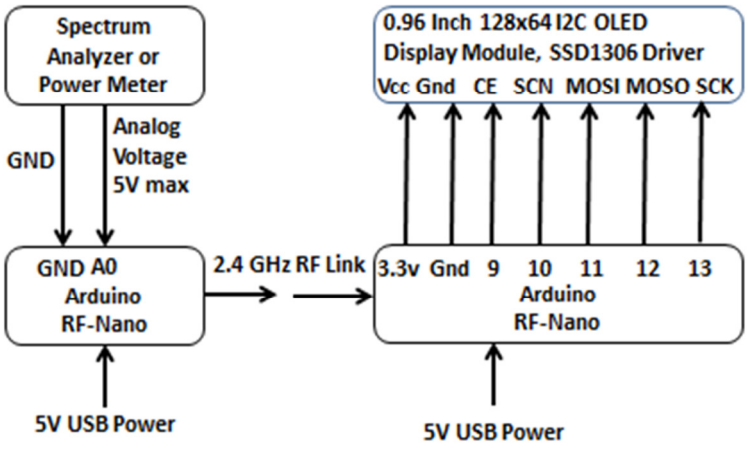
An experimental wireless remote digital display for test equipment providing a analog output – K.Banke N6IZW

Occasionally it might be convenient to have a wireless remote digital display for a power meter or spectrum analyzer that is equipped with an analog output. In particular a remote digital display for an HP8562A spectrum analyzer was needed. The HP8562A has a video output that when set for zero span and single sweep provides a 0-2 volt DC output proportional to the displayed signal level. The equipment consists of a pair of Arduino RF Nano boards along with a 1” Oled display and 2 USB power banks.

The original intended use was to provide a means of pointing/peaking a 30” dish antenna with feed and 5.66 GHz source at a patch antenna connected to the spectrum analyzer some 120’ away for antenna gain and pattern testing of the patch antenna. For this purpose the exact calibration of the wireless display was not needed as the real need was to find the maximum signal level obtainable. As calibrated the digital display agreed within about plus or minus 0.5 dB with the spectrum analyzer display. The accuracy is probably limited by variations of the transmit side analog to digital converter with DC power and temperature



Wireless RF Remote Digital Display

Transmitter Sketch

```
#include <SPI.h>
#include <RF24.h>
#include <nRF24L01.h>

RF24 radio(10, 9);  // CE, CSN
const byte address[10] = "ADDRESS01";
const uint64_t pipe = 0xE8E8F0F0E1LL;
#define v_sensor A0  // voltage sensor connected with A0
float correctionfactor = 0.30;
float vout = 0.0;
float vin = 0.0;
//float data = 0 ;
float data[6];
float value1 = 0.0;
void setup() {
  Serial.begin(9600);
  radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
  pinMode(v_sensor, INPUT);
};
void loop() {
```

```
  // read the value at analog input
   value1 = analogRead(v_sensor);
   vout = (value1 * 5.0) / 1023.0;
   //vin = vout / (R2/(R1+R2));
   //vin = vin - correctionfactor;

data = vout;
radio.write( data, sizeof(data) );
 }
}
```

Receiver Sketch

```
//pins connections
//vcc 3.3
//gnd gnd
//ce pin9
//scn pin10
//sck pin13
//mosi pin11
//moso pin12
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
#include <Arduino.h>

#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#define CE_PIN 10
#define CSN_PIN 9
const uint64_t pipe = 0xE8E8F0F0E1LL;
RF24 radio(CE_PIN, CSN_PIN);
float data[6];  // depending on the number of sensors used

float voltage = 0.000;
float voltagev = 0.000;
void setup() {
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();

  Serial.begin(9600);
  delay(1000);
  Serial.println("Nrf24L01 Receiver Starting");
```

```
  radio.begin();
  radio.openReadingPipe(1, pipe);
  radio.startListening();
}
void loop() {
  int i = 0;
  if (radio.available()) {
    voltage = 0;
    for (i = 0; i < 10; i++) {
      radio.read(data, sizeof(data));
      voltage = data[0] + voltage;
      Serial.println(voltage);
      delay(20);
    }

    voltage = voltage / 10.00;
    //Serial.println(voltage);
    voltagev = voltage;
    voltage = (-voltage + 440) / 4.4;// Change for calibration as needed
    //voltage = (voltage * 5.00) / 1023.00;
    //Serial.println(voltage);
    display.setTextSize(4);
    display.setTextColor(WHITE);
    display.setCursor(5, 15);
    display.println(voltage);
    display.display();
    //delay(500);
    display.clearDisplay();
  }
}
```