

```

1:  /*
2:  libxbee - a C library to aid the use of Digi's Series 1 XBee modules
3:  running in API mode (AP=2).
4:
5:  Copyright (C) 2009 Attie Grande (attie@attie.co.uk)
6:
7:  This program is free software: you can redistribute it and/or modify
8:  it under the terms of the GNU General Public License as published by
9:  the Free Software Foundation, either version 3 of the License, or
10: (at your option) any later version.
11:
12: This program is distributed in the hope that it will be useful,
13: but WITHOUT ANY WARRANTY; without even the implied warranty of
14: MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
15: GNU General Public License for more details.
16:
17: You should have received a copy of the GNU General Public License
18: along with this program. If not, see <http://www.gnu.org/licenses/>.
19: */
20: #ifndef XBEE_H
21: #define XBEE_H
22:
23: #if !defined(__GNUC__) && !defined(_WIN32)
24: #error "This library is only currently compatible with Linux and Win32"
25: #endif
26:
27: #ifdef __cplusplus
28: extern "C" {
29: #endif
30:
31: #ifndef __LIBXBEE_API_H
32: typedef void* xbee_hnd;
33: #endif
34:
35: #include <stdarg.h>
36:
37: #ifdef __GNUC__ /* ---- */
38: #include <semaphore.h>
39: typedef pthread_mutex_t xbee_mutex_t;
40: typedef pthread_cond_t xbee_cond_t;
41: typedef pthread_t xbee_thread_t;
42: typedef sem_t xbee_sem_t;
43: typedef FILE* xbee_file_t;
44: #else /* ----- */
45: #include <Windows.h>
46: typedef CRITICAL_SECTION xbee_mutex_t;
47: typedef CONDITION_VARIABLE xbee_cond_t;
48: typedef HANDLE xbee_thread_t;
49: typedef HANDLE xbee_sem_t;
50: typedef HANDLE xbee_file_t;
51: #endif /* ----- */
52:
53: enum xbee_types {
54: xbee_unknown,
55:
56: xbee_localAT, /* frame ID */
57: xbee_remoteAT,
58:
59: xbee_16bitRemoteAT, /* frame ID */
60: xbee_64bitRemoteAT, /* frame ID */
61:
62: xbee_16bitData, /* frame ID for ACKs */
63: xbee_64bitData, /* frame ID for ACKs */
64:
65: xbee_16bitIO,
66: xbee_64bitIO,
67:
68: xbee_txStatus,
69: xbee_modemStatus
70: };
71: typedef enum xbee_types xbee_types;
72:
73: typedef struct xbee_sample xbee_sample;
74: struct xbee_sample {
75: /* X A5 A4 A3 A2 A1 A0 D8 D7 D6 D5 D4 D3 D2 D1 D0 */
76: unsigned short IOmask; /* IO */
77: /* X X X X X X X X D8 D7 D6 D5 D4 D3 D2 D1 D0 */
78: unsigned short IOdigital; /* IO */
79: /* X X X X X D D D D D D D D D D */
80: unsigned short IOanalog[6]; /* IO */
81: };
82:
83: typedef struct xbee_pkt xbee_pkt;
84: struct xbee_pkt {
85: unsigned int sAddr64 : 1; /* yes / no */

```

```

86:  unsigned int dataPkt      : 1; /* if no - AT packet */
87:  unsigned int txStatusPkt  : 1;
88:  unsigned int modemStatusPkt : 1;
89:  unsigned int remoteATPkt  : 1;
90:  unsigned int IOPkt        : 1;
91:  unsigned int __spare__    : 2;
92:
93:  unsigned char frameID;      /* AT      Status */
94:  unsigned char atCmd[2];    /* AT      */
95:
96:  unsigned char status;      /* AT Data Status */ /* status / options */
97:  unsigned char samples;
98:  unsigned char RSSI;        /* Data */
99:
100: unsigned char Addr16[2];    /* AT Data */
101:
102: unsigned char Addr64[8];    /* AT Data */
103:
104: unsigned char data[128];    /* AT Data */
105:
106: unsigned int datalen;
107: xbee_types type;
108:
109: xbee_pkt *next;
110:
111: xbee_sample IOdata[1]; /* this array can be extended by using a this trick:
112:                        p = calloc(sizeof(xbee_pkt) + (sizeof(xbee_sample) * (samples - 1))) */
113: };
114:
115: typedef struct xbee_con xbee_con;
116: struct xbee_con {
117:     unsigned int tAddr64      : 1;
118:     unsigned int atQueue     : 1; /* queues AT commands until AC is sent */
119:     unsigned int txDisableACK : 1;
120:     unsigned int txBroadcast  : 1; /* broadcasts to PAN */
121:     unsigned int destroySelf  : 1; /* if set, the callback thread will destroy the connection
122:                                    after all of the packets have been processed */
123:     unsigned int waitForACK   : 1; /* waits for the ACK or NAK after transmission */
124:     unsigned int __spare__    : 2;
125:     xbee_types type;
126:     unsigned char frameID;
127:     unsigned char tAddr[8];    /* 64-bit 0-7 16-bit 0-1 */
128:     void (*callback)(xbee_con*, xbee_pkt*); /* call back function */
129:     void *callbackList;
130:     xbee_mutex_t callbackmutex;
131:     xbee_mutex_t callbackListmutex;
132:     xbee_mutex_t Txmutex;
133:     xbee_sem_t waitForACKsem;
134:     volatile unsigned char ACKstatus; /* 255 = waiting, 0 = success, 1 = no ack, 2 = cca fail, 3 = purged */
135:     xbee_con *next;
136: };
137:
138: int xbee_setup(char *path, int baudrate);
139: int xbee_setuplog(char *path, int baudrate, int logfd);
140: int xbee_setupAPI(char *path, int baudrate, char cmdSeq, int cmdTime);
141: int xbee_setuplogAPI(char *path, int baudrate, int logfd, char cmdSeq, int cmdTime);
142: xbee_hnd _xbee_setup(char *path, int baudrate);
143: xbee_hnd _xbee_setuplog(char *path, int baudrate, int logfd);
144: xbee_hnd _xbee_setupAPI(char *path, int baudrate, char cmdSeq, int cmdTime);
145: xbee_hnd _xbee_setuplogAPI(char *path, int baudrate, int logfd, char cmdSeq, int cmdTime);
146:
147: int xbee_end(void);
148: int _xbee_end(xbee_hnd xbee);
149:
150: void xbee_logit(char *str);
151: void _xbee_logit(xbee_hnd xbee, char *str);
152:
153: xbee_con *xbee_newcon(unsigned char frameID, xbee_types type, ...);
154: xbee_con *_xbee_newcon(xbee_hnd xbee, unsigned char frameID, xbee_types type, ...);
155: xbee_con *_xbee_vnewcon(xbee_hnd xbee, unsigned char frameID, xbee_types type, va_list ap);
156:
157: void xbee_flushcon(xbee_con *con);
158: void _xbee_flushcon(xbee_hnd xbee, xbee_con *con);
159:
160: void xbee_endcon2(xbee_con **con, int alreadyUnlinked);
161: void _xbee_endcon2(xbee_hnd xbee, xbee_con **con, int alreadyUnlinked);
162: #define xbee_endcon(x) xbee_endcon2(&(x), 0)
163: #define _xbee_endcon(xbee, x) _xbee_endcon2((xbee), &(x), 0)
164:
165: int xbee_nsenddata(xbee_con *con, char *data, int length);
166: int _xbee_nsenddata(xbee_hnd xbee, xbee_con *con, char *data, int length);
167: #ifdef __GNUC__ /* ---- */
168: int xbee_senddata(xbee_con *con, char *format, ...)
169:     __attribute__((format(printf, 2, 3)));
170: int _xbee_senddata(xbee_hnd xbee, xbee_con *con, char *format, ...)

```

```
171:         __attribute__ ((format (printf,3,4)));
172: int xbee_vsnddata(xbee_con *con, char *format, va_list ap)
173:         __attribute__ ((format (printf,2,0)));
174: int _xbee_vsnddata(xbee_hnd xbee, xbee_con *con, char *format, va_list ap)
175:         __attribute__ ((format (printf,3,0)));
176: #else /* ----- */
177: int xbee_snddata(xbee_con *con, char *format, ...);
178: int _xbee_snddata(xbee_hnd xbee, xbee_con *con, char *format, ...);
179: int xbee_vsnddata(xbee_con *con, char *format, va_list ap);
180: int _xbee_vsnddata(xbee_hnd xbee, xbee_con *con, char *format, va_list ap);
181:
182: /* oh and just 'cos windows has rubbish memory management rules... this too */
183: void xbee_free(void *ptr);
184: #endif /* ----- */
185:
186: xbee_pkt *xbee_getpacket(xbee_con *con);
187: xbee_pkt *_xbee_getpacket(xbee_hnd xbee, xbee_con *con);
188: xbee_pkt *xbee_getpacketwait(xbee_con *con);
189: xbee_pkt *_xbee_getpacketwait(xbee_hnd xbee, xbee_con *con);
190:
191: int xbee_hasdigital(xbee_pkt *pkt, int sample, int input);
192: int xbee_getdigital(xbee_pkt *pkt, int sample, int input);
193:
194: int xbee_hasanalog(xbee_pkt *pkt, int sample, int input);
195: double xbee_getanalog(xbee_pkt *pkt, int sample, int input, double Vref);
196:
197: const char *xbee_svn_version(void);
198: const char *xbee_build_info(void);
199:
200: void xbee_listen_stop(xbee_hnd xbee);
201:
202: #ifdef __cplusplus
203: }
204: #endif
205:
206: #endif
```