```
/*
    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/>.
*/

#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);                                     \
  }

struct t_data {
  unsigned char data[128];
  unsigned int length;
};
typedef struct t_data t_data;

struct t_info {
  int i;
};
typedef struct t_info t_info;

struct {
#ifdef __GNUC__ /* ---- */
  pthread_mutex_t conmutex;
  pthread_mutex_t pktmutex;
  pthread_mutex_t sendmutex;
  pthread_t listent;

  FILE *tty;
  int ttyfd;
#else /* -------------- */
  HANDLE conmutex;
  HANDLE pktmutex;
  HANDLE sendmutex;
  HANDLE listent;

  HANDLE tty;
  int ttyr;
  int ttyw;

  OVERLAPPED ttyovrw;
  OVERLAPPED ttyovrr;
  OVERLAPPED ttyovrs;
#endif /* ------------- */

  char *path; /* serial port path */

  FILE *log;
  int logfd;

  xbee_con *conlist;

  xbee_pkt *pktlist;
  xbee_pkt *pktlast;
```

```
  int pktcount;

  int listenrun;

  int oldAPI;
  char cmdSeq;
  int cmdTime;
} xbee;

/* ready flag.
   needs to be set to -1 so that the listen thread can begin.
   then 1 so that functions can be used (after setup of course...) */
volatile int xbee_ready = 0;

static int init_serial(int baudrate);

static void *Xmalloc(size_t size);
static void *Xrealloc(void *ptr, size_t size);
static void Xfree2(void **ptr);
#define Xfree(x) Xfree2((void **)&x)

static void xbee_logf(const char *logformat, const char *function, char *format, ...);
#define xbee_log(...) xbee_logf("%s(): %s\n",__FUNCTION__,__VA_ARGS__)
#define xbee_logc(...) xbee_logf("%s(): %s",__FUNCTION__,__VA_ARGS__)

static int xbee_startAPI(void);

static int xbee_select(struct timeval *timeout);

static int xbee_sendAT(char *command, char *retBuf, int retBuflen);
static int xbee_sendATdelay(int guardTime, char *command, char *retBuf, int retBuflen);

static int xbee_parse_io(xbee_pkt *p, unsigned char *d, int maskOffset, int sampleOffset, int sample);
static void xbee_listen_wrapper(t_info *info);
static int xbee_listen(t_info *info);
static unsigned char xbee_getbyte(void);
static unsigned char xbee_getrawbyte(void);
static int xbee_matchpktcon(xbee_pkt *pkt, xbee_con *con);

static t_data *xbee_make_pkt(unsigned char *data, int len);
static void xbee_send_pkt(t_data *pkt);
```