```
 1: /*    Copyright (c) 1990, 1991, 1992, 1993 UNIX System Laboratories, Inc.    */
 2: /*    Copyright (c) 1988, 1990 AT&T    */
 3: /*      All Rights Reserved      */
 4:
 5: /*    THIS IS UNPUBLISHED PROPRIETARY SOURCE CODE OF         */
 6: /*    UNIX System Laboratories, Inc.                         */
 7: /*    The copyright notice above does not evidence any       */
 8: /*    actual or intended publication of such source code.    */
 9:
10: #ifndef _LIBELF_H
11: #define _LIBELF_H
12:
13: #ident    "@(#)sgs-inc:common/libelf.h    1.8.3.1"
14:
15: #include <sys/types.h>
16: #include "sys/elf.h"
17:
18:
19: #undef _
20: #ifdef __STDC__
21:     typedef void        Elf_Void;
22: #    define _(a)        a
23: #else
24:     typedef char        Elf_Void;
25: #    define _(a)        ()
26: #    ifndef _SIZE_T
27: #        define _SIZE_T
28: #        ifndef size_t
29: #            define size_t    unsigned int
30: #        endif
31: #    endif
32: #    undef const
33: #    define const
34: #endif
35:
36:
37: /*    commands
38:  */
39:
40: typedef enum {
41:     ELF_C_NULL = 0,    /* must be first, 0 */
42:     ELF_C_READ,
43:     ELF_C_WRITE,
44:     ELF_C_IMPURE_WRITE,
45:     ELF_C_CLR,
46:     ELF_C_SET,
47:     ELF_C_FDDONE,
48:     ELF_C_FDREAD,
49:     ELF_C_RDWR,
50:     ELF_C_NUM    /* must be last */
51: } Elf_Cmd;
52:
53:
54: /*    flags
55:  */
56:
57: #define ELF_F_DIRTY    0x1
58: #define ELF_F_LAYOUT    0x4
59:
60:
61: /*    file types
62:  */
63:
64: typedef enum {
65:     ELF_K_NONE = 0,    /* must be first, 0 */
66:     ELF_K_AR,
67:     ELF_K_COFF,
68:     ELF_K_ELF,
69:     ELF_K_NUM    /* must be last */
70: } Elf_Kind;
71:
72:
73: /*    translation types
74:  */
75:
76: typedef enum {
77:     ELF_T_BYTE = 0,    /* must be first, 0 */
78:     ELF_T_ADDR,
```



```
 1: /*
 2: @configure_input@
 3: libelf.h - public header file for libelf.
 4: Copyright (C) 1995 Michael Riepe <riepe@ifwsn4.ifw.uni-hannover.de>
 5:
 6: This library is free software; you can redistribute it and/or
 7: modify it under the terms of the GNU Library General Public
 8: License as published by the Free Software Foundation; either
 9: version 2 of the License, or (at your option) any later version.
10:
11: This library is distributed in the hope that it will be useful,
12: but WITHOUT ANY WARRANTY; without even the implied warranty of
13: MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
14: Library General Public License for more details.
15:
16: You should have received a copy of the GNU Library General Public
17: License along with this library; if not, write to the Free Software
18: Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
19: */
20:
21: #ifndef _LIBELF_H
22: #define _LIBELF_H
23:
24: #include <sys/types.h>
25: #include <@elf_h@>
26:
27: #ifdef __cplusplus
28: extern "C" {
29: #endif
30:
31: #ifndef __P
32: # if __STDC__ || defined(__cplusplus)
33: #  define __P(args) args
34: # else
35: #  define __P(args) ()
36: # endif
37: #endif
38:
39: /*
40:  * Commands
41:  */
42: typedef enum {
43:     ELF_C_NULL = 0,    /* must be first, 0 */
44:     ELF_C_READ,
45:     ELF_C_WRITE,
46:     ELF_C_CLR,
47:     ELF_C_SET,
48:     ELF_C_FDDONE,
49:     ELF_C_FDREAD,
50:     ELF_C_RDWR,
51:     ELF_C_NUM        /* must be last */
52: } Elf_Cmd;
53:
54:
55: /*
56:  * Flags
57:  */
58: #define    ELF_F_DIRTY    0x1
59: #define    ELF_F_LAYOUT    0x4
60:
61: /*
62:  * File types
63:  */
64: typedef enum {
65:     ELF_K_NONE = 0,    /* must be first, 0 */
66:     ELF_K_AR,
67:     ELF_K_COFF,
68:     ELF_K_ELF,
69:     ELF_K_NUM        /* must be last */
70: } Elf_Kind;
71:
72: /*
73:  * Data types
74:  */
75: typedef enum {
76:     ELF_T_BYTE = 0,    /* must be first, 0 */
77:     ELF_T_ADDR,
78:     ELF_T_DYN,
```


(For readability, not all matches are indicated.)

```
 79:     ELF_T_DYN,
 80:     ELF_T_EHDR,
 81:     ELF_T_HALF,
 82:     ELF_T_OFF,
 83:     ELF_T_PHDR,
 84:     ELF_T_RELA,
 85:     ELF_T_REL,
 86:     ELF_T_SHDR,
 87:     ELF_T_SWORD,
 88:     ELF_T_SYM,
 89:     ELF_T_WORD,
 90:     ELF_T_NUM    /* must be last */
 91: } Elf_Type;
 92:
 93:
 94: typedef struct Elf    Elf;
 95: typedef struct Elf_Scn    Elf_Scn;
 96:
 97:
 98: /*    archive member header
 99:  */
100:
101: typedef struct {
102:     char        *ar_name;
103:     time_t        ar_date;
104:     long         ar_uid;
105:     long         ar_gid;
106:     unsigned long    ar_mode;
107:     off_t        ar_size;
108:     char         *ar_rawname;
109: } Elf_Arhdr;
110:
111:
112: /*    archive symbol table
113:  */
114:
115: typedef struct {
116:     char        *as_name;
117:     size_t        as_off;
118:     unsigned long    as_hash;
119: } Elf_Arsym;
120:
121:
122: /*    data descriptor
123:  */
124:
125: typedef struct {
126:     Elf_Void    *d_buf;
127:     Elf_Type    d_type;
128:     size_t        d_size;
129:     off_t        d_off;        /* offset into section */
130:     size_t        d_align;    /* alignment in section */
131:     unsigned    d_version;    /* elf version */
132: } Elf_Data;
133:
134:
135: /*    function declarations
136:  */
137:
138: Elf        *elf_begin    _((int, Elf_Cmd, Elf *));
139: int        elf_cntl    _((Elf *, Elf_Cmd));
140: int        elf_end        _((Elf *));
141: const char    *elf_errmsg    _((int));
142: int        elf_errno    _((void));
143: void        elf_fill    _((int));
144: unsigned    elf_flagdata    _((Elf_Data *, Elf_Cmd, unsigned));
145: unsigned    elf_flagehdr    _((Elf *, Elf_Cmd,  unsigned));
146: unsigned    elf_flagelf    _((Elf *, Elf_Cmd, unsigned));
147: unsigned    elf_flagphdr    _((Elf *, Elf_Cmd, unsigned));
148: unsigned    elf_flagscn    _((Elf_Scn *, Elf_Cmd, unsigned));
149: unsigned    elf_flagshdr    _((Elf_Scn *, Elf_Cmd, unsigned));
150: size_t        elf32_fsize    _((Elf_Type, size_t, unsigned));
151: Elf_Arhdr    *elf_getarhdr    _((Elf *));
152: Elf_Arsym    *elf_getarsym    _((Elf *, size_t *));
153: off_t        elf_getbase    _((Elf *));
154: Elf_Data    *elf_getdata    _((Elf_Scn *, Elf_Data *));
155: Elf32_Ehdr    *elf32_getehdr    _((Elf *));
156: char        *elf_getident    _((Elf *, size_t *));
```



```
 79:     ELF_T_EHDR,
 80:     ELF_T_HALF,
 81:     ELF_T_OFF,
 82:     ELF_T_PHDR,
 83:     ELF_T_RELA,
 84:     ELF_T_REL,
 85:     ELF_T_SHDR,
 86:     ELF_T_SWORD,
 87:     ELF_T_SYM,
 88:     ELF_T_WORD,
 89:     ELF_T_NUM        /* must be last */
 90: } Elf_Type;
 91:
 92: /*
 93:  * Elf descriptor
 94:  */
 95: typedef struct Elf    Elf;
 96:
 97: /*
 98:  * Section descriptor
 99:  */
100: typedef struct Elf_Scn    Elf_Scn;
101:
102: /*
103:  * Archive member header
104:  */
105: typedef struct {
106:     char*        ar_name;
107:     time_t        ar_date;
108:     long        ar_uid;
109:     long         ar_gid;
110:     unsigned long    ar_mode;
111:     off_t        ar_size;
112:     char*        ar_rawname;
113: } Elf_Arhdr;
114:
115: /*
116:  * Archive symbol table
117:  */
118: typedef struct {
119:     char*        as_name;
120:     size_t        as_off;
121:     unsigned long    as_hash;
122: } Elf_Arsym;
123:
124: /*
125:  * Data descriptor
126:  */
127: typedef struct {
128:     void*        d_buf;
129:     Elf_Type        d_type;
130:     size_t        d_size;
131:     off_t        d_off;
132:     size_t        d_align;
133:     unsigned        d_version;
134: } Elf_Data;
135:
136: /*
137:  * Function declarations
138:  */
139: extern Elf *elf_begin __P((int __fd, Elf_Cmd __cmd, Elf *__ref));
140: extern int elf_cntl __P((Elf *__elf, Elf_Cmd __cmd));
141: extern int elf_end __P((Elf *__elf));
142: extern const char *elf_errmsg __P((int __err));
143: extern int elf_errno __P((void));
144: extern void elf_fill __P((int __fill))
145: extern unsigned elf_flagdata __P((Elf_Data *__data, Elf_Cmd __cmd,
146:     unsigned __flags));
147: extern unsigned elf_flagehdr __P((Elf *__elf, Elf_Cmd __cmd,
148:     unsigned __flags));
149: extern unsigned elf_flagelf __P((Elf *__elf, Elf_Cmd __cmd,
150:     unsigned __flags));
151: extern unsigned elf_flagphdr __P((Elf *__elf, Elf_Cmd __cmd,
152:     unsigned __flags));
153: extern unsigned elf_flagscn __P((Elf_Scn *__scn, Elf_Cmd __cmd,
154:     unsigned __flags));
155: extern unsigned elf_flagshdr __P((Elf_Scn *__scn, Elf_Cmd __cmd,
156:     unsigned __flags));
```


(For readability, not all matches are indicated.)

```
157: Elf32_Phdr    *elf32_getphdr    _((Elf *));
158: Elf_Scn        *elf_getscn    _((Elf *elf, size_t));
159: Elf32_Shdr    *elf32_getshdr    _((Elf_Scn *));
160: unsigned long    elf_hash    _((const char *));
161: Elf_Kind    elf_kind    _((Elf *));
162: size_t        elf_ndxscn    _((Elf_Scn *));
163: Elf_Data    *elf_newdata    _((Elf_Scn *));
164: Elf32_Ehdr    *elf32_newehdr    _((Elf *));
165: Elf32_Phdr    *elf32_newphdr    _((Elf *, size_t));
166: Elf_Scn        *elf_newscn    _((Elf *));
167: Elf_Scn        *elf_nextscn    _((Elf *, Elf_Scn *));
168: Elf_Cmd        elf_next    _((Elf *));
169: size_t        elf_rand    _((Elf *, size_t));
170: Elf_Data    *elf_rawdata    _((Elf_Scn *, Elf_Data *));
171: char        *elf_rawfile    _((Elf *, size_t *));
172: char        *elf_strptr    _((Elf *, size_t, size_t));
173: off_t        elf_update    _((Elf *, Elf_Cmd));
174: unsigned    elf_version    _((unsigned));
175: Elf_Data    *elf32_xlatetof    _((Elf_Data *, const Elf_Data *, unsigned));
176: Elf_Data    *elf32_xlatetom    _((Elf_Data *, const Elf_Data *, unsigned));
177:
178: #undef    _
179:
180: #endif
```

```
157: extern size_t elf32_fsize __P((Elf_Type __type, size_t __count,
158:     unsigned __ver));
159: extern Elf_Arhdr *elf_getarhdr __P((Elf *__elf));
160: extern Elf_Arsym *elf_getarsym __P((Elf *__elf, size_t *__ptr));
161: extern off_t elf_getbase __P((Elf *__elf));
162: extern Elf_Data *elf_getdata __P((Elf_Scn *__scn, Elf_Data *__data));
163: extern Elf32_Ehdr *elf32_getehdr __P((Elf *__elf));
164: extern char *elf_getident __P((Elf *__elf, size_t *__ptr));
165: extern Elf32_Phdr *elf32_getphdr __P((Elf *__elf));
166: extern Elf_Scn *elf_getscn __P((Elf *__elf, size_t __index));
167: extern Elf32_Shdr *elf32_getshdr __P((Elf_Scn *__scn));
168: extern unsigned long elf_hash __P((const char *__name));
169: extern Elf_Kind elf_kind __P((Elf *__elf));
170: extern size_t elf_ndxscn __P((Elf_Scn *__scn));
171: extern Elf_Data *elf_newdata __P((Elf_Scn *__scn));
172: extern Elf32_Ehdr *elf32_newehdr __P((Elf *__elf));
173: extern Elf32_Phdr *elf32_newphdr __P((Elf *__elf, size_t __count));
174: extern Elf_Scn *elf_newscn __P((Elf *__elf));
175: extern Elf_Cmd elf_next __P((Elf *__elf));
176: extern Elf_Scn *elf_nextscn __P((Elf *__elf, Elf_Scn *__scn));
177: extern size_t elf_rand __P((Elf *__elf, size_t __offset));
178: extern Elf_Data *elf_rawdata __P((Elf_Scn *__scn, Elf_Data *__data));
179: extern char *elf_rawfile __P((Elf *__elf, size_t *__ptr));
180: extern char *elf_strptr __P((Elf *__elf, size_t __section, size_t __offset));
181: extern off_t elf_update __P((Elf *__elf, Elf_Cmd __cmd));
182: extern unsigned elf_version __P((unsigned __ver));
183: extern Elf_Data *elf32_xlatetof __P((Elf_Data *__dst, const Elf_Data *__src,
184:     unsigned __encode));
185: extern Elf_Data *elf32_xlatetom __P((Elf_Data *__dst, const Elf_Data *__src,
186:     unsigned __encode));
187:
188: #ifdef __cplusplus
189: }
190: #endif
191:
192: #endif /* _LIBELF_H */
```


(For readability, not all matches are indicated.)