_root)) != 0) {  \
  if (__comp < 0) {     \
   __tmp = SPLAY_LEFT((head)->sph_root, field); \
   if (__tmp == NULL)    \
    break;     \
   if ((cmp)(elm, __tmp) < 0){   \
    SPLAY_ROTATE_RIGHT(head, __tmp, field); \
    if (SPLAY_LEFT((head)->sph_root, field) == NULL)\
     break;    \
   }      \
   SPLAY_LINKLEFT(head, __right, field);  \
  } else if (__comp > 0) {    \
   __tmp = SPLAY_RIGHT((head)->sph_root, field); \
   if (__tmp == NULL)    \
    break;     \
   if ((cmp)(elm, __tmp) > 0){   \
    SPLAY_ROTATE_LEFT(head, __tmp, field); \
    if (SPLAY_RIGHT((head)->sph_root, field) == NULL)\
     break;    \
   }      \
   SPLAY_LINKRIGHT(head, __left, field);  \
  }       \
 }       \
 SPLAY_ASSEMBLE(head, &__node, __left, __right, field);  \
}        \
        \
/* Splay with either the minimum or the maximum element   \
* Used to find minimum or maximum element in tree.   \
*/        \
void name##_SPLAY_MINMAX(struct name *head, int __comp) \
{        \
 struct type __node, *__left, *__right, *__tmp;   \
\
 SPLAY_LEFT(&__node, field) = SPLAY_RIGHT(&__node, field) = NULL;\
 __left = __right = &__node;     \
\
 while (1) {       \
  if (__comp < 0) {     \
   __tmp = SPLAY_LEFT((head)->sph_root, field); \
   if (__tmp == NULL)    \
    break;     \
   if (__comp < 0){    \
    SPLAY_ROTATE_RIGHT(head, __tmp, field); \
```

```
   if (SPLAY_LEFT((head)->sph_root, field) == NULL)\
    break;    \
   }      \
   SPLAY_LINKLEFT(head, __right, field);  \
  } else if (__comp > 0) {    \
   __tmp = SPLAY_RIGHT((head)->sph_root, field); \
   if (__tmp == NULL)    \
    break;     \
   if (__comp > 0) {    \
    SPLAY_ROTATE_LEFT(head, __tmp, field); \
    if (SPLAY_RIGHT((head)->sph_root, field) == NULL)\
     break;    \
   }      \
   SPLAY_LINKRIGHT(head, __left, field);  \
  }       \
 }       \
 SPLAY_ASSEMBLE(head, &__node, __left, __right, field);  \
}

#define SPLAY_NEGINF -1
#define SPLAY_INF 1

#define SPLAY_INSERT(name, x, y) name##_SPLAY_INSERT(x, y)
#define SPLAY_REMOVE(name, x, y) name##_SPLAY_REMOVE(x, y)
#define SPLAY_FIND(name, x, y)  name##_SPLAY_FIND(x, y)
#define SPLAY_NEXT(name, x, y)  name##_SPLAY_NEXT(x, y)
#define SPLAY_MIN(name, x)  (SPLAY_EMPTY(x) ? NULL \
     : name##_SPLAY_MIN_MAX(x, SPLAY_NEGINF))
#define SPLAY_MAX(name, x)  (SPLAY_EMPTY(x) ? NULL \
     : name##_SPLAY_MIN_MAX(x, SPLAY_INF))

#define SPLAY_FOREACH(x, name, head)     \
 for ((x) = SPLAY_MIN(name, head);    \
      (x) != NULL;      \
      (x) = SPLAY_NEXT(name, head, x))

/* Macros that define a red-black tree */
#define RB_HEAD(name, type)      \
struct name {        \
 struct type *rbh_root; /* root of the tree */   \
}

#define RB_INITIALIZER(root)      \
 { NULL }

#define RB_INIT(root) do {      \
 (root)->rbh_root = NULL;     \
} while (/*CONSTCOND*/ 0)
```

```
#define RB_BLACK 0
#define RB_RED  1
#define RB_ENTRY(type)       \
struct {        \
 struct type *rbe_left;  /* left element */  \
 struct type *rbe_right;  /* right element */  \
 struct type *rbe_parent; /* parent element */  \
 int rbe_color;   /* node color */  \
}

#define RB_LEFT(elm, field)  (elm)->field.rbe_left
#define RB_RIGHT(elm, field)  (elm)->field.rbe_right
#define RB_PARENT(elm, field)  (elm)->field.rbe_parent
#define RB_COLOR(elm, field)  (elm)->field.rbe_color
#define RB_ROOT(head)   (head)->rbh_root
#define RB_EMPTY(head)   (RB_ROOT(head) == NULL)

#define RB_SET(elm, parent, field) do {     \
 RB_PARENT(elm, field) = parent;     \
 RB_LEFT(elm, field) = RB_RIGHT(elm, field) = NULL;  \
 RB_COLOR(elm, field) = RB_RED;     \
} while (/*CONSTCOND*/ 0)

#define RB_SET_BLACKRED(black, red, field) do {    \
 RB_COLOR(black, field) = RB_BLACK;    \
 RB_COLOR(red, field) = RB_RED;     \
} while (/*CONSTCOND*/ 0)

#ifndef RB_AUGMENT
#define RB_AUGMENT(x) do {} while (/*CONSTCOND*/ 0)
#endif

#define RB_ROTATE_LEFT(head, elm, tmp, field) do {   \
 (tmp) = RB_RIGHT(elm, field);     \
 if ((RB_RIGHT(elm, field) = RB_LEFT(tmp, field)) != NULL) { \
  RB_PARENT(RB_LEFT(tmp, field), field) = (elm);  \
 }        \
 RB_AUGMENT(elm);      \
 if ((RB_PARENT(tmp, field) = RB_PARENT(elm, field)) != NULL) { \
  if ((elm) == RB_LEFT(RB_PARENT(elm, field), field)) \
   RB_LEFT(RB_PARENT(elm, field), field) = (tmp); \
  else       \
   RB_RIGHT(RB_PARENT(elm, field), field) = (tmp); \
 } else        \
  (head)->rbh_root = (tmp);    \
 RB_LEFT(tmp, field) = (elm);     \
 RB_PARENT(elm, field) = (tmp);     \
```

```
 RB_AUGMENT(tmp);      \
 if ((RB_PARENT(tmp, field)))     \
  RB_AUGMENT(RB_PARENT(tmp, field));   \
} while (/*CONSTCOND*/ 0)

#define RB_ROTATE_RIGHT(head, elm, tmp, field) do {   \
 (tmp) = RB_LEFT(elm, field);     \
 if ((RB_LEFT(elm, field) = RB_RIGHT(tmp, field)) != NULL) { \
  RB_PARENT(RB_RIGHT(tmp, field), field) = (elm);  \
 }        \
 RB_AUGMENT(elm);      \
 if ((RB_PARENT(tmp, field) = RB_PARENT(elm, field)) != NULL) { \
  if ((elm) == RB_LEFT(RB_PARENT(elm, field), field)) \
   RB_LEFT(RB_PARENT(elm, field), field) = (tmp); \
  else       \
   RB_RIGHT(RB_PARENT(elm, field), field) = (tmp); \
 } else        \
  (head)->rbh_root = (tmp);    \
 RB_RIGHT(tmp, field) = (elm);     \
 RB_PARENT(elm, field) = (tmp);     \
 RB_AUGMENT(tmp);      \
 if ((RB_PARENT(tmp, field)))     \
  RB_AUGMENT(RB_PARENT(tmp, field));   \
} while (/*CONSTCOND*/ 0)

/* Generates prototypes and inline functions */
#define RB_PROTOTYPE(name, type, field, cmp)    \
 RB_PROTOTYPE_INTERNAL(name, type, field, cmp,)
#define RB_PROTOTYPE_STATIC(name, type, field, cmp)   \
 RB_PROTOTYPE_INTERNAL(name, type, field, cmp, __unused static)
#define RB_PROTOTYPE_INTERNAL(name, type, field, cmp, attr)  \
attr void name##_RB_INSERT_COLOR(struct name *, struct type *);  \
attr void name##_RB_REMOVE_COLOR(struct name *, struct type *, struct type *);\
attr struct type *name##_RB_REMOVE(struct name *, struct type *); \
attr struct type *name##_RB_INSERT(struct name *, struct type *); \
attr struct type *name##_RB_FIND(struct name *, struct type *);  \
attr struct type *name##_RB_NFIND(struct name *, struct type *); \
attr struct type *name##_RB_NEXT(struct type *);   \
attr struct type *name##_RB_PREV(struct type *);   \
attr struct type *name##_RB_MINMAX(struct name *, int);   \
        \

/* Main rb operation.
* Moves node close to the key of elm to top
*/
#define RB_GENERATE(name, type, field, cmp)    \
 RB_GENERATE_INTERNAL(name, type, field, cmp,)
#define RB_GENERATE_STATIC(name, type, field, cmp)   \
```

```
 RB_GENERATE_INTERNAL(name, type, field, cmp, __unused static)
#define RB_GENERATE_INTERNAL(name, type, field, cmp, attr)  \
attr void        \
name##_RB_INSERT_COLOR(struct name *head, struct type *elm)  \
{        \
 struct type *parent, *gparent, *tmp;    \
 while ((parent = RB_PARENT(elm, field)) != NULL &&  \
     RB_COLOR(parent, field) == RB_RED) {   \
  gparent = RB_PARENT(parent, field);   \
  if (parent == RB_LEFT(gparent, field)) {  \
   tmp = RB_RIGHT(gparent, field);   \
   if (tmp && RB_COLOR(tmp, field) == RB_RED) { \
    RB_COLOR(tmp, field) = RB_BLACK; \
    RB_SET_BLACKRED(parent, gparent, field);\
    elm = gparent;    \
    continue;    \
   }      \
   if (RB_RIGHT(parent, field) == elm) {  \
    RB_ROTATE_LEFT(head, parent, tmp, field);\
    tmp = parent;    \
    parent = elm;    \
    elm = tmp;    \
   }      \
   RB_SET_BLACKRED(parent, gparent, field); \
   RB_ROTATE_RIGHT(head, gparent, tmp, field); \
  } else {      \
   tmp = RB_LEFT(gparent, field);   \
   if (tmp && RB_COLOR(tmp, field) == RB_RED) { \
    RB_COLOR(tmp, field) = RB_BLACK; \
    RB_SET_BLACKRED(parent, gparent, field);\
    elm = gparent;    \
    continue;    \
   }      \
   if (RB_LEFT(parent, field) == elm) {  \
    RB_ROTATE_RIGHT(head, parent, tmp, field);\
    tmp = parent;    \
    parent = elm;    \
    elm = tmp;    \
   }      \
   RB_SET_BLACKRED(parent, gparent, field); \
   RB_ROTATE_LEFT(head, gparent, tmp, field); \
  }      \
 }       \
 RB_COLOR(head->rbh_root, field) = RB_BLACK;   \
}        \
        \
attr void        \
name##_RB_REMOVE_COLOR(struct name *head, struct type *parent, struct type *elm) \
```

```
{ \
 struct type *tmp; \
 while ((elm == NULL || RB_COLOR(elm, field) == RB_BLACK) && \
     elm != RB_ROOT(head)) { \
  if (RB_LEFT(parent, field) == elm) { \
   tmp = RB_RIGHT(parent, field); \
   if (RB_COLOR(tmp, field) == RB_RED) { \
    RB_SET_BLACKRED(tmp, parent, field); \
    RB_ROTATE_LEFT(head, parent, tmp, field);\
    tmp = RB_RIGHT(parent, field); \
   } \
   if ((RB_LEFT(tmp, field) == NULL || \
       RB_COLOR(RB_LEFT(tmp, field), field) == RB_BLACK) &&\
       (RB_RIGHT(tmp, field) == NULL || \
       RB_COLOR(RB_RIGHT(tmp, field), field) == RB_BLACK)) {\
    RB_COLOR(tmp, field) = RB_RED; \
    elm = parent; \
    parent = RB_PARENT(elm, field); \
   } else { \
    if (RB_RIGHT(tmp, field) == NULL || \
        RB_COLOR(RB_RIGHT(tmp, field), field) == RB_BLACK) {\
     struct type *oleft; \
     if ((oleft = RB_LEFT(tmp, field)) \
         != NULL) \
      RB_COLOR(oleft, field) = RB_BLACK;\
     RB_COLOR(tmp, field) = RB_RED; \
     RB_ROTATE_RIGHT(head, tmp, oleft, field);\
     tmp = RB_RIGHT(parent, field); \
    } \
    RB_COLOR(tmp, field) = RB_COLOR(parent, field);\
    RB_COLOR(parent, field) = RB_BLACK; \
    if (RB_RIGHT(tmp, field)) \
     RB_COLOR(RB_RIGHT(tmp, field), field) = RB_BLACK;\
    RB_ROTATE_LEFT(head, parent, tmp, field);\
    elm = RB_ROOT(head); \
    break; \
   } \
  } else { \
   tmp = RB_LEFT(parent, field); \
   if (RB_COLOR(tmp, field) == RB_RED) { \
    RB_SET_BLACKRED(tmp, parent, field); \
    RB_ROTATE_RIGHT(head, parent, tmp, field);\
    tmp = RB_LEFT(parent, field); \
   } \
   if ((RB_LEFT(tmp, field) == NULL || \
       RB_COLOR(RB_LEFT(tmp, field), field) == RB_BLACK) &&\
       (RB_RIGHT(tmp, field) == NULL || \
       RB_COLOR(RB_RIGHT(tmp, field), field) == RB_BLACK)) {\
```

```
   RB_COLOR(tmp, field) = RB_RED;  \
   elm = parent;    \
   parent = RB_PARENT(elm, field);  \
  } else {     \
   if (RB_LEFT(tmp, field) == NULL || \
       RB_COLOR(RB_LEFT(tmp, field), field) == RB_BLACK) {\
    struct type *oright;  \
    if ((oright = RB_RIGHT(tmp, field)) \
        != NULL)   \
     RB_COLOR(oright, field) = RB_BLACK;\
    RB_COLOR(tmp, field) = RB_RED; \
    RB_ROTATE_LEFT(head, tmp, oright, field);\
    tmp = RB_LEFT(parent, field); \
   }     \
   RB_COLOR(tmp, field) = RB_COLOR(parent, field);\
   RB_COLOR(parent, field) = RB_BLACK; \
   if (RB_LEFT(tmp, field))  \
    RB_COLOR(RB_LEFT(tmp, field), field) = RB_BLACK;\
   RB_ROTATE_RIGHT(head, parent, tmp, field);\
   elm = RB_ROOT(head);   \
   break;     \
  }      \
 }       \
 }       \
 if (elm)       \
  RB_COLOR(elm, field) = RB_BLACK;   \
}        \
         \
attr struct type *       \
name##_RB_REMOVE(struct name *head, struct type *elm)   \
{        \
 struct type *child, *parent, *old = elm;   \
 int color;       \
 if (RB_LEFT(elm, field) == NULL)    \
  child = RB_RIGHT(elm, field);    \
 else if (RB_RIGHT(elm, field) == NULL)    \
  child = RB_LEFT(elm, field);    \
 else {        \
  struct type *left;     \
  elm = RB_RIGHT(elm, field);    \
  while ((left = RB_LEFT(elm, field)) != NULL)  \
   elm = left;     \
  child = RB_RIGHT(elm, field);    \
  parent = RB_PARENT(elm, field);    \
  color = RB_COLOR(elm, field);    \
  if (child)      \
   RB_PARENT(child, field) = parent;  \
  if (parent) {      \
```

```
   if (RB_LEFT(parent, field) == elm)  \
    RB_LEFT(parent, field) = child;  \
   else      \
    RB_RIGHT(parent, field) = child; \
   RB_AUGMENT(parent);    \
  } else       \
   RB_ROOT(head) = child;    \
  if (RB_PARENT(elm, field) == old)   \
   parent = elm;     \
  (elm)->field = (old)->field;    \
  if (RB_PARENT(old, field)) {    \
   if (RB_LEFT(RB_PARENT(old, field), field) == old)\
    RB_LEFT(RB_PARENT(old, field), field) = elm;\
   else      \
    RB_RIGHT(RB_PARENT(old, field), field) = elm;\
   RB_AUGMENT(RB_PARENT(old, field));  \
  } else       \
   RB_ROOT(head) = elm;    \
  RB_PARENT(RB_LEFT(old, field), field) = elm;  \
  if (RB_RIGHT(old, field))    \
   RB_PARENT(RB_RIGHT(old, field), field) = elm; \
  if (parent) {      \
   left = parent;     \
   do {      \
    RB_AUGMENT(left);   \
   } while ((left = RB_PARENT(left, field)) != NULL); \
  }       \
  goto color;      \
 }       \
 parent = RB_PARENT(elm, field);     \
 color = RB_COLOR(elm, field);     \
 if (child)       \
  RB_PARENT(child, field) = parent;   \
 if (parent) {       \
  if (RB_LEFT(parent, field) == elm)   \
   RB_LEFT(parent, field) = child;   \
  else       \
   RB_RIGHT(parent, field) = child;  \
  RB_AUGMENT(parent);     \
 } else        \
  RB_ROOT(head) = child;     \
color:        \
 if (color == RB_BLACK)      \
  name##_RB_REMOVE_COLOR(head, parent, child);  \
 return (old);       \
}        \
        \
/* Inserts a node into the RB tree */     \
```

```
attr struct type *	\
name##_RB_INSERT(struct name *head, struct type *elm)	\
{	\
 struct type *tmp;	\
 struct type *parent = NULL;	\
 int comp = 0;	\
 tmp = RB_ROOT(head);	\
 while (tmp) {	\
  parent = tmp;	\
  comp = (cmp)(elm, parent);	\
  if (comp < 0)	\
   tmp = RB_LEFT(tmp, field);	\
  else if (comp > 0)	\
   tmp = RB_RIGHT(tmp, field);	\
  else	\
   return (tmp);	\
 }	\
 RB_SET(elm, parent, field);	\
 if (parent != NULL) {	\
  if (comp < 0)	\
   RB_LEFT(parent, field) = elm;	\
  else	\
   RB_RIGHT(parent, field) = elm;	\
  RB_AUGMENT(parent);	\
 } else	\
  RB_ROOT(head) = elm;	\
 name##_RB_INSERT_COLOR(head, elm);	\
 return (NULL);	\
}	\
	\
/* Finds the node with the same key as elm */	\
attr struct type *	\
name##_RB_FIND(struct name *head, struct type *elm)	\
{	\
 struct type *tmp = RB_ROOT(head);	\
 int comp;	\
 while (tmp) {	\
  comp = cmp(elm, tmp);	\
  if (comp < 0)	\
   tmp = RB_LEFT(tmp, field);	\
  else if (comp > 0)	\
   tmp = RB_RIGHT(tmp, field);	\
  else	\
   return (tmp);	\
 }	\
 return (NULL);	\
}	\
	\
```

```
/* Finds the first node greater than or equal to the search key */ \
attr struct type *       \
name##_RB_NFIND(struct name *head, struct type *elm)   \
{         \
 struct type *tmp = RB_ROOT(head);    \
 struct type *res = NULL;     \
 int comp;       \
 while (tmp) {       \
  comp = cmp(elm, tmp);     \
  if (comp < 0) {      \
   res = tmp;     \
   tmp = RB_LEFT(tmp, field);   \
  }       \
  else if (comp > 0)     \
   tmp = RB_RIGHT(tmp, field);   \
  else       \
   return (tmp);     \
 }        \
 return (res);       \
}         \
        \
/* ARGSUSED */        \
attr struct type *       \
name##_RB_NEXT(struct type *elm)     \
{         \
 if (RB_RIGHT(elm, field)) {     \
  elm = RB_RIGHT(elm, field);    \
  while (RB_LEFT(elm, field))    \
   elm = RB_LEFT(elm, field);   \
 } else {       \
  if (RB_PARENT(elm, field) &&    \
      (elm == RB_LEFT(RB_PARENT(elm, field), field))) \
   elm = RB_PARENT(elm, field);   \
  else {       \
   while (RB_PARENT(elm, field) &&   \
       (elm == RB_RIGHT(RB_PARENT(elm, field), field)))\
    elm = RB_PARENT(elm, field);  \
   elm = RB_PARENT(elm, field);   \
  }       \
 }       \
 return (elm);       \
}         \
        \
/* ARGSUSED */        \
attr struct type *       \
name##_RB_PREV(struct type *elm)     \
{         \
 if (RB_LEFT(elm, field)) {     \
```

```
  elm = RB_LEFT(elm, field);    \
  while (RB_RIGHT(elm, field))    \
   elm = RB_RIGHT(elm, field);   \
 } else {       \
  if (RB_PARENT(elm, field) &&    \
      (elm == RB_RIGHT(RB_PARENT(elm, field), field))) \
   elm = RB_PARENT(elm, field);   \
  else {       \
   while (RB_PARENT(elm, field) &&   \
       (elm == RB_LEFT(RB_PARENT(elm, field), field)))\
    elm = RB_PARENT(elm, field);  \
   elm = RB_PARENT(elm, field);   \
  }       \
 }       \
 return (elm);       \
}         \
         \
attr struct type *       \
name##_RB_MINMAX(struct name *head, int val)    \
{         \
 struct type *tmp = RB_ROOT(head);    \
 struct type *parent = NULL;     \
 while (tmp) {       \
  parent = tmp;      \
  if (val < 0)      \
   tmp = RB_LEFT(tmp, field);   \
  else       \
   tmp = RB_RIGHT(tmp, field);   \
 }       \
 return (parent);      \
}

#define RB_NEGINF -1
#define RB_INF 1

#define RB_INSERT(name, x, y) name##_RB_INSERT(x, y)
#define RB_REMOVE(name, x, y) name##_RB_REMOVE(x, y)
#define RB_FIND(name, x, y) name##_RB_FIND(x, y)
#define RB_NFIND(name, x, y) name##_RB_NFIND(x, y)
#define RB_NEXT(name, x, y) name##_RB_NEXT(y)
#define RB_PREV(name, x, y) name##_RB_PREV(y)
#define RB_MIN(name, x)  name##_RB_MINMAX(x, RB_NEGINF)
#define RB_MAX(name, x)  name##_RB_MINMAX(x, RB_INF)

#define RB_FOREACH(x, name, head)     \
 for ((x) = RB_MIN(name, head);     \
      (x) != NULL;      \
      (x) = name##_RB_NEXT(x))
```

```
#define RB_FOREACH_FROM(x, name, y)     \
 for ((x) = (y);       \
     ((x) != NULL) && ((y) = name##_RB_NEXT(x), (x) != NULL); \
      (x) = (y))

#define RB_FOREACH_SAFE(x, name, head, y)    \
 for ((x) = RB_MIN(name, head);     \
     ((x) != NULL) && ((y) = name##_RB_NEXT(x), (x) != NULL); \
      (x) = (y))

#define RB_FOREACH_REVERSE(x, name, head)    \
 for ((x) = RB_MAX(name, head);     \
      (x) != NULL;      \
      (x) = name##_RB_PREV(x))

#define RB_FOREACH_REVERSE_FROM(x, name, y)    \
 for ((x) = (y);       \
     ((x) != NULL) && ((y) = name##_RB_PREV(x), (x) != NULL); \
      (x) = (y))

#define RB_FOREACH_REVERSE_SAFE(x, name, head, y)   \
 for ((x) = RB_MAX(name, head);     \
     ((x) != NULL) && ((y) = name##_RB_PREV(x), (x) != NULL); \
      (x) = (y))

#endif /* _SYS_TREE_H_ */
```

1.8 libcrypto 1.0-1.0.2j :r0

1.8.1 Available under license :

Copyright (C) 1995-1997 Eric Young (eay@cryptsoft.com)
All rights reserved.

This package is an DES implementation written by Eric Young (eay@cryptsoft.com).
The implementation was written so as to conform with MIT's libdes.

This library is free for commercial and non-commercial use as long as
the following conditions are aheared to. The following conditions
apply to all code found in this distribution.

Copyright remains Eric Young's, and as such any Copyright notices in
the code are not to be removed.
If this package is used in a product, Eric Young should be given attribution
as the author of that the SSL library. This can be in the form of a textual
message at program startup or in documentation (online or textual) provided
with the package.

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 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. All advertising materials mentioning features or use of this software
   must display the following acknowledgement:
   This product includes software developed by Eric Young (eay@cryptsoft.com)

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``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.

The license and distribution terms for any publically available version or
derivative of this code cannot be changed. i.e. this code cannot simply be
copied and put under another distrubution license
[including the GNU Public License.]

The reason behind this being stated in this direct manner is past
experience in code simply being copied and the attribution removed
from it and then being distributed as part of other packages. This
implementation was a non-trivial and unpaid effort.
Copyright (C) 1995-1997 Eric Young (eay@cryptsoft.com)
All rights reserved.

This package is an Blowfish implementation written
by Eric Young (eay@cryptsoft.com).

This library is free for commercial and non-commercial use as long as
the following conditions are aheared to. The following conditions
apply to all code found in this distribution.

Copyright remains Eric Young's, and as such any Copyright notices in
the code are not to be removed.

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 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. All advertising materials mentioning features or use of this software
must display the following acknowledgement:
This product includes software developed by Eric Young (eay@cryptsoft.com)

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``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.

The license and distribution terms for any publically available version or
derivative of this code cannot be changed. i.e. this code cannot simply be
copied and put under another distrubution license
[including the GNU Public License.]

The reason behind this being stated in this direct manner is past
experience in code simply being copied and the attribution removed
from it and then being distributed as part of other packages. This
implementation was a non-trivial and unpaid effort.

LICENSE ISSUES
==============

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of
the OpenSSL License and the original SSLeay license apply to the toolkit.
See below for the actual license texts. Actually both licenses are BSD-style
Open Source licenses. In case of any license issues related to OpenSSL
please contact openssl-core@openssl.org.

OpenSSL License

/* ==
* Copyright (c) 1998-2016 The OpenSSL Project. All rights reserved.
*

*
*/

Original SSLeay License

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
* All rights reserved.
*
* This package is an SSL implementation written
* by Eric Young (eay@cryptsoft.com).
* The implementation was written so as to conform with Netscapes SSL.
*
* This library is free for commercial and non-commercial use as long as
* the following conditions are aheared to. The following conditions
* apply to all code found in this distribution, be it the RC4, RSA,
* lhash, DES, etc., code; not just the SSL code. The SSL documentation
* included with this distribution is covered by the same copyright terms
* except that the holder is Tim Hudson (tjh@cryptsoft.com).
*
* Copyright remains Eric Young's, and as such any Copyright notices in
* the code are not to be removed.
* If this package is used in a product, Eric Young should be given attribution
* as the author of the parts of the library used.
* This can be in the form of a textual message at program startup or
* in documentation (online or textual) provided with the package.
*
* 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 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. All advertising materials mentioning features or use of this software
*   must display the following acknowledgement:
*   "This product includes cryptographic software written by
*   Eric Young (eay@cryptsoft.com)"
*   The word 'cryptographic' can be left out if the rouines from the library
*   being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
*   the apps directory (application code) you must include an acknowledgement:
*   "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``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.
*
* The licence and distribution terms for any publically available version or
* derivative of this code cannot be changed. i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/

1.9 libssl 1.0-1.0.2j :r0

1.9.1 Available under license :

Copyright (C) 1995-1997 Eric Young (eay@cryptsoft.com)
All rights reserved.

This package is an DES implementation written by Eric Young (eay@cryptsoft.com).
The implementation was written so as to conform with MIT's libdes.

This library is free for commercial and non-commercial use as long as
the following conditions are aheared to. The following conditions
apply to all code found in this distribution.

Copyright remains Eric Young's, and as such any Copyright notices in
the code are not to be removed.
If this package is used in a product, Eric Young should be given attribution
as the author of that the SSL library. This can be in the form of a textual
message at program startup or in documentation (online or textual) provided
with the package.

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 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. All advertising materials mentioning features or use of this software
  must display the following acknowledgement:
  This product includes software developed by Eric Young (eay@cryptsoft.com)

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``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.

The license and distribution terms for any publically available version or
derivative of this code cannot be changed. i.e. this code cannot simply be
copied and put under another distrubution license
[including the GNU Public License.]

The reason behind this being stated in this direct manner is past
experience in code simply being copied and the attribution removed
from it and then being distributed as part of other packages. This
implementation was a non-trivial and unpaid effort.
Copyright (C) 1995-1997 Eric Young (eay@cryptsoft.com)
All rights reserved.

This package is an Blowfish implementation written
by Eric Young (eay@cryptsoft.com).

This library is free for commercial and non-commercial use as long as
the following conditions are aheared to. The following conditions
apply to all code found in this distribution.

Copyright remains Eric Young's, and as such any Copyright notices in
the code are not to be removed.

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 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. All advertising materials mentioning features or use of this software
  must display the following acknowledgement:
  This product includes software developed by Eric Young (eay@cryptsoft.com)

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE

LICENSE ISSUES
==============

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of
the OpenSSL License and the original SSLeay license apply to the toolkit.
See below for the actual license texts. Actually both licenses are BSD-style
Open Source licenses. In case of any license issues related to OpenSSL
please contact openssl-core@openssl.org.

OpenSSL License
----

*   software must display the following acknowledgment:
*   "This product includes software developed by the OpenSSL Project
*   for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
*
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
*   endorse or promote products derived from this software without
*   prior written permission. For written permission, please contact
*   openssl-core@openssl.org.
*
* 5. Products derived from this software may not be called "OpenSSL"
*   nor may "OpenSSL" appear in their names without prior written
*   permission of the OpenSSL Project.
*
* 6. Redistributions of any form whatsoever must retain the following
*   acknowledgment:
*   "This product includes software developed by the OpenSSL Project
*   for use in the OpenSSL Toolkit (http://www.openssl.org/)"
*
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
* EXPRESSED 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 OpenSSL PROJECT OR
* ITS 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 product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com). This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*/

Original SSLeay License

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
* All rights reserved.
*
* This package is an SSL implementation written
* by Eric Young (eay@cryptsoft.com).
* The implementation was written so as to conform with Netscapes SSL.
*

*/

1.10 musl 1.1.14 :r11

1.10.1 Available under license :

musl as a whole is licensed under the following standard MIT license:

Authors/contributors include:

Alex Dowad
Alexander Monakov
Anthony G. Basile
Arvid Picciani
Bobby Bingham
Boris Brezillon
Brent Cook
Chris Spiegel
Clment Vasseur
Daniel Micay
Denys Vlasenko
Emil Renner Berthing
Felix Fietkau
Felix Janda
Gianluca Anzolin

Hauke Mehrtens
Hiltjo Posthuma
Isaac Dunham
Jens Gustedt
Jeremy Huntwork
Jo-Philipp Wich
Joakim Sindholt
John Spencer
Josiah Worcester
Justin Cormack
Khem Raj
Kylie McClain
Luca Barbato
Luka Perkov
M Farkas-Dyck (Strake)
Michael Forney
Natanael Copa
Nicholas J. Kain
orc
Pascal Cuoq
Petr Hosek
Pierre Carrier
Rich Felker
Richard Pennington
Shiz
sin
Solar Designer
Stefan Kristiansson
Szabolcs Nagy
Timo Ters
Trutz Behn
Valentin Ochs
William Haddon

Portions of this software are derived from third-party works licensed under terms compatible with the above MIT license:

The TRE regular expression implementation (src/regex/reg* and src/regex/tre*) is Copyright 2001-2008 Ville Laurikari and licensed under a 2-clause BSD license (license text in the source files). The included version has been heavily modified by Rich Felker in 2012, in the interests of size, simplicity, and namespace cleanliness.

Much of the math library code (src/math/* and src/complex/*) is
Copyright 1993,2004 Sun Microsystems or
Copyright 2003-2011 David Schultz or
Copyright 2003-2009 Steven G. Kargl or
Copyright 2003-2009 Bruce D. Evans or

Copyright 2008 Stephen L. Moshier
and labelled as such in comments in the individual source files. All
have been licensed under extremely permissive terms.

The ARM memcpy code (src/string/armel/memcpy.s) is Copyright 2008
The Android Open Source Project and is licensed under a two-clause BSD
license. It was taken from Bionic libc, used on Android.

The implementation of DES for crypt (src/misc/crypt_des.c) is
Copyright 1994 David Burren. It is licensed under a BSD license.

The implementation of blowfish crypt (src/misc/crypt_blowfish.c) was
originally written by Solar Designer and placed into the public
domain. The code also comes with a fallback permissive license for use
in jurisdictions that may not recognize the public domain.

The smoothsort implementation (src/stdlib/qsort.c) is Copyright 2011
Valentin Ochs and is licensed under an MIT-style license.

The BSD PRNG implementation (src/prng/random.c) and XSI search API
(src/search/*.c) functions are Copyright 2011 Szabolcs Nagy and
licensed under following terms: "Permission to use, copy, modify,
and/or distribute this code for any purpose with or without fee is
hereby granted. There is no warranty."

The x86_64 port was written by Nicholas J. Kain. Several files (crt)
were released into the public domain; others are licensed under the
standard MIT license terms at the top of this file. See individual
files for their copyright status.

The mips and microblaze ports were originally written by Richard
Pennington for use in the ellcc project. The original code was adapted
by Rich Felker for build system and code conventions during upstream
integration. It is licensed under the standard MIT terms.

The powerpc port was also originally written by Richard Pennington,
and later supplemented and integrated by John Spencer. It is licensed
under the standard MIT terms.

All other files which have no copyright comments are original works
produced specifically for use as part of this library, written either
by Rich Felker, the main author of the library, or by one or more
contibutors listed above. Details on authorship of individual files
can be found in the git version control history of the project. The
omission of copyright and license comments in each file is in the
interest of source tree size.

All public header files (include/* and arch/*/bits/*) should be

treated as Public Domain as they intentionally contain no content
which can be covered by copyright. Some source modules may fall in
this category as well. If you believe that a file is so trivial that
it should be in the Public Domain, please contact the authors and
request an explicit statement releasing it from copyright.

The following files are trivial, believed not to be copyrightable in
the first place, and hereby explicitly released to the Public Domain:

All public headers: include/*, arch/*/bits/*
Startup files: crt/*

1.11 musl-utils 1.1.14 :r14

1.11.1 Available under license :

musl as a whole is licensed under the following standard MIT license:

Copyright 2005-2014 Rich Felker, et al.

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.

Authors/contributors include:

Alex Dowad
Alexander Monakov
Anthony G. Basile
Arvid Picciani
Bobby Bingham

Boris Brezillon
Brent Cook
Chris Spiegel
Clment Vasseur
Daniel Micay
Denys Vlasenko
Emil Renner Berthing
Felix Fietkau
Felix Janda
Gianluca Anzolin
Hauke Mehrtens
Hiltjo Posthuma
Isaac Dunham
Jens Gustedt
Jeremy Huntwork
Jo-Philipp Wich
Joakim Sindholt
John Spencer
Josiah Worcester
Justin Cormack
Khem Raj
Kylie McClain
Luca Barbato
Luka Perkov
M Farkas-Dyck (Strake)
Michael Forney
Natanael Copa
Nicholas J. Kain
orc
Pascal Cuoq
Petr Hosek
Pierre Carrier
Rich Felker
Richard Pennington
Shiz
sin
Solar Designer
Stefan Kristiansson
Szabolcs Nagy
Timo Ters
Trutz Behn
Valentin Ochs
William Haddon

Portions of this software are derived from third-party works licensed under terms compatible with the above MIT license:

The TRE regular expression implementation (src/regex/reg* and

src/regex/tre*) is Copyright 2001-2008 Ville Laurikari and licensed
under a 2-clause BSD license (license text in the source files). The
included version has been heavily modified by Rich Felker in 2012, in
the interests of size, simplicity, and namespace cleanliness.

Much of the math library code (src/math/* and src/complex/*) is
Copyright 1993,2004 Sun Microsystems or
Copyright 2003-2011 David Schultz or
Copyright 2003-2009 Steven G. Kargl or
Copyright 2003-2009 Bruce D. Evans or
Copyright 2008 Stephen L. Moshier
and labelled as such in comments in the individual source files. All
have been licensed under extremely permissive terms.

The ARM memcpy code (src/string/armel/memcpy.s) is Copyright 2008
The Android Open Source Project and is licensed under a two-clause BSD
license. It was taken from Bionic libc, used on Android.

The implementation of DES for crypt (src/misc/crypt_des.c) is
Copyright 1994 David Burren. It is licensed under a BSD license.

The implementation of blowfish crypt (src/misc/crypt_blowfish.c) was
originally written by Solar Designer and placed into the public
domain. The code also comes with a fallback permissive license for use
in jurisdictions that may not recognize the public domain.

The smoothsort implementation (src/stdlib/qsort.c) is Copyright 2011
Valentin Ochs and is licensed under an MIT-style license.

The BSD PRNG implementation (src/prng/random.c) and XSI search API
(src/search/*.c) functions are Copyright 2011 Szabolcs Nagy and
licensed under following terms: "Permission to use, copy, modify,
and/or distribute this code for any purpose with or without fee is
hereby granted. There is no warranty."

The x86_64 port was written by Nicholas J. Kain. Several files (crt)
were released into the public domain; others are licensed under the
standard MIT license terms at the top of this file. See individual
files for their copyright status.

The mips and microblaze ports were originally written by Richard
Pennington for use in the ellcc project. The original code was adapted
by Rich Felker for build system and code conventions during upstream
integration. It is licensed under the standard MIT terms.

The powerpc port was also originally written by Richard Pennington,
and later supplemented and integrated by John Spencer. It is licensed
under the standard MIT terms.

All other files which have no copyright comments are original works
produced specifically for use as part of this library, written either
by Rich Felker, the main author of the library, or by one or more
contibutors listed above. Details on authorship of individual files
can be found in the git version control history of the project. The
omission of copyright and license comments in each file is in the
interest of source tree size.

All public header files (include/* and arch/*/bits/*) should be
treated as Public Domain as they intentionally contain no content
which can be covered by copyright. Some source modules may fall in
this category as well. If you believe that a file is so trivial that
it should be in the Public Domain, please contact the authors and
request an explicit statement releasing it from copyright.

The following files are trivial, believed not to be copyrightable in
the first place, and hereby explicitly released to the Public Domain:

All public headers: include/*, arch/*/bits/*
Startup files: crt/*

1.12 openrc 0.21 :r2

1.12.1 Available under license :

Bud1 % @ @ @ @ E% DSDB ` @ @ @
Copyright (c) 2007-2008, Roy Marples <roy@marples.name>
Copyright (c) 2007-2015, the OpenRC authors
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.

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 OWNER 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.
0

1.13 scanelf 1.1.6 :r0

1.13.1 Available under license :

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.

1.14 zlib 1.2.8 :(28 Apr 2013)

1.14.1 Available under license :

/* zlib.h -- interface of the 'zlib' general purpose compression library
version 1.2.8, April 28th, 2013

Copyright (C) 1995-2013 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of 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 following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly   Mark Adler
jloup@gzip.org   madler@alumni.caltech.edu

The data format used by the zlib library is described by RFCs (Request for Comments) 1950 to 1952 in the files http://tools.ietf.org/html/rfc1950 (zlib format), rfc1951 (deflate format) and rfc1952 (gzip format).
*/

```
----------------------------------------------------------------------------
--                    ZLib for Ada thick binding.                          --
--                                                                         --
--              Copyright (C) 2002-2004 Dmitriy Anisimkov                  --
--                                                                         --
--  This library 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 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      --
--  General Public License for more details.                               --
--                                                                         --
--  You should have received a copy of the GNU 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.         --
--                                                                         --
--  As a special exception, if other files instantiate generics from this  --
--  unit, or you link this unit with other files to produce an executable, --
--  this  unit  does not  by itself cause  the resulting executable to be  --
--  covered by the GNU General Public License. This exception does not     --
--  however invalidate any other reasons why the executable file  might be --
--  covered by the  GNU Public License.                                    --
----------------------------------------------------------------------------
```