AD8307 Modular Power Meter Project

Hardware

Bill of Materials

	Cost	Ship	Sub-total	Source
AD8307 Module	$16.18	$0	$16.18	Ali Express
Nextion 4024 3.2 in module	$21.16	$4.91	$26.07	Ali Express
Veroboard	$8.99	$2.86	$11.85	Ali Express
Arduino Nano	$2.08	$1.04	$3.12	Ali Express
			$57.22	

AD8307 Module

The Analog Device AD8307 chip is designed to convert an RF signal between -75dBm and +17dBm, in the range of 0-500MHz, to a linear voltage representation of the logarithmic input power. This creates a voltage swing at the output of the AD8307 that is suitable for Analog to Digital conversion.

Nextion Display

The TFT LCD display contains the circuitry required to operate as a human to machine interface with only a pair of serial data lines and +5-volt power required for operation.

Arduino Nano clone

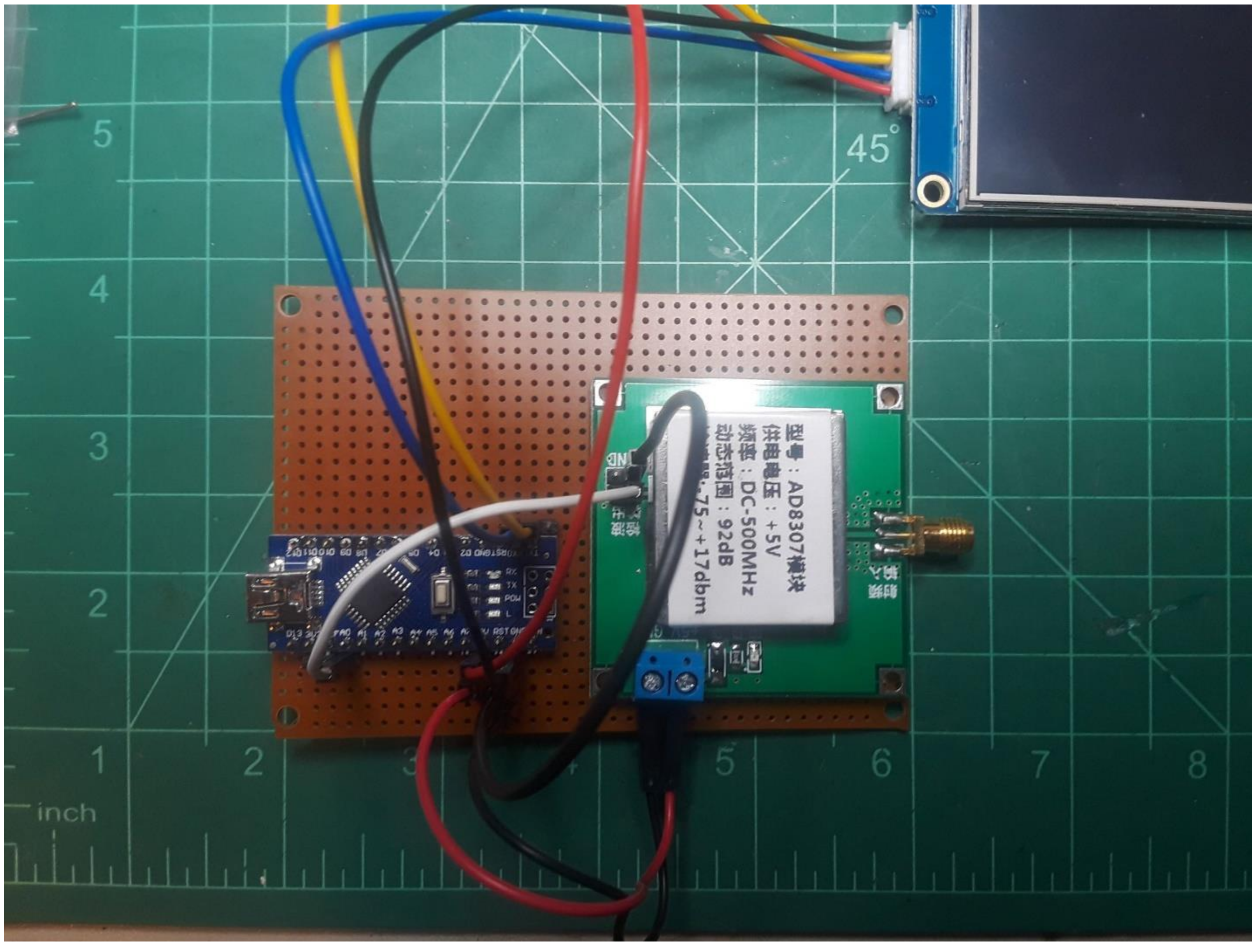
The Arduino is an ideal platform for lightweight data conversion, and serial communications to 1 device. In this project the Nano firmware is coded to convert the analog voltage from the AD8307 to a digital representation suitable for display by the Nextion device.

Wiring

Arduino Pin	Other Module	Color or Designation
A0	AD8307	White in photo
GND	AD8307 signal ground	Black in photo
5V	AD8307 +5V, Nextion Display	Red wire
GND	AD8307 GND, Nextion Display	Black wires
TX1	Nextion Display	Yellow wire
RX0	Nextion Display	Blue wire (optional*)

Connect 2 power and 2 signal wires to the AD8307 module. Create a power bus on the Veroboard and simply connect the Nano pins to their corresponding module pins/wires as noted above. Header strips are supplied for use with the Nextion Display interface cable as well as with the AD8307 module if desired.

*Note that the Blue wire to RX0 on the Nano is only used if/when the Nextion Display is programmed to send serial data ***to*** the Arduino, otherwise only the Yellow signal wire is required along with the +5 V power wires.



Firmware

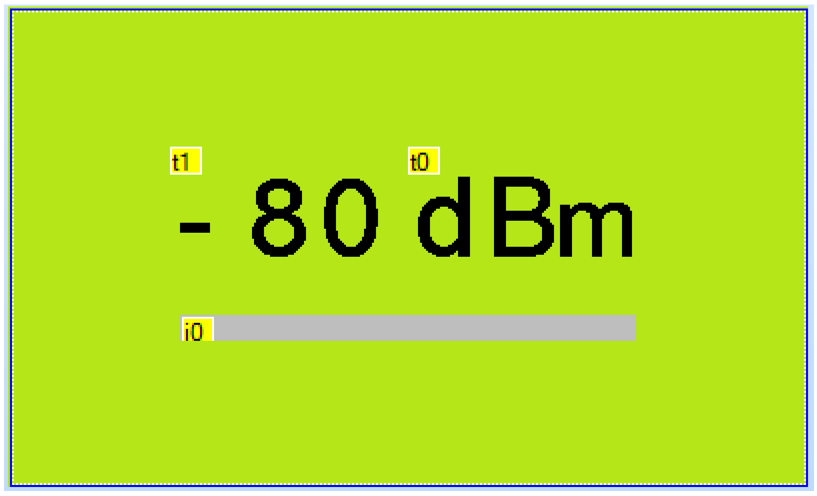
Nextion

The Nextion Human Machine Interface or HMI for this project is the NX4024T032_011 module, a 3.2 inch smart display with 4M RAM running an ARM microprocessor at 48MHz.

The Nextion display is “programmed” using the Nextion Editor application software.

First, download the editor ZIP archive at: https://nextion.tech/resources/download/ then extract the archive to a convenient location. The resulting folder will contain the NextionEditor.exe file, and you can drag/drop a desktop shortcut to this file.

Open the editor and click on the Open icon on the toolbar to display the File > Open dialog. Open the supplied **power-meter.HMI** file, and the Display tab in the center window of the editor should be showing the image below.



The elements of the screen include the background color (46883 in decimal), two Text elements t0 and t1, and the progress bar indicator i0. Text element t0 is initialized to the value “dBm”, and Text element t1 is set to “-80”, and Progress Indicator i0 is set to 0.

In the Arduino software discussed below, these control elements will be referenced in code when commends are sent through the hardware serial interface over to the Nextion display during operation.

Arduino

The purpose of the Arduino Nano code is two-fold:

1. To provide conversion of the voltage output of the AD8307 module to dBm
2. To send to the Nextion display module the real-time dBm and % values for display

The analog to digital conversion of the AD8307 output voltage is accomplished using the AnalogRead() method to copy the digital representation, an Integer value, of the voltage present on the A0 pin of the Nano. This integer value is numerically scaled to provide values for the i0 indicator and the t1 dBm value.

The numeric information is sent to the Nextion display over the hardware serial interface provided by the TX1 pin and GND. The output data is setup using Serial.print() to load the data buffer and Serial.write() to send 0x00 three times in sequence to finalize the command being sent to the Nextion display command interface.

There is a calibration table that allows for adding or subtracting offset values in order for the dBm values being displayed to maintain a linear relationship with respect to the RF input. The calibration table has been setup to tweak specific ranges of dBm values where non-linearities exist in the AD8307 chip output.

The source code is commented and included below for reference, but is available for download at Github here: https://github.com/rlramirez77/AD8307-Power-Meter

```
/*
  AnalogReadSerial adapted for the AD8307 logarithmic sensor connected to A0
*/
// include the ITEAD Nextion library header file ONLY if touch command data will flow TO Arduino
// #include <nextion.h>

int sensorValue = 0;
float volts = 0.0;
float raw8307Voltage = 0.0;
String strValue = "";

// the setup routine runs once when you press reset:
void setup() {
  // initialize HW serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever
void loop() {
  // read the voltage input on analog pin 0:
  sensorValue = analogRead(A0);
  // Convert the digital sensor value to a voltage
  volts = (sensorValue/1023.0) * 5.0;

  //
  // The following code blocks create a serial stream that the Nextion Display firmware will understand
  // and interpret as commands. The conversions to percentage (%) and dBm units is made on the fly
when loading the serial
  // data buffer with Serial.print(...).
```

```
//

// Display bar value
Serial.print("j0.val=");
Serial.print(int(volts * 37.0)-5); // Scale the bar
Serial.write(0xff);
Serial.write(0xff);
Serial.write(0xff);

// Scale the raw voltage into A0
raw8307Voltage = (volts * 39.0)-85.0; // Scale the dBm range

// Serial.print("raw: " + String(raw8307Voltage) + "\n"); // Debug line

//
// Calibration table for 50 MHz, -80 to +17 dBm
//
if (raw8307Voltage > -80.0 && raw8307Voltage < -54.01){// -80 to -58 dBm
  raw8307Voltage = raw8307Voltage * 1.054;
}
if (raw8307Voltage > -54.00 && raw8307Voltage < -44.01){// -57 to -49 dBm
  raw8307Voltage = raw8307Voltage * 1.083;
}
if (raw8307Voltage > -44.0 && raw8307Voltage < -35.01){// -48 to -40 dBm
  raw8307Voltage = raw8307Voltage * 1.12;
}
if (raw8307Voltage > -35.0 && raw8307Voltage < -29.51){// -39 to -35 dBm
  raw8307Voltage = raw8307Voltage * 1.155;
}
```

```
if (raw8307Voltage > -29.5 && raw8307Voltage < -25.51){// -35 to -32 dBm
  raw8307Voltage = raw8307Voltage * 1.21;
}
if (raw8307Voltage > -25.5 && raw8307Voltage < -22.51){// -31 to -29 dBm
  raw8307Voltage = raw8307Voltage * 1.24;
}
if (raw8307Voltage > -22.5 && raw8307Voltage < -20.01){// -28 to -27 dBm
  raw8307Voltage = raw8307Voltage * 1.27;
}
if (raw8307Voltage > -20.0 && raw8307Voltage < -15.61){// -26 to - 22 dBm
  raw8307Voltage = raw8307Voltage * 1.355;
}
if (raw8307Voltage > -15.6 && raw8307Voltage < -13.61){// -26 to - 22 dBm
  raw8307Voltage = raw8307Voltage * 1.47;
}
if (raw8307Voltage > -13.6 && raw8307Voltage < -11.01){// -21 to -17 dBm
  raw8307Voltage = raw8307Voltage * 1.62;
}
if (raw8307Voltage > -11.0 && raw8307Voltage < -7.51){ // -16 to -14 dBm
  raw8307Voltage = (raw8307Voltage * 1.1) - 6.8;
}
if (raw8307Voltage > -7.5 && raw8307Voltage < -3.1){ // -13 to -11 dBm
  raw8307Voltage = (raw8307Voltage * 1.1) - 7.0;
}
if (raw8307Voltage > -3.0 && raw8307Voltage < -1.1){ // -11 to -10 dBm
  raw8307Voltage = (raw8307Voltage * 1.1) - 8.0;
}
if (raw8307Voltage > -1.0 && raw8307Voltage < 4.99){ // -9 to -4 dBm
  raw8307Voltage = (raw8307Voltage * 1.1) - 9.0;
```

```
 }
 if (raw8307Voltage > 5.0 && raw8307Voltage < 7.99){ // -4 to -1 dBm
  raw8307Voltage = (raw8307Voltage * 1.1) - 10.0;
 }
 if (raw8307Voltage > 8.0 && raw8307Voltage < 12.99){ // 0 to +3 dBm
  raw8307Voltage = raw8307Voltage - 9.0;
 }
 if (raw8307Voltage > 13.00 && raw8307Voltage < 27.00){ // +4 to +17 dBm
  raw8307Voltage = raw8307Voltage - 9.5;
 }

 //Serial.print("new: " + String(raw8307Voltage) + "\n"); // Debug line

 // Display dBm value
 strValue = String(raw8307Voltage);
 Serial.print("t1.txt=");
 Serial.print("\"" + strValue + "\"");
 Serial.write(0xff);
 Serial.write(0xff);
 Serial.write(0xff);

 // Delay in milliseconds to keep the update rate reasonable and not overrun the serial data buffer
 delay(100);
}
```

Use

Power

Power should be either supplied using a suitable mini-USB cable, or by wiring the +5V and GND bus rails you created on the Veroboard to a +5V wall wart. One advantage of using the USB cable is that external battery packs for USB devices can be used to operate the meter on battery power. Total current draw was measured to be just over **130 mA** under USB operation.

Signal

The SMA connector on the AD8307 module is used for direct RF input, 0-500 MHz, -80 to +17 dBm.