```
/*
  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/>.
*/

#ifndef XBEE_H
#define XBEE_H

#if !defined(__GNUC__) && !defined(_WIN32)
#error "This library is only currently compatible with Linux and Win32"
#endif

#ifdef __cplusplus
extern "C" {
#endif

#include <stdarg.h>

enum xbee_types {
  xbee_unknown,

  xbee_localAT,       /* frame ID */
  xbee_remoteAT,

  xbee_16bitRemoteAT, /* frame ID */
  xbee_64bitRemoteAT, /* frame ID */

  xbee_16bitData,     /* frame ID for ACKs */
  xbee_64bitData,     /* frame ID for ACKs */

  xbee_16bitIO,
  xbee_64bitIO,

  xbee_txStatus,
  xbee_modemStatus
};
typedef enum xbee_types xbee_types;

typedef struct xbee_sample xbee_sample;
struct xbee_sample {
  /* X  A5 A4 A3 A2 A1 A0 D8    D7 D6 D5 D4 D3 D2 D1 D0  */
  unsigned short IOmask;          /*                  IO */
  /* X  X  X  X  X  X  X  D8    D7 D6 D5 D4 D3 D2 D1 D0  */
  unsigned short IOdigital;       /*                  IO */
  /* X  X  X  X  X  D  D  D     D  D  D  D  D  D  D  D   */
  unsigned short IOanalog[6];     /*                  IO */
};

typedef struct xbee_pkt xbee_pkt;
struct xbee_pkt {
  unsigned int sAddr64        : 1; /* yes / no */
  unsigned int dataPkt        : 1; /* if no - AT packet */
  unsigned int txStatusPkt    : 1;
  unsigned int modemStatusPkt : 1;
  unsigned int remoteATPkt    : 1;
  unsigned int IOPkt          : 1;
  unsigned int __spare__      : 2;

  unsigned char frameID;          /* AT        Status    */
  unsigned char atCmd[2];         /* AT                  */

  unsigned char status;           /* AT  Data  Status    */ /* status / options */
  unsigned char samples;
  unsigned char RSSI;             /*     Data            */

  unsigned char Addr16[2];        /* AT  Data            */

  unsigned char Addr64[8];        /* AT  Data            */

  unsigned char data[128];        /* AT  Data            */
```

```
 86:
 87:   unsigned int datalen;
 88:   xbee_types type;
 89:
 90:   xbee_pkt *next;
 91:
 92:   xbee_sample IOdata[1];  /* this array can be extended by using a this trick:
 93:                              p = calloc(sizeof(xbee_pkt) + (sizeof(xbee_sample) * (samples - 1))) */
 94: };
 95:
 96: typedef struct xbee_con xbee_con;
 97: struct xbee_con {
 98:   unsigned int tAddr64       : 1;
 99:   unsigned int atQueue       : 1; /* queues AT commands until AC is sent */
100:   unsigned int txDisableACK  : 1;
101:   unsigned int txBroadcast   : 1; /* broadcasts to PAN */
102:   unsigned int destroySelf   : 1; /* if set, the callback thread will destroy the connection
103:                                      after all of the packets have been processed */
104:   unsigned int __spare__     : 3;
105:   xbee_types type;
106:   unsigned char frameID;
107:   unsigned char tAddr[8];         /* 64-bit 0-7   16-bit 0-1 */
108:   void (*callback)(xbee_con*,xbee_pkt*); /* call back function */
109:   void *callbackList;
110: #ifdef __GNUC__ /* ---- */
111:   pthread_mutex_t callbackmutex;
112:   pthread_mutex_t callbackListmutex;
113: #else /* -------------- */
114:   HANDLE callbackmutex;
115:   HANDLE callbackListmutex;
116: #endif /* ------------- */
117:   xbee_con *next;
118: };
119:
120: int xbee_setup(char *path, int baudrate);
121: int xbee_setuplog(char *path, int baudrate, int logfd);
122: int xbee_setupAPI(char *path, int baudrate, char cmdSeq, int cmdTime);
123: int xbee_setuplogAPI(char *path, int baudrate, int logfd, char cmdSeq, int cmdTime);
124:
125: int xbee_end(void);
126:
127: xbee_con *xbee_newcon(unsigned char frameID, xbee_types type, ...);
128:
129: void xbee_flushcon(xbee_con *con);
130:
131: void xbee_endcon2(xbee_con **con, int skipUnlink);
132: #define xbee_endcon(x) xbee_endcon2(&(x),0)
133:
134: #ifdef __GNUC__ /* ---- */
135: int xbee_senddata(xbee_con *con, char *format, ...) __attribute__ ((format (printf,2,3)));
136: int xbee_vsenddata(xbee_con *con, char *format, va_list ap) __attribute__ ((format (printf,2,0)));
137: #else /* -------------- */
138: int xbee_senddata(xbee_con *con, char *format, ...);
139: int xbee_vsenddata(xbee_con *con, char *format, va_list ap);
140:
141: /* oh and just 'cos windows has rubbish memory management rules... this too */
142: void xbee_free(void *ptr);
143: #endif /* ------------- */
144:
145: int xbee_nsenddata(xbee_con *con, char *data, int length);
146:
147: xbee_pkt *xbee_getpacketwait(xbee_con *con);
148: xbee_pkt *xbee_getpacket(xbee_con *con);
149:
150: int xbee_hasdigital(xbee_pkt *pkt, int sample, int input);
151: int xbee_getdigital(xbee_pkt *pkt, int sample, int input);
152:
153: int xbee_hasanalog(xbee_pkt *pkt, int sample, int input);
154: double xbee_getanalog(xbee_pkt *pkt, int sample, int input, double Vref);
155:
156: const char *xbee_svn_version(void);
157:
158: void xbee_listen_stop(void);
159:
160: #ifdef __cplusplus
161: }
162: #endif
163:
164: #endif
```