```
/*
  libxbee - a C library to aid the use of Digi's Series 1 XBee modules
            running in API mode (AP=2).

  Copyright (C) 2009  Attie Grande (attie@attie.co.uk)

  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 3 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, see <http://www.gnu.org/licenses/>.
*/

#include <stdio.h>
#include <stdlib.h>

#include <stdarg.h>

#include <string.h>
#include <fcntl.h>
#include <errno.h>
#include <signal.h>

#ifdef __GNUC__ /* ---- */
#include <unistd.h>
#include <termios.h>
#define __USE_GNU
#include <pthread.h>
#undef __USE_GNU
#include <sys/time.h>
#else /* -------------- */
#include <Windows.h>
#include <io.h>
#include <time.h>
#include <sys/timeb.h>
#endif /* ------------- */

#ifdef __UMAKEFILE
  #define HOST_OS "Embedded"
#elif defined(__GNUC__)
  #define HOST_OS "Linux"
#elif defined(_WIN32)
  #define HOST_OS "Win32"
#else
  #define HOST_OS "UNKNOWN"
#endif

#define TRUE 1
#define FALSE 0

#define M8(x) (x & 0xFF)

/* various connection types */
#define XBEE_LOCAL_AT     0x88
#define XBEE_LOCAL_ATREQ  0x08
#define XBEE_LOCAL_ATQUE  0x09

#define XBEE_REMOTE_AT    0x97
#define XBEE_REMOTE_ATREQ 0x17

#define XBEE_MODEM_STATUS 0x8A

/* XBee Series 1 stuff */
#define XBEE_TX_STATUS    0x89
#define XBEE_64BIT_DATATX 0x00
#define XBEE_64BIT_DATARX 0x80
#define XBEE_16BIT_DATATX 0x01
#define XBEE_16BIT_DATARX 0x81
#define XBEE_64BIT_IO     0x82
#define XBEE_16BIT_IO     0x83

/* XBee Series 2 stuff */
#define XBEE2_DATATX      0x10
#define XBEE2_DATARX      0x90
#define XBEE2_TX_STATUS   0x8B

typedef struct xbee_hnd* xbee_hnd;

```

```
  86: #define __LIBXBEE_API_H
  87: #include "xbee.h"
  88:
  89: typedef struct t_threadList t_threadList;
  90: struct t_threadList {
  91:   xbee_thread_t thread;
  92:   t_threadList *next;
  93: };
  94:
  95: struct xbee_hnd {
  96:   xbee_file_t tty;
  97: #ifdef __GNUC__ /* ---- */
  98:   int ttyfd;
  99: #else /* -------------- */
 100:   int ttyr;
 101:   int ttyw;
 102:   int ttyeof;
 103:
 104:   OVERLAPPED ttyovrw;
 105:   OVERLAPPED ttyovrr;
 106:   OVERLAPPED ttyovrs;
 107: #endif /* ------------- */
 108:
 109:   char *path; /* serial port path */
 110:
 111:   xbee_mutex_t logmutex;
 112:   FILE *log;
 113:   int logfd;
 114:
 115:   xbee_mutex_t conmutex;
 116:   xbee_con *conlist;
 117:
 118:   xbee_mutex_t pktmutex;
 119:   xbee_pkt *pktlist;
 120:   xbee_pkt *pktlast;
 121:   int pktcount;
 122:
 123:   xbee_mutex_t sendmutex;
 124:
 125:   xbee_thread_t listent;
 126:
 127:   xbee_thread_t threadt;
 128:   xbee_mutex_t  threadmutex;
 129:   xbee_sem_t    threadsem;
 130:   t_threadList *threadList;
 131:
 132:   int run;
 133:
 134:   int oldAPI;
 135:   char cmdSeq;
 136:   int cmdTime;
 137:
 138:   /* ready flag.
 139:      needs to be set to -1 so that the listen thread can begin. */
 140:   volatile int xbee_ready;
 141:
 142:   xbee_hnd next;
 143: };
 144: xbee_hnd default_xbee = NULL;
 145: xbee_mutex_t xbee_hnd_mutex;
 146:
 147: typedef struct t_data t_data;
 148: struct t_data {
 149:   unsigned char data[128];
 150:   unsigned int length;
 151: };
 152:
 153: typedef struct t_LTinfo t_LTinfo;
 154: struct t_LTinfo {
 155:   int i;
 156:   xbee_hnd xbee;
 157: };
 158:
 159: typedef struct t_CBinfo t_CBinfo;
 160: struct t_CBinfo {
 161:   xbee_hnd xbee;
 162:   xbee_con *con;
 163: };
 164:
 165: typedef struct t_callback_list t_callback_list;
 166: struct t_callback_list {
 167:   xbee_pkt *pkt;
 168:   t_callback_list *next;
 169: };
 170:
```

```
  171: static void *Xmalloc2(xbee_hnd xbee, size_t size);
  172: static void *Xcalloc2(xbee_hnd xbee, size_t size);
  173: static void *Xrealloc2(xbee_hnd xbee, void *ptr, size_t size);
  174: static void Xfree2(void **ptr);
  175: #define Xmalloc(x)     Xmalloc2(xbee,(x))
  176: #define Xcalloc(x)     Xcalloc2(xbee,(x))
  177: #define Xrealloc(x,y)  Xrealloc2(xbee,(x),(y))
  178: #define Xfree(x)       Xfree2((void **)&x)
  179:
  180: /* usage:
  181:     xbee_logSf()   lock the log
  182:     xbee_logEf()   unlock the log
  183:
  184:     xbee_log()     lock   print with \n      unlock          # to print a single line
  185:     xbee_logc()    lock   print with no \n                   # to print a single line with a custom ending
  186:     xbee_logcf()          print \n           unlock          # to end a custom-ended single line
  187:
  188:     xbee_logS()    lock   print with \n                      # to start a continuous block
  189:     xbee_logI()           print with \n                      # to continue a continuous block
  190:     xbee_logIc()          print with no \n                  # to continue a continuous block with a custom
ending
  191:     xbee_logIcf()         print \n                          # to continue a continuous block with ended cus
tom-ended line
  192:     xbee_logE()           print with \n      unlock          # to end a continuous block
  193: */
  194: static void xbee_logf(xbee_hnd xbee, const char *logformat, const char *file,
  195:                       const int line, const char *function, char *format, ...);
  196: #define LOG_FORMAT "[%s:%d] %s(): %s"
  197:
  198: #define xbee_logSf()      if (xbee->log) { xbee_mutex_lock(xbee->logmutex);   }
  199: #define xbee_logEf()      if (xbee->log) { xbee_mutex_unlock(xbee->logmutex); }
  200:
  201: #define xbee_log(...)     if (xbee->log) { xbee_logSf(); xbee_logf(xbee,LOG_FORMAT"\n",__FILE__,__LINE__,__F
UNCTION__,__VA_ARGS__); xbee_logEf(); }
  202: #define xbee_logc(...)    if (xbee->log) { xbee_logSf(); xbee_logf(xbee,LOG_FORMAT    ,__FILE__,__LINE__,__F
UNCTION__,__VA_ARGS__);               }
  203: #define xbee_logcf()      if (xbee->log) {               fprintf(xbee->log, "\n");
                    xbee_logEf(); }
  204:
  205: #define xbee_logS(...)    if (xbee->log) { xbee_logSf(); xbee_logf(xbee,LOG_FORMAT"\n",__FILE__,__LINE__,__F
UNCTION__,__VA_ARGS__);               }
  206: #define xbee_logI(...)    if (xbee->log) {               xbee_logf(xbee,LOG_FORMAT"\n",__FILE__,__LINE__,__F
UNCTION__,__VA_ARGS__);               }
  207: #define xbee_logIc(...)   if (xbee->log) {               xbee_logf(xbee,LOG_FORMAT    ,__FILE__,__LINE__,__F
UNCTION__,__VA_ARGS__);               }
  208: #define xbee_logIcf()     if (xbee->log) {               fprintf(xbee->log, "\n");
                                      }
  209: #define xbee_logE(...)    if (xbee->log) {               xbee_logf(xbee,LOG_FORMAT"\n",__FILE__,__LINE__,__F
UNCTION__,__VA_ARGS__); xbee_logEf(); }
  210:
  211: #define xbee_perror(str)                                 \
  212:   if (xbee->log) xbee_logI("%s:%s",str,strerror(errno)); \
  213:   perror(str);
  214:
  215: static int xbee_startAPI(xbee_hnd xbee);
  216:
  217: static int xbee_sendAT(xbee_hnd xbee, char *command, char *retBuf, int retBuflen);
  218: static int xbee_sendATdelay(xbee_hnd xbee, int guardTime, char *command, char *retBuf, int retBuflen);
  219:
  220: static int xbee_parse_io(xbee_hnd xbee, xbee_pkt *p, unsigned char *d,
  221:                          int maskOffset, int sampleOffset, int sample);
  222:
  223: static void xbee_thread_watch(xbee_hnd xbee);
  224: static void xbee_listen_wrapper(xbee_hnd xbee);
  225: static int xbee_listen(xbee_hnd xbee);
  226: static unsigned char xbee_getbyte(xbee_hnd xbee);
  227: static unsigned char xbee_getrawbyte(xbee_hnd xbee);
  228: static int xbee_matchpktcon(xbee_hnd xbee, xbee_pkt *pkt, xbee_con *con);
  229:
  230: static t_data *xbee_make_pkt(xbee_hnd xbee, unsigned char *data, int len);
  231: static int _xbee_send_pkt(xbee_hnd xbee, t_data *pkt, xbee_con *con);
  232: static void xbee_callbackWrapper(t_CBinfo *info);
  233:
  234: /* these functions can be found in the xsys files */
  235: static int init_serial(xbee_hnd xbee, int baudrate);
  236: static int xbee_select(xbee_hnd xbee, struct timeval *timeout);
  237:
  238: #ifdef __GNUC__ /* ---- */
  239: #include "xsys/linux.c"
  240: #else /* -------------- */
  241: #include "xsys\win32.c"
  242: #endif /* ------------- */
  243:
  244: #ifndef Win32Message
  245: #define Win32Message()
```

```
246: #endif
247:
248: #define ISREADY(a)      if (!xbee || !xbee->xbee_ready) {                                   \
249:                           if (stderr) fprintf(stderr,"libxbee: Run xbee_setup() first!...\n");  \
250:                           Win32Message();                                                   \
251:                           a;                                                                \
252:                         }
253: #define ISREADYP()      ISREADY(return)
254: #define ISREADYR(a)     ISREADY(return a)
```