

```
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:
21: #include "globals.h"
22:
23: int main(int argc, char *argv[]) {
24:     xbee_con *con, *con2;
25:     xbee_pkt *pkt, *p;
26:
27:     if (xbee_setup("/dev/ttyUSB0",57600) == -1) {
28:         perror("xbee_setup()");
29:         exit(1);
30:     }
31:
32:     /*if ((con = xbee_newcon(NULL,'X',xbee_localAT)) == (void *)-1) {
33:         printf("error creating connection...\n");
34:         exit(1);
35:     }
36:
37:     while(1){sleep(10);}
38:
39:     xbee_senddata(con,"CH%c",0x0C);
40:     sleep(1);
41:     xbee_senddata(con,"ID%c%c",0x33, 0x32);
42:     sleep(1);
43:     xbee_senddata(con,"DH%c%c%c%c",0x00,0x00,0x00,0x00);
44:     sleep(1);
45:     xbee_senddata(con,"DL%c%c%c%c",0x00,0x00,0x00,0x00);
46:     sleep(1);
47:     xbee_senddata(con,"MY%c%c",0x00,0x00);
48:     sleep(1);
49:     // SH - read only
50:     // SL - read only
51:     xbee_senddata(con,"RR%c",0x00);
52:     sleep(1);
53:     xbee_senddata(con,"RN%c",0x00);
54:     sleep(1);
55:     xbee_senddata(con,"MM%c",0x00);
56:     sleep(1);
57:     xbee_senddata(con,"NT%c",0x19);
58:     sleep(1);
59:     xbee_senddata(con,"NO%c",0x00);
60:     sleep(1);
61:     xbee_senddata(con,"CE%c",0x00);
62:     sleep(1);
63:     xbee_senddata(con,"SC%c%c",0x1F,0xFE);
64:     sleep(1);
65:     xbee_senddata(con,"SD%c",0x04);
66:     sleep(1);
67:     xbee_senddata(con,"A1%c",0x00);
68:     sleep(1);
69:     xbee_senddata(con,"A2%c",0x00);
70:     sleep(1);
71:     // AI - read only
72:     xbee_senddata(con,"EE%c",0x00);
73:     sleep(1);
74:     //xbee_senddata(con,"KY%c",0x00);
75:     //sleep(1);
76:     xbee_senddata(con,"NIS","TIGGER");
77:     sleep(1);
78:     xbee_senddata(con,"PL%c",0x04);
79:     sleep(1);
80:     xbee_senddata(con,"CA%c",0x2C);
81:     sleep(1);
82:     xbee_senddata(con,"SM%c",0x00);
83:     sleep(1);
84:     xbee_senddata(con,"ST%c%c",0x13,0x88);
85:     sleep(1);
```

```
86: xbee_senddata(con, "SP%c%c", 0x00, 0x00);
87: sleep(1);
88: xbee_senddata(con, "DP%c%c", 0x03, 0xE8);
89: sleep(1);
90: xbee_senddata(con, "SO%c", 0x00);
91: sleep(1);
92: xbee_senddata(con, "BD%c", 0x06);
93: sleep(1);
94: xbee_senddata(con, "RO%c", 0x03);
95: sleep(1);
96: xbee_senddata(con, "AP%c", 0x02);
97: sleep(1);
98: xbee_senddata(con, "PR%c", 0xFF);
99: sleep(1);
100: xbee_senddata(con, "D8%c", 0x00);
101: sleep(1);
102: xbee_senddata(con, "D7%c", 0x01);
103: sleep(1);
104: xbee_senddata(con, "D6%c", 0x00);
105: sleep(1);
106: xbee_senddata(con, "D5%c", 0x01);
107: sleep(1);
108: xbee_senddata(con, "D4%c", 0x00);
109: sleep(1);
110: xbee_senddata(con, "D3%c", 0x00);
111: sleep(1);
112: xbee_senddata(con, "D2%c", 0x00);
113: sleep(1);
114: xbee_senddata(con, "D1%c", 0x00);
115: sleep(1);
116: xbee_senddata(con, "D0%c", 0x00);
117: sleep(1);
118: xbee_senddata(con, "IU%c", 0x00);
119: sleep(1);
120: xbee_senddata(con, "IT%c", 0x01);
121: sleep(1);
122: xbee_senddata(con, "IC%c", 0x00);
123: sleep(1);
124: xbee_senddata(con, "IR%c%c", 0x00, 0x00);
125: sleep(1);
126: xbee_senddata(con, "IA%c%c%c%c%c%c%c%c", 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF);
127: sleep(1);
128: xbee_senddata(con, "T0%c", 0xFF);
129: sleep(1);
130: xbee_senddata(con, "T1%c", 0xFF);
131: sleep(1);
132: xbee_senddata(con, "T2%c", 0xFF);
133: sleep(1);
134: xbee_senddata(con, "T3%c", 0xFF);
135: sleep(1);
136: xbee_senddata(con, "T4%c", 0xFF);
137: sleep(1);
138: xbee_senddata(con, "T5%c", 0xFF);
139: sleep(1);
140: xbee_senddata(con, "T6%c", 0xFF);
141: sleep(1);
142: xbee_senddata(con, "T7%c", 0xFF);
143: sleep(1);
144: xbee_senddata(con, "P0%c", 0x01);
145: sleep(1);
146: xbee_senddata(con, "P1%c", 0x00);
147: sleep(1);
148: xbee_senddata(con, "PT%c", 0xFF);
149: sleep(1);
150: xbee_senddata(con, "RP%c", 0x28);
151: sleep(1);
152: // VR - read only
153: // HV - read only
154: // DB - read only
155: // EC - read only
156: // EA - read only
157: // DD - read only
158: xbee_senddata(con, "CT%c", 0x64);
159: sleep(1);
160: xbee_senddata(con, "GT%c%c", 0x03, 0xE8);
161: sleep(1);
162: xbee_senddata(con, "CC%c", 0x2B);
163: sleep(1);
164:
165: sleep(10);
166: */
167:
168: /* test 64bit IO and Data */
169: con = xbee_newcon('I', xbee_64bitIO, 0x0013A200, 0x403af247);
170: con2 = xbee_newcon('I', xbee_64bitData, 0x0013A200, 0x403af247);
```

```

171:
172: while (1) {
173:     while ((pkt = xbee_getpacket(con)) != NULL) {
174:         if (pkt->Iomask & 0x0001) printf("D0: %c ", ((pkt->Iodata & 0x0001)?'1':'0'));
175:         if (pkt->Iomask & 0x0002) printf("D1: %c ", ((pkt->Iodata & 0x0002)?'1':'0'));
176:         if (pkt->Iomask & 0x0004) printf("D2: %c ", ((pkt->Iodata & 0x0004)?'1':'0'));
177:         if (pkt->Iomask & 0x0008) printf("D3: %c ", ((pkt->Iodata & 0x0008)?'1':'0'));
178:         if (pkt->Iomask & 0x0010) printf("D4: %c ", ((pkt->Iodata & 0x0010)?'1':'0'));
179:         if (pkt->Iomask & 0x0020) printf("D5: %c ", ((pkt->Iodata & 0x0020)?'1':'0'));
180:         if (pkt->Iomask & 0x0040) printf("D6: %c ", ((pkt->Iodata & 0x0040)?'1':'0'));
181:         if (pkt->Iomask & 0x0080) printf("D7: %c ", ((pkt->Iodata & 0x0080)?'1':'0'));
182:         if (pkt->Iomask & 0x0100) printf("D8: %c ", ((pkt->Iodata & 0x0100)?'1':'0'));
183: #define Vref 3.23
184:         if (pkt->Iomask & 0x0200) printf("A0: %.2fv ", (Vref/1024)*pkt->IOanalog[0]);
185:         if (pkt->Iomask & 0x0400) printf("A1: %.2fv ", (Vref/1024)*pkt->IOanalog[1]);
186:         if (pkt->Iomask & 0x0800) printf("A2: %.2fv ", (Vref/1024)*pkt->IOanalog[2]);
187:         if (pkt->Iomask & 0x1000) printf("A3: %.2fv ", (Vref/1024)*pkt->IOanalog[3]);
188:         if (pkt->Iomask & 0x2000) printf("A4: %.2fv ", (Vref/1024)*pkt->IOanalog[4]);
189:         if (pkt->Iomask & 0x4000) printf("A5: %.2fv ", (Vref/1024)*pkt->IOanalog[5]);
190:         printf("\n");
191:         p = xbee_senddata(con2, "the time is %d\r", time(NULL));
192:         free(pkt);
193:         if (p) {
194:             switch (p->status) {
195:                 case 0x01: printf("XBee: txStatus: No ACK\n"); break;
196:                 case 0x02: printf("XBee: txStatus: CCA Failure\n"); break;
197:                 case 0x03: printf("XBee: txStatus: Purged\n"); break;
198:             }
199:             free(p);
200:         }
201:     }
202:     while ((pkt = xbee_getpacket(con2)) != NULL) {
203:         printf("he said '%s'\n", pkt->data);
204:         p = xbee_senddata(con2, "you said '%s'\r", pkt->data);
205:         free(pkt);
206:         if (p) {
207:             switch (p->status) {
208:                 case 0x01: printf("XBee: txStatus: No ACK\n"); break;
209:                 case 0x02: printf("XBee: txStatus: CCA Failure\n"); break;
210:                 case 0x03: printf("XBee: txStatus: Purged\n"); break;
211:             }
212:             free(p);
213:         }
214:     }
215:     usleep(100);
216: }
217:
218: return 0;
219: }

```