```
/*
  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>
#include <pthread.h>
#include <sys/time.h>
#else /* -------------- */
#include <Windows.h>
#include <io.h>
#include <time.h>
#include <sys/timeb.h>
#endif /* ------------- */

#include "xbee.h"

#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 ISREADY                                     \
  if (!xbee_ready) {                                \
    if (stderr) fprintf(stderr,"libxbee: Run xbee_setup() first!...\n"); \
    exit(1);                                        \
  }

#define M8(x) (x & 0xFF)
#define FDO(x,y,z)                              \
  if (((x) = fdopen((y),(z))) == NULL) {        \
    perror("fopen()");                          \
    return(-1);                                 \
  }
#define FO(x,y,z)                               \
  if (((x) = open((y),(z))) == -1) {            \
    perror("open()");                           \
    return(-1);                                 \
  }

/* 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

```

```
 86: #define XBEE_TX_STATUS    0x89
 87: #define XBEE_64BIT_DATATX 0x00
 88: #define XBEE_64BIT_DATA   0x80
 89: #define XBEE_16BIT_DATATX 0x01
 90: #define XBEE_16BIT_DATA   0x81
 91:
 92: #define XBEE_64BIT_IO     0x82
 93: #define XBEE_16BIT_IO     0x83
 94:
 95: typedef struct t_data t_data;
 96: struct t_data {
 97:   unsigned char data[128];
 98:   unsigned int length;
 99: };
100:
101: typedef struct t_info t_info;
102: struct t_info {
103:   int i;
104: };
105:
106: typedef struct t_callback_list t_callback_list;
107: struct t_callback_list {
108:   xbee_pkt *pkt;
109:   t_callback_list *next;
110: };
111:
112: struct {
113: #ifdef __GNUC__ /* ---- */
114:   pthread_mutex_t logmutex;
115:   pthread_mutex_t conmutex;
116:   pthread_mutex_t pktmutex;
117:   pthread_mutex_t sendmutex;
118:   pthread_t listent;
119:
120:   FILE *tty;
121:   int ttyfd;
122: #else /* -------------- */
123:   HANDLE logmutex;
124:   HANDLE conmutex;
125:   HANDLE pktmutex;
126:   HANDLE sendmutex;
127:   HANDLE listent;
128:
129:   HANDLE tty;
130:   int ttyr;
131:   int ttyw;
132:
133:   OVERLAPPED ttyovrw;
134:   OVERLAPPED ttyovrr;
135:   OVERLAPPED ttyovrs;
136: #endif /* ------------- */
137:
138:   char *path; /* serial port path */
139:
140:   FILE *log;
141:   int logfd;
142:
143:   xbee_con *conlist;
144:
145:   xbee_pkt *pktlist;
146:   xbee_pkt *pktlast;
147:   int pktcount;
148:
149:   int listenrun;
150:
151:   int oldAPI;
152:   char cmdSeq;
153:   int cmdTime;
154: } xbee;
155:
156: /* ready flag.
157:    needs to be set to -1 so that the listen thread can begin.
158:    then 1 so that functions can be used (after setup of course...) */
159: volatile int xbee_ready = 0;
160:
161: static void *Xmalloc(size_t size);
162: static void *Xcalloc(size_t size);
163: static void *Xrealloc(void *ptr, size_t size);
164: static void Xfree2(void **ptr);
165: #define Xfree(x) Xfree2((void **)&x)
166:
167: static void xbee_logf(const char *logformat, int unlock, const char *file,
168:                       const int line, const char *function, char *format, ...);
169: #define xbee_log(...) xbee_logf("[%s:%d] %s(): %s\n",1,__FILE__,__LINE__,__FUNCTION__,__VA_ARGS__)
170: #define xbee_logc(...) xbee_logf("[%s:%d] %s(): %s",0,__FILE__,__LINE__,__FUNCTION__,__VA_ARGS__)
```

```
171: #define xbee_logcf()                \
172:   fprintf(xbee.log,"\n");           \
173:   xbee_mutex_unlock(xbee.logmutex);  \
174:
175: static int xbee_startAPI(void);
176:
177: static int xbee_sendAT(char *command, char *retBuf, int retBuflen);
178: static int xbee_sendATdelay(int guardTime, char *command, char *retBuf, int retBuflen);
179:
180: static int xbee_parse_io(xbee_pkt *p, unsigned char *d, int maskOffset, int sampleOffset, int sample);
181: static void xbee_listen_wrapper(t_info *info);
182: static int xbee_listen(t_info *info);
183: static unsigned char xbee_getbyte(void);
184: static unsigned char xbee_getrawbyte(void);
185: static int xbee_matchpktcon(xbee_pkt *pkt, xbee_con *con);
186:
187: static t_data *xbee_make_pkt(unsigned char *data, int len);
188: static void xbee_send_pkt(t_data *pkt);
189: static void xbee_callbackWrapper(xbee_con *con);
190:
191: /* these functions can be found in the xsys files */
192: static int init_serial(int baudrate);
193: static int xbee_select(struct timeval *timeout);
```