

```
1: // {$STAMP BS2p}
2: // {$PBASIC 2.5}
3: // {$PORT COM1}
4:
5: //by W3SZ
6: //Arduino code tested with hardware and shown to be working 8-24-2017
7: // This program is supposed to take band control data from the N3FTI Bandswitch
8: // and use it to set the appropriate transmit and receive if signal levels by
9: // setting programmable attenuators for each band from 50 MHz thru 24 GHz.
10: // The band-select signal is input as a 4 bit binary signal and the logic is set
11: // so that the appropriate signals are then sent to the programmable attenuators for
12: // both the transmit and receive lines.
13:
14: // The input signal matrix is as follows:
15: // Band      A          B          C          D
16: // 50         0          0          0          0
17: // 144        1          0          0          0
18: // 222        0          1          0          0
19: // 432        1          1          0          0
20: // 903        0          0          1          0
21: // 1296       1          0          1          0
22: // 2304       0          1          1          0
23: // 3456       1          1          1          0
24: // 5760       0          0          0          1
25: // 10G        1          0          0          1
26: // 24G        0          1          0          1
27: // 47G        1          1          0          1
28: //
29: // A = LPT pin 2
30: // B = LPT pin 7
31: // C = LPT pin 8
32: // D = LPT pin 9
33: // Grnd = LPT pin 15
34:
35: // Declare and Initialize receive attenuation level variables for each band
36: int RX50 = 10;
37: int RX144 = 15;
38: int RX222 = 20;
39: int RX432 = 25;
40: int RX903 = 30;
41: int RX1296 = 35;
42: int RX2304 = 40;
43: int RX3G = 45;
44: int RX5G = 50;
45: int RX10G = 55;
46: int RX24G = 60;
47:
48: // Declare and Initialize transmit attenuation level variables for each band
49: int TX50 = 0;
50: int TX144 = 2;
51: int TX222 = 4;
52: int TX432 = 8;
53: int TX903 = 16;
54: int TX1296 = 32;
55: int TX2304 = 1;
56: int TX3G = 5;
57: int TX5G = 9;
58: int TX10G = 17;
59: int TX24G = 33;
60:
61: // Declare and initialize input frequency variable from N3FTI Device
62: int FREQ = 0;
63: // FREQ CAN BE
64: // 0 50 MHZ
65: // 1 144 MHZ
66: // 2 222 MHZ
```

```
67: // 3 432 MHZ
68: // 4 903 MHZ
69: // 5 1296 MHZ
70: // 6 2304 MHZ
71: // 7 3G
72: // 8 5G
73: // 9 10G
74: // 10 24G
75:
76: // Declare and initialize RXOUT and TXOUT. These are attenuation levels to be set
77: int RXOUT = 0;
78: int TXOUT = 0;
79:
80: // Declare and initialize control bit variables for Rx
81: int RCV1 = 0;
82: int RCV2 = 0;
83: int RCV4 = 0;
84: int RCV8 = 0;
85: int RCV16 = 0;
86: int RCV32 = 0;
87:
88: // Declare control bit variables for Tx
89: int TX1 = 0;
90: int TX2 = 0;
91: int TX4 = 0;
92: int TX8 = 0;
93: int TX16 = 0;
94: int TX32 = 0;
95:
96: // Define shorthand reference for input pins
97: const int PinA = A0;
98: const int PinB = A1;
99: const int PinC = A2;
100: const int PinD = A3;
101:
102: //define and initialize input pin read variables
103: int A = 0;
104: int B = 0;
105: int C = 0;
106: int D = 0;
107:
108: //Define shorthand reference for output pins
109: const int TxOUT1 =4;
110: const int TxOUT2 =5;
111: const int TxOUT4 =6;
112: const int TxOUT8 =7;
113: const int TxOUT16 =8;
114: const int TxOUT32 =9;
115:
116: const int RxOUT1 =10;
117: const int RxOUT2 =11;
118: const int RxOUT4 =12;
119: const int RxOUT8 =13;
120: const int RxOUT16 =14;
121: const int RxOUT32 =15;
122:
123: void setup() {
124:     // put your setup code here, to run once:
125:     //setup input and output pins
126:     pinMode(PinA, INPUT);
127:     pinMode(PinB, INPUT);
128:     pinMode(PinC, INPUT);
129:     pinMode(PinD, INPUT);
130:
131:     pinMode(TxOUT1, OUTPUT);
132:     digitalWrite(TxOUT1, LOW);
```

```
133: pinMode(TxOUT2, OUTPUT);
134: digitalWrite(TxOUT2, LOW);
135: pinMode(TxOUT4, OUTPUT);
136: digitalWrite(TxOUT4, LOW);
137: pinMode(TxOUT8, OUTPUT);
138: digitalWrite(TxOUT8, LOW);
139: pinMode(TxOUT16, OUTPUT);
140: digitalWrite(TxOUT16, LOW);
141: pinMode(TxOUT32, OUTPUT);
142: digitalWrite(TxOUT32, LOW);
143:
144: pinMode(RxOUT1, OUTPUT);
145: digitalWrite(RxOUT1, LOW);
146: pinMode(RxOUT2, OUTPUT);
147: digitalWrite(RxOUT2, LOW);
148: pinMode(RxOUT4, OUTPUT);
149: digitalWrite(RxOUT4, LOW);
150: pinMode(RxOUT8, OUTPUT);
151: digitalWrite(RxOUT8, LOW);
152: pinMode(RxOUT16, OUTPUT);
153: digitalWrite(RxOUT16, LOW);
154: pinMode(RxOUT32, OUTPUT);
155: digitalWrite(RxOUT32, LOW);
156:
157: Serial.begin(9600);
158: Serial.println("Arduino IF Tx/Rx Automatic Attenuator");
159: Serial.println("Takes BCD data from N3FTI Bandswitch and");
160: Serial.println("sets programmable Attenuators for Tx and Rx 0-63 dB");
161: Serial.println("Derived from Basic Stamp Program for this purpose");
162: Serial.println("also by W3SZ");
163:
164: }
165:
166: void loop() {
167:
168: // Main program loop follows
169:
170: // Read BCD input from N3FTI controller
171: A = digitalRead(PinA);
172: B = digitalRead(PinB);
173: C = digitalRead(PinC);
174: D = digitalRead(PinD);
175:
176: // Calculate band from BCD input
177: FREQ = A + (B*2) + (C*4) + (D*8);
178: /*
179: Serial.println(" ");
180: Serial.println(" ");
181: Serial.println(" ");
182: Serial.println(" ");
183: Serial.println(" ");
184: Serial.println(" ");
185: Serial.println("Starting Loop");
186: Serial.println("FREQUENCY BAND Input is: ");
187: Serial.println(FREQ);
188: */
189: //set RXOUT and TXOUT attenuation levels based on BCD input from N3FTI
190: switch (FREQ)
191: {
192: case 0: {
193:   RXOUT = RX50;
194:   TXOUT = TX50;
195:   break;
196: }
197: case 1: {
198:   RXOUT = RX144;
```

```
199: TXOUT = TX144;
200: break;
201: }
202: case 2: {
203: RXOUT = RX222;
204: TXOUT = TX222;
205: break;
206: }
207: case 3: {
208: RXOUT = RX432;
209: TXOUT = TX432;
210: break;
211: }
212: case 4: {
213: RXOUT = RX903;
214: TXOUT = TX903;
215: break;
216: }
217: case 5: {
218: RXOUT = RX1296;
219: TXOUT = TX1296;
220: break;
221: }
222: case 6: {
223: RXOUT = RX2304;
224: TXOUT = TX2304;
225: break;
226: }
227: case 7: {
228: RXOUT = RX3G;
229: TXOUT = TX3G;
230: break;
231: }
232: case 8: {
233: RXOUT = RX5G;
234: TXOUT = TX5G;
235: break;
236: }
237: case 9: {
238: RXOUT = RX10G;
239: TXOUT = TX10G;
240: break;
241: }
242: case 10: {
243: RXOUT = RX24G;
244: TXOUT = TX24G;
245: break;
246: }
247: case 11: {
248: RXOUT = RX24G;
249: TXOUT = TX24G;
250: break;
251: }
252: default: {
253: RXOUT = RX24G;
254: TXOUT = TX24G;
255: break;
256: }
257: }
258:
259: // DETERMINE RCV and TX output pin levels based on values of RXOUT and TXOUT
260: if (RXOUT >= 32) {
261: RCV32 = 1;
262: RXOUT = RXOUT - 32;
263: }
264: else {
```

```
265: RCV32 = 0;
266: }
267:
268: if (RXOUT >= 16) {
269:   RCV16 = 1;
270:   RXOUT = RXOUT - 16;
271: }
272: else {
273: RCV16 = 0;
274: }
275:
276: if (RXOUT >= 8) {
277:   RCV8 = 1;
278:   RXOUT = RXOUT - 8;
279: }
280: else {
281: RCV8 = 0;
282: }
283:
284: if (RXOUT >= 4) {
285:   RCV4 = 1;
286:   RXOUT=RXOUT - 4;
287: }
288: else {
289: RCV4 = 0;
290: }
291:
292: if (RXOUT >= 2) {
293:   RCV2 = 1;
294:   RXOUT = RXOUT - 2;
295: }
296: else {
297: RCV2 = 0;
298: }
299:
300: RCV1 = RXOUT;
301:
302: if (TXOUT >= 32) {
303:   TX32 = 1;
304:   TXOUT = TXOUT - 32;
305: }
306: else {
307:   TX32 = 0;
308: }
309:
310: if (TXOUT >= 16) {
311:   TX16 = 1;
312:   TXOUT = TXOUT - 16;
313: }
314: else {
315:   TX16 = 0;
316: }
317:
318: if (TXOUT >= 8) {
319:   TX8 = 1;
320:   TXOUT = TXOUT - 8;
321: }
322: else {
323:   TX8 = 0;
324: }
325:
326: if (TXOUT >= 4) {
327:   TX4 = 1;
328:   TXOUT=TXOUT - 4;
329: }
330: else {
```

```
331: TX4 = 0;
332: }
333:
334: if (TXOUT >= 2) {
335:   TX2 = 1;
336:   TXOUT = TXOUT - 2;
337: }
338: else {
339:   TX2 = 0;
340: }
341:
342: TX1 = TXOUT;
343:
344: // Use RCV and TX levels as just determined to set output pin levels
345: digitalWrite(TxOUT1, TX1);
346: digitalWrite(TxOUT2, TX2);
347: digitalWrite(TxOUT4, TX4);
348: digitalWrite(TxOUT8, TX8);
349: digitalWrite(TxOUT16, TX16);
350: digitalWrite(TxOUT32, TX32);
351:
352: digitalWrite(RxOUT1, RCV1);
353: digitalWrite(RxOUT2, RCV2);
354: digitalWrite(RxOUT4, RCV4);
355: digitalWrite(RxOUT8, RCV8);
356: digitalWrite(RxOUT16, RCV16);
357: digitalWrite(RxOUT32, RCV32);
358:
359: /*
360: Serial.println("TxOUT1 is: ");
361: Serial.println(TX1);
362: Serial.println(" ");
363: Serial.println("TxOUT2 is: ");
364: Serial.println(TX2);
365: Serial.println(" ");
366: Serial.println("TxOUT4 is: ");
367: Serial.println(TX4);
368: Serial.println(" ");
369: Serial.println("TxOUT8 is: ");
370: Serial.println(TX8);
371: Serial.println(" ");
372: Serial.println("TxOUT16 is: ");
373: Serial.println(TX16);
374: Serial.println(" ");
375: Serial.println("TxOUT32 is: ");
376: Serial.println(TX32);
377: Serial.println(" ");
378:
379: Serial.println("RxOUT1 is: ");
380: Serial.println(RCV1);
381: Serial.println(" ");
382: Serial.println("RxOUT2 is: ");
383: Serial.println(RCV2);
384: Serial.println(" ");
385: Serial.println("RxOUT4 is: ");
386: Serial.println(RCV4);
387: Serial.println(" ");
388: Serial.println("RxOUT8 is: ");
389: Serial.println(RCV8);
390: Serial.println(" ");
391: Serial.println("RxOUT16 is: ");
392: Serial.println(RCV16);
393: Serial.println(" ");
394: Serial.println("RxOUT32 is: ");
395: Serial.println(RCV32);
396: Serial.println(" ");
```

```
397: */  
398: }  
399:
```