```
/*
  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 /* ------------- */

#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)
#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

#define XBEE_TX_STATUS    0x89
#define XBEE_64BIT_DATATX 0x00
#define XBEE_64BIT_DATA   0x80
#define XBEE_16BIT_DATATX 0x01
#define XBEE_16BIT_DATA   0x81

#define XBEE_64BIT_IO     0x82
#define XBEE_16BIT_IO     0x83
```

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

```
171: #define xbee_logc(...) xbee_logf(xbee,LOG_FORMAT,0,__FILE__,__LINE__,__FUNCTION__,__VA_ARGS__)
172: #define xbee_logcf(xbee)               \
173:   fprintf((xbee)->log,"\n");           \
174:   xbee_mutex_unlock((xbee)->logmutex); \
175:
176: static int xbee_startAPI(xbee_hnd xbee);
177:
178: static int xbee_sendAT(xbee_hnd xbee, char *command, char *retBuf, int retBuflen);
179: static int xbee_sendATdelay(xbee_hnd xbee, int guardTime, char *command, char *retBuf, int retBuflen);
180:
181: static int xbee_parse_io(xbee_hnd xbee, xbee_pkt *p, unsigned char *d,
182:                          int maskOffset, int sampleOffset, int sample);
183:
184: static void xbee_listen_wrapper(t_LTinfo *info);
185: static int xbee_listen(xbee_hnd xbee, t_LTinfo *info);
186: static unsigned char xbee_getbyte(xbee_hnd xbee);
187: static unsigned char xbee_getrawbyte(xbee_hnd xbee);
188: static int xbee_matchpktcon(xbee_hnd xbee, xbee_pkt *pkt, xbee_con *con);
189:
190: static t_data *xbee_make_pkt(xbee_hnd xbee, unsigned char *data, int len);
191: static int xbee_send_pkt(xbee_hnd xbee, t_data *pkt, xbee_con *con);
192: static void xbee_callbackWrapper(t_CBinfo *info);
193:
194: /* these functions can be found in the xsys files */
195: static int init_serial(xbee_hnd xbee, int baudrate);
196: static int xbee_select(xbee_hnd xbee, struct timeval *timeout);
197:
198: #ifdef __GNUC__ /* ---- */
199: #include "xsys/linux.c"
200: #else /* -------------- */
201: #include "xsys\win32.c"
202: #endif /* ------------- */
```