```
' {$STAMP BS2p}
' {$PBASIC 2.5}
' {$PORT COM1}
‘Basic Stamp Code on this side!
' This program is supposed to take band control
‘data from the N3FTI Bandswitch
' and use it to set the appropriate transmit and
‘receive IF signal levels by
' setting programmable attenuators for each band
‘from 50 MHz thru 24 GHz.
' The band-select signal is input as a 4 bit binary
‘signal and the logic is set
' so that the appropriate signals are then sent to the
‘programmable attenuators for
' both the transmit and receive lines.

' The input signal matrix is as follows:
' Band    A       B       C           D
' 50      0       0       0           0
' 144     1       0       0           0
' 222     0       1       0           0
' 432     1       1       0           0
' 903     0       0       1           0
' 1296    1       0       1           0
' 2304    0       1       1           0
' 3456    1       1       1           0
' 5760    0       0       0           1
' 10G     1       0       0           1
' 24G     0       1       0           1
' 47G     1       1       0           1
'
' A = LPT pin 2
' B = LPT pin 7
' C = LPT pin 8
' D = LPT pin 9
' Grnd = LPT pin 15

' Declare attenuation level variables for receive
RX50 VAR Byte
RX144  VAR Byte
RX222 VAR Byte
RX432 VAR Byte
RX903 VAR Byte
RX1296 VAR Byte
RX2304 VAR Byte
RX3G VAR Byte
RX5G VAR Byte
RX10G  VAR Byte
RX24G  VAR Byte
```

```
//Arduino code on this side!
//both by W3SZ
//Arduino code tested with hardware and shown
//to be working 8-24-2017
// This program is supposed to take band control
//data from the N3FTI Bandswitch
// and use it to set the appropriate transmit and
//receive if signal levels by
// setting programmable attenuators for each
//band from 50 MHz thru 24 GHz.
// The band-select signal is input as a 4 bit
//binary signal and the logic is set
// so that the appropriate signals are then sent
//to the programmable attenuators for
// both the transmit and receive lines.

// The input signal matrix is as follows:
// Band    A       B       C           D
// 50      0       0       0           0
// 144     1       0       0           0
// 222     0       1       0           0
// 432     1       1       0           0
// 903     0       0       1           0
// 1296    1       0       1           0
// 2304    0       1       1           0
// 3456    1       1       1           0
// 5760    0       0       0           1
// 10G     1       0       0           1
// 24G     0       1       0           1
// 47G     1       1       0           1
//
// A = LPT pin 2
// B = LPT pin 7
// C = LPT pin 8
// D = LPT pin 9
// Grnd = LPT pin 15

// Declare/Initialize receive attenuation level var
int RX50 = 10;
int RX144 = 15;
int RX222 = 20;
int RX432 = 25;
int RX903 = 30;
int RX1296 = 35;
int RX2304 = 40;
int RX3G = 45;
int RX5G = 50;
int RX10G = 55;
int RX24G = 60;
```

```
' Declare attenuation level variables for transmit
TX50 VAR Byte
TX144 VAR Byte
TX222 VAR Byte
TX432 VAR Byte
TX903 VAR Byte
TX1296 VAR Byte
TX2304 VAR Byte
TX3G VAR Byte
TX5G VAR Byte
TX10G VAR Byte
TX24G VAR Byte

' A Nib is 4 bits
' Declare input frequency var from N3FTI Device
FREQ VAR Nib
' FREQ CAN BE
' 0 50 MHZ
' 1 144 MHZ
' 2 222 MHZ
' 3 432 MHZ
' 4 903 MHZ
' 5 1296 MHZ
' 6 2304 MHZ
' 7 3G
' 8 5G
' 9 10G
' 10 24G

' Declare RXOUT and TXOUT.
RXOUT VAR Byte
TXOUT VAR Byte

' Initialize receive atten level var for each band
RX50 = 00
RX144 = 00
RX222 = 00
RX432 = 16
RX903 = 08
RX1296 = 0
RX2304 = 18
RX3G = 7
RX5G = 8
RX10G = 8
RX24G = 2
```

```
// Declare/Initialize transmit attenuation level vars
int TX50 = 0;
int TX144 = 2;
int TX222 = 4;
int TX432 = 8;
int TX903 = 16;
int TX1296 = 32;
int TX2304 = 1;
int TX3G = 5;
int TX5G = 9;
int TX10G = 17;
int TX24G = 33;

// Declare/initialize input freq var from N3FTI Dev

int FREQ = 0;
// FREQ CAN BE
// 0 50 MHZ
// 1 144 MHZ
// 2 222 MHZ
// 3 432 MHZ
// 4 903 MHZ
// 5 1296 MHZ
// 6 2304 MHZ
// 7 3G
// 8 5G
// 9 10G
// 10 24G

// Declare and initialize RXOUT and TXOUT.
int RXOUT = 0;
int TXOUT = 0;
```

```
' Initialize transmit atten level var for each band
TX50 = 0
TX144 = 17
TX222 = 11
TX432 = 04
TX903 = 13
TX1296 = 0
TX2304 = 2
TX3G = 20
TX5G = 0
TX10G = 0
TX24G = 0

' Declare control bit variables for Rx
RCV1  VAR Bit
RCV2  VAR Bit
RCV4  VAR Bit
RCV8  VAR Bit
RCV16 VAR Bit
RCV32 VAR Bit

' Declare control bit variables for Tx
TX1  VAR Bit
TX2  VAR Bit
TX4  VAR Bit
TX8  VAR Bit
TX16 VAR Bit
TX32 VAR Bit

' Define shorthand reference for input pins
A PIN 0
B PIN 1
C PIN 2
D PIN 3

' Read BCD input from N3FTI controller
INPUT A
INPUT B
INPUT C
INPUT D

'Set pins 4-15 as output pins
OUTPUT 4
OUTPUT 5
OUTPUT 6
OUTPUT 7
OUTPUT 8
OUTPUT 9
```

```
// Declare and ititialize control bit variables for Rx
int RCV1 = 0;
int RCV2 = 0;
int RCV4 = 0;
int RCV8 = 0;
int RCV16 = 0;
int RCV32 = 0;

// Declare control bit variables for Tx
int TX1 = 0;
int TX2 = 0;
int TX4 = 0;
int TX8 = 0;
int TX16 = 0;
int TX32 = 0;

// Define shorthand reference for input pins
const int PinA = A0;
const int PinB = A1;
const int PinC = A2;
const int PinD = A3;

//define and initialize input pin read variables
int A = 0;
int B = 0;
int C = 0;
int D = 0;

//Define shorthand reference for output pins
const int TxOUT1 =4;
const int TxOUT2 =5;
const int TxOUT4 =6;
const int TxOUT8 =7;
const int TxOUT16 =8;
const int TxOUT32 =9;
```

OUTPUT 10
OUTPUT 11
OUTPUT 12
OUTPUT 13
OUTPUT 14
OUTPUT 15

```
const int RxOUT1 =10;
const int RxOUT2 =11;
const int RxOUT4 =12;
const int RxOUT8 =13;
const int RxOUT16 =14;
const int RxOUT32 =15;

void setup() {
  // put your setup code here, to run once:
  //setup input and output pins
pinMode(PinA, INPUT);
pinMode(PinB, INPUT);
pinMode(PinC, INPUT);
pinMode(PinD, INPUT);

pinMode(TxOUT1, OUTPUT);
digitalWrite(TxOUT1, LOW);
pinMode(TxOUT2, OUTPUT);
digitalWrite(TxOUT2, LOW);
pinMode(TxOUT4, OUTPUT);
digitalWrite(TxOUT4, LOW);
pinMode(TxOUT8, OUTPUT);
digitalWrite(TxOUT8, LOW);
pinMode(TxOUT16, OUTPUT);
digitalWrite(TxOUT16, LOW);
pinMode(TxOUT32, OUTPUT);
digitalWrite(TxOUT32, LOW);

pinMode(RxOUT1, OUTPUT);
digitalWrite(RxOUT1, LOW);
pinMode(RxOUT2, OUTPUT);
digitalWrite(RxOUT2, LOW);
pinMode(RxOUT4, OUTPUT);
digitalWrite(RxOUT4, LOW);
pinMode(RxOUT8, OUTPUT);
digitalWrite(RxOUT8, LOW);
pinMode(RxOUT16, OUTPUT);
digitalWrite(RxOUT16, LOW);
pinMode(RxOUT32, OUTPUT);
digitalWrite(RxOUT32, LOW);
}
```

```
' Main program loop follows

DO

' Calculate band from BCD input
FREQ = A + (B*2) + (C*4) + (D*8)

'set RXOUT and TXOUT attenuation levels
‘based on BCD input from N3FTI
SELECT FREQ
CASE = 0
  RXOUT = RX50
  TXOUT = TX50
CASE = 1
  RXOUT = RX144
  TXOUT = TX144
CASE = 2
  RXOUT = RX222
  TXOUT = TX222
CASE = 3
  RXOUT = RX432
  TXOUT = TX432
CASE = 4
  RXOUT = RX903
  TXOUT = TX903
CASE = 5
  RXOUT = RX1296
  TXOUT = TX1296
CASE = 6
  RXOUT = RX2304
  TXOUT = TX2304
CASE = 7
  RXOUT = RX3G
  TXOUT = TX3G
CASE = 8
  RXOUT = RX5G
  TXOUT = TX5G
```

```
void loop() {

// Read BCD input from N3FTI controller
A = digitalRead(PinA);
B = digitalRead(PinB);
C = digitalRead(PinC);
D = digitalRead(PinD);

// Calculate band from BCD input
FREQ = A + (B*2) + (C*4) + (D*8);

//set RXOUT and TXOUT attenuation levels
//based on BCD input from N3FTI
switch (FREQ) {
case  0: {
  RXOUT = RX50;
  TXOUT = TX50;
  break; }
case  1: {
  RXOUT = RX144;
  TXOUT = TX144;
  break; }
case  2: {
  RXOUT = RX222;
  TXOUT = TX222;
  break; }
case  3: {
  RXOUT = RX432;
  TXOUT = TX432;
  break; }
case  4: {
  RXOUT = RX903;
  TXOUT = TX903;
  break; }
case  5: {
  RXOUT = RX1296;
  TXOUT = TX1296;
  break; }
case  6: {
  RXOUT = RX2304;
TXOUT = TX2304;
  break; }
```

```
CASE = 9
  RXOUT = RX10G
  TXOUT = TX10G
CASE = 10
  RXOUT = RX24G
  TXOUT = TX24G
CASE > 10
  RXOUT = RX24G
  TXOUT = TX24G
ENDSELECT

' DETERMINE RCV and TX output pin levels
‘ based on values of RXOUT and TXOUT
IF (RXOUT >= 32) THEN
  RCV32 = 1
  RXOUT = RXOUT - 32
ELSE
RCV32 = 0
ENDIF

IF (RXOUT >= 16) THEN
  RCV16 = 1
  RXOUT = RXOUT - 16
ELSE
RCV16 = 0
ENDIF

IF (RXOUT >= 8) THEN
  RCV8 = 1
  RXOUT = RXOUT - 8
ELSE
RCV8 = 0
ENDIF

IF (RXOUT >= 4) THEN
  RCV4 = 1
  RXOUT=RXOUT - 4
ELSE
RCV4 = 0
ENDIF

  IF (RXOUT >= 2) THEN
  RCV2 = 1
  RXOUT = RXOUT – 2
ELSE
RCV2 = 0
ENDIF

RCV1 = RXOUT
```

```
case  7: {
  RXOUT = RX3G;
  TXOUT = TX3G;
  break; }
case  8: {
  RXOUT = RX5G;
  TXOUT = TX5G;
  break; }
case  9: {
  RXOUT = RX10G;
  TXOUT = TX10G;
  break; }
case  10: {
  RXOUT = RX24G;
  TXOUT = TX24G;
break;  }
case  11: {
  RXOUT = RX24G;
  TXOUT = TX24G;
  break; }}

/ / DETERMINE RCV and TX output pin levels
//  based on values of RXOUT and TXOUT
if (RXOUT >= 32) {
  RCV32 = 1;
  RXOUT = RXOUT – 32; }
else {
RCV32 = 0; }
if (RXOUT >= 16) {
  RCV16 = 1;
  RXOUT = RXOUT – 16; }
else {
RCV16 = 0; }
if (RXOUT >= 8) {
  RCV8 = 1;
  RXOUT = RXOUT – 8; }
else {
RCV8 = 0; }
if (RXOUT >= 4) {
  RCV4 = 1;
  RXOUT=RXOUT – 4; }
else {
RCV4 = 0; }
if (RXOUT >= 2) {
  RCV2 = 1;
  RXOUT = RXOUT – 2; }
else {
RCV2 = 0; }
RCV1 = RXOUT;
```

```
IF (TXOUT >= 32) THEN
  TX32 = 1
  TXOUT = TXOUT - 32
ELSE
  TX32 = 0
ENDIF

IF (TXOUT  >= 16) THEN
  TX16 = 1
  TXOUT = TXOUT - 16
ELSE
  TX16 = 0
ENDIF

IF (TXOUT >= 8) THEN
  TX8 = 1
  TXOUT = TXOUT - 8
ELSE
  TX8 = 0
ENDIF

IF (TXOUT >= 4) THEN
  TX4 = 1
  TXOUT=TXOUT - 4
ELSE
 TX4 = 0
ENDIF

  IF (TXOUT >= 2) THEN
  TX2 = 1
  TXOUT = TXOUT - 2
ELSE
  TX2 = 0
ENDIF

TX1 = TXOUT
```

```
if (TXOUT >= 32) {
  TX32 = 1;
  TXOUT = TXOUT - 32;
}
else {
  TX32 = 0;
}

if (TXOUT  >= 16) {
  TX16 = 1;
  TXOUT = TXOUT - 16;
}
else {
  TX16 = 0;
}

if (TXOUT >= 8) {
  TX8 = 1;
  TXOUT = TXOUT - 8;
}
else {
  TX8 = 0;
}

if (TXOUT >= 4) {
  TX4 = 1;
  TXOUT=TXOUT - 4;
}
else {
 TX4 = 0;
}

if (TXOUT >= 2) {
  TX2 = 1;
  TXOUT = TXOUT - 2;
}
else {
  TX2 = 0;
}

TX1 = TXOUT;
```

```
' Use RCV and TX levels as just determined to
‘ set output pin levels
OUT4 = TX1
OUT5 = TX2
OUT6 = TX4
OUT7 = TX8
OUT8 = TX16
OUT9 = TX32

OUT10 = RCV1
OUT11 = RCV2
OUT12 = RCV4
OUT13 = RCV8
OUT14 = RCV16
OUT15 = RCV32

' Go back to beginning of loop and repeat
LOOP

END
```

```
// Use RCV and TX levels as just determined to
// set output pin levels
digitalWrite(TxOUT1, TX1);
digitalWrite(TxOUT2, TX2);
digitalWrite(TxOUT4, TX4);
digitalWrite(TxOUT8, TX8);
digitalWrite(TxOUT16, TX16);
digitalWrite(TxOUT32, TX32);

digitalWrite(RxOUT1, RCV1);
digitalWrite(RxOUT2, RCV2);
digitalWrite(RxOUT4, RCV4);
digitalWrite(RxOUT8, RCV8);
digitalWrite(RxOUT16, RCV16);
digitalWrite(RxOUT32, RCV32);
}
```