**4-20ma Generator/Tester Using Arduino**

By [PieterB14](https://www.instructables.com/member/PieterB14/) in [Circuits](https://www.instructables.com/circuits/)[Arduino](https://www.instructables.com/circuits/arduino/projects/)

5,424

8

[6](https://www.instructables.com/4-20ma-GeneratorTester-Using-Arduino/#ible-comments)

license

Save PDFFavorite

**Introduction: 4-20ma Generator/Tester Using Arduino**

[](https://content.instructables.com/FK6/UKSU/KHJ4P7T4/FK6UKSUKHJ4P7T4.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=b331db2f0c1df2c4390caeea1186c4ab)

4-20mA generators are available on ebay, but I for one love the DIY part of things and using parts I have laying around.

I wanted to test our PLC’s analog inputs to verify our scada readings and to test the output of 4-20mA instruments. There are loads of current to voltage converters and voltage to current converters for arduino on ebay, they need calibration though. I can use this to calibrate any of those converters found on ebay and the likes.

I decided that I will DIY a generator and tester. At this point in time it is still a work in progress and a prototype.

I had an old 2.1 sound system that was not being used (small speakers). So I used one of the speaker boxes as an enclosure. I also had an amplifier that died because of lightning, I removed the speaker terminal from that amplifier to make connecting up a breeze. I intend to make a PCB in the future and a better enclosure.

**Supplies**

Parts List.

LCD // 20x4 (adapt code if yours is smaller)

LM7808 // 8volt regulator

LED // Any type or size

Resistor for LED // Appropriate for the LED type and 8volt

100 ohm resistor + 47 ohm resistor in series // Will be used as shunt resistor

10K resistor // Arduino analog in protection against high voltage

22K resistor // To stop A0 from floating

Trimpot 100 ohm + 47 ohm resistor in series // PT100 simulator

35 volt capacitor // I used 470uF, just to keep supply voltage fluctuations down

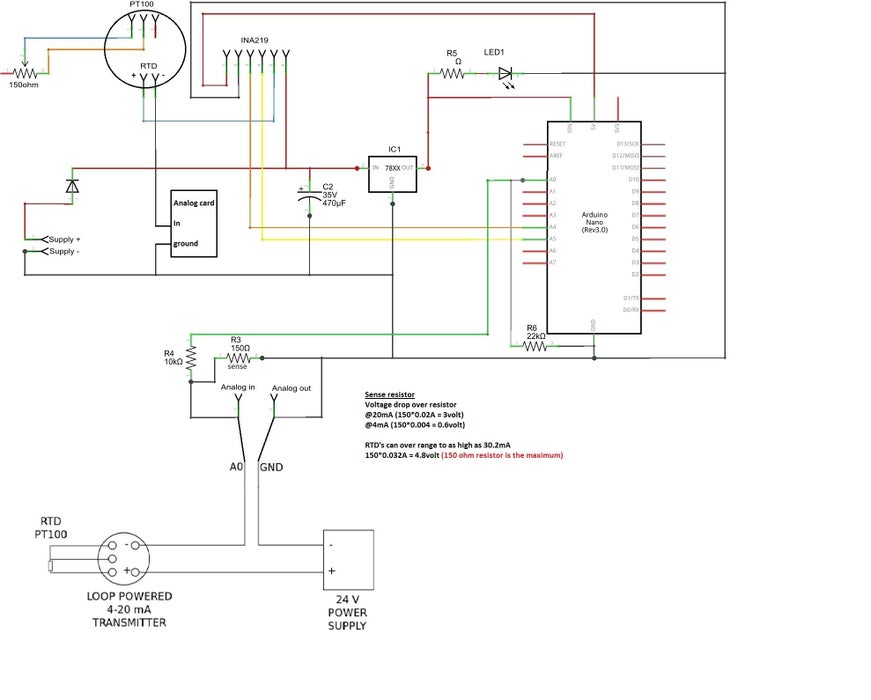
RTD (PT100 transducer) //Span does not matter (range)

DIODE (for polarity protection)

INA219

Arduino

**Step 1:**

[](https://content.instructables.com/F48/IG2S/KHJ4P7H5/F48IG2SKHJ4P7H5.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=1573558d9ef692c4f6ebdf02de862f05)

Following the schematic should get you started on where to add the parts and wire them.

The LM7808 allows a maximum of 25volts input which is fine for PLC systems, they generally use 24volt power supplies. Add a heatsink to the regulator and do not use it for extended periods. Dropping 16volts causes the regulator to generate a lot of heat.

The input supply feeds the regulator and connects to the INA219 VIN, in this configuration the INA219 will also be able to measure the correct supply voltage minus the voltage drop from the diode. You should measure your diode voltage drop and add it to the code so that you get the correct supply voltage readout.

From the INA219 VOUT to the RTD+ powers up the RTD. RTD- to ground completes the circuit.

To test a PLC analog card you would connect RTD- to the input on the analog card and the ground from the card to arduino ground. (Make sure to disconnect any instrument attached to the channel being tested).

R5 and LED1, indicating system is powered up.

The regulator feeds into the arduino VIN (arduino has built in regulator to 5volts).

Arduino 5V pin goes to INA219 to power the on-board chip. INA219 GND to arduino ground.

Trim pot wiper to RTD PIN1 and Trim pot pin 3 to RTD pin 2 will simulate a PT100 connection. (Swap the wires if turning the trim pot clockwise does not increase the mA).

**Step 2: Instrument Output Test**

To test instrument output extra parts is needed, like a shunt resistor. Normal 0.25W resistors will do the job just fine. You can leave the shunt resistor and add a second INA219 to test instrument output. I only had one left so I used a resistor instead.

Testing using a shunt can only be done on the device’s negative side. If you use the positive side you will supply your arduino with more than 4 times the allowed voltage and let the smoke out.

Add the shunt resistor in series with negative wire of the instrument. The side of the shunt closest to the device will become the positive analog for arduino. The other side of the shunt closest to the power supply will become the arduino ground completing the analog input circuit.

150 ohm shunt resistor is the absolute maximum that should be used when using an arduino. The resistor has a voltage drop linear to the mA flowing through it. The greater the mA the greater the voltage.

At 20mA current # 150ohm\*0.02A = 3volt to arduino.

At 4mA current # 150ohm\*0.004A = 0.6volt to arduino.

Now you might want the voltage to be closer to 5 volts so that you can us the full ADC range of the arduino. (Not a good idea).

RTD’s can reach 30.2mA output (Mine does). 150ohm\*0.03A = 4.8volts. That is as close as I would like to be.

Another website indicated to use a 250ohm resistor.

At 20mA current # 250ohm\*0.02A = 5volt to arduino.

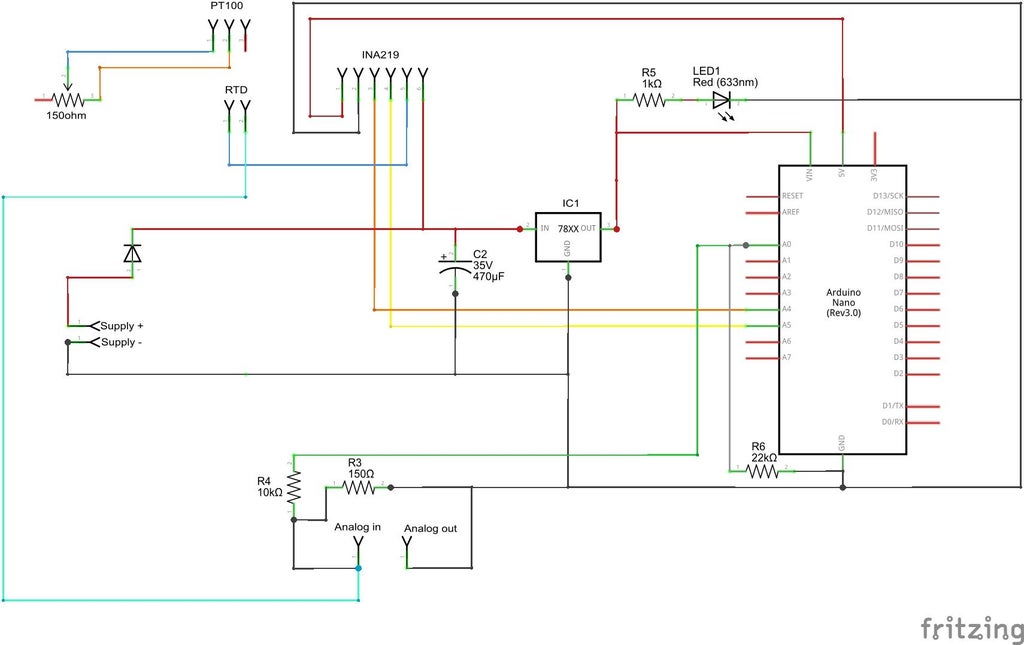
At 30mA current # 250ohm\*0.03A = 7.5volt to arduino.

You risk burning your ADC and arduino.

To test an instrument out in the field, take a 12volt battery with you and connect it to the supply input. Using an external power source will not influence the current PLC setup.

To test an analog input card in the field, take a 12volt battery with you. Disconnect the instrument + from the circuit. Connect ground to instrument ground and the RTD- to the disconnected instrument wire.

**Step 3: Calibration**

[](https://content.instructables.com/FVV/YHRW/KHJ4P7H8/FVVYHRWKHJ4P7H8.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=24591fe1a3053d3327e03b40e5c13d4c)

To calibrate your shunt resistor reading, wire RTD- to the shunt Analog in. Set your trim pot so that the generated mA is 4mA. If your device mA is not equal then modify the first value in the code at line 84. Increasing this value will lower the mA readout.

Then set your trim pot to generate 20mA. If your device mA is not equal then modify the second value in the code at line 84.

So your 4-20mA will now become 0.6-3volts (theoretical). More than enough range. Using the library from eRCaGuy, oversampling will give you a better and stable readout.

Hopefully you read this. This is my first instructable, so please take it easy if I have made a mistake somewhere or left something out.

This project is probably not the best way to go about it, but it works for me and was fun doing it.

Some ideas I have extra…

Add a servo to rotate the trim pot inside the box.

Add push buttons to rotate servo left or right.

Add a digital temperature sensor to the regulator heatsink to warn for dangerous heat.

**Step 4: Programming Arduino**

#include <Wire.h>

// #include <LiquidCrystal\_SR.h> //Uncomment if you use an LCD with a shift register.

#include <LiquidCrystal.h>

#include <TimeLib.h>

#include <Adafruit\_INA219.h>

#include <eRCaGuy\_NewAnalogRead.h>

//A4 = (SDA)

//A5 = (SCL)

Adafruit\_INA219 ina219;

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// LiquidCrystal\_SR lcd(3, 4, 2); //Uncomment if you use an LCD with a shift register.

// | | |\_\_\_\_ Latch Pin

// | \\_\_\_\_\_\_ Clock Pin

// \\_\_\_\_\_\_\_\_ Data/Enable Pin

byte bitsOfResolution = 12; //commanded oversampled resolution

unsigned long numSamplesToAvg = 20; //number of samples AT THE OVERSAMPLED RESOLUTION that you want to take and average

ADC\_prescaler\_t ADCSpeed = ADC\_DEFAULT;

unsigned long previousMillis = 0;

float shuntvoltage = 0.0; //From INA219

float busvoltage = 0.0; //From INA219

float current\_mA = 0.0; //From INA219

float loadvoltage = 0.0; //From INA219

float arduinovoltage = 0.0; //Voltage calculation from A0 pin

Unsigned long A0analogReading = 0;

byte analogIn = A0;

float ma\_mapped = 0.0; //Map voltage from A0 to 4-20mA

void setup() {

adc.setADCSpeed(ADCSpeed);

adc.setBitsOfResolution(bitsOfResolution);

adc.setNumSamplesToAvg(numSamplesToAvg);

uint32\_t currentFrequency;

ina219.begin();

ina219.setCalibration\_32V\_30mA(); //Modified library for more precision on mA

lcd.begin(20, 4); // initialize the LCD

lcd.clear();

lcd.home (); // go home

lcd.print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

delay(2000);

lcd.clear();

}

void loop()

{

unsigned long currentMillis = millis();

const long interval = 100;

//&&&&&&&&&&&&&&&&&

Read I2C devices at intervals and do some calculations

&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

if (currentMillis - previousMillis >= interval) {

previousMillis = currentMillis;

Interval();

}

Print\_To\_LCD(); // I probably don’t need to update the LCD this fast and can be moved to below Interval()

}

void

Interval() {

shuntvoltage = ina219.getShuntVoltage\_mV();

busvoltage = ina219.getBusVoltage\_V();

current\_mA = ina219.getCurrent\_mA();

loadvoltage = (busvoltage + (shuntvoltage / 1000)) + 0.71; // +0.71 is my diode voltage drop

A0analogReading = adc.newAnalogRead(analogIn);

arduinovoltage = (5.0 \* A0analogReading); //Calculated to mV

ma\_mapped = map(arduinovoltage, 752, 8459, 30, 220) / 10.0; //Map cannot use floats. Add a 0 behind the mapped value and divide by 10 to get float readout.

//Mapping from voltage calculation gives more stable readout then using the raw adc reading.

if (shuntvoltage >= -0.10 && shuntvoltage <= -0.01 ) //With no load the INA219 tends to read below -0.01, well mine does.

{

current\_mA = 0;

busvoltage = 0;

loadvoltage = 0;

shuntvoltage = 0;

}

}

void

Print\_To\_LCD() {

lcd.setCursor(0, 0);

if (ma\_mapped < 1.25) { //With no current this is my mA reading, so I just chuck it away.

lcd.print("\* 4-20mA Generator \*");

}

else {

lcd.print("\*\* Analog Tester \*\*");

}

lcd.setCursor(0, 1);

lcd.print("Device :");

lcd.setCursor(10, 1);

if (ma\_mapped < 1.25) {

lcd.print("no device ");

}

else {

lcd.print(ma\_mapped);

}

lcd.print("mA ");

lcd.setCursor(0, 2);

lcd.print("Generate:");

lcd.setCursor(10, 2);

lcd.print(current\_mA);

lcd.print("mA ");

lcd.setCursor(0, 3);

lcd.print("Supply :");

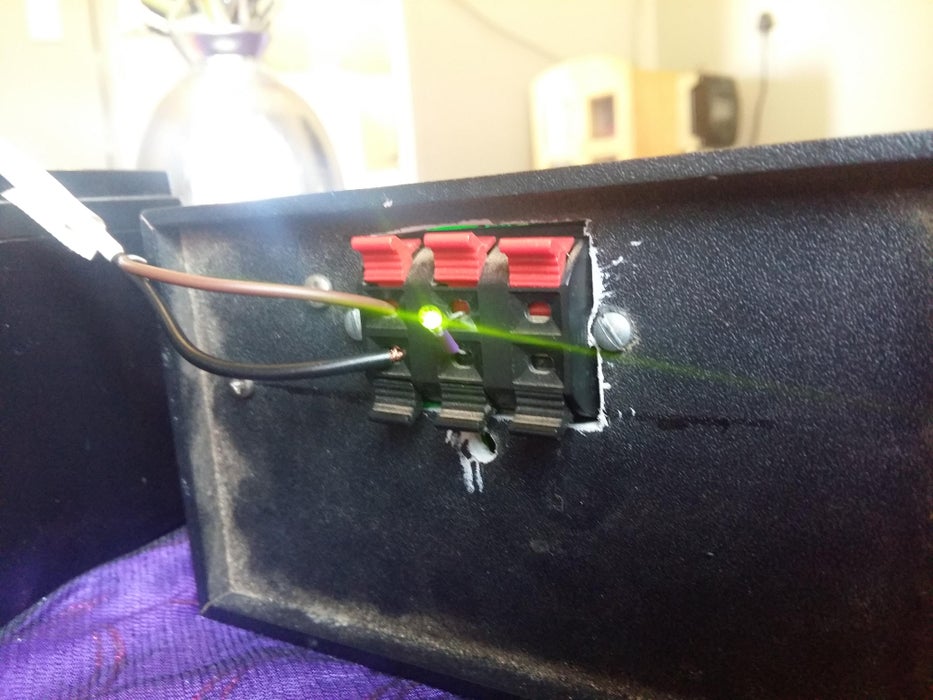
lcd.setCursor(10, 3);

lcd.print(loadvoltage);

lcd.print("V ");

}

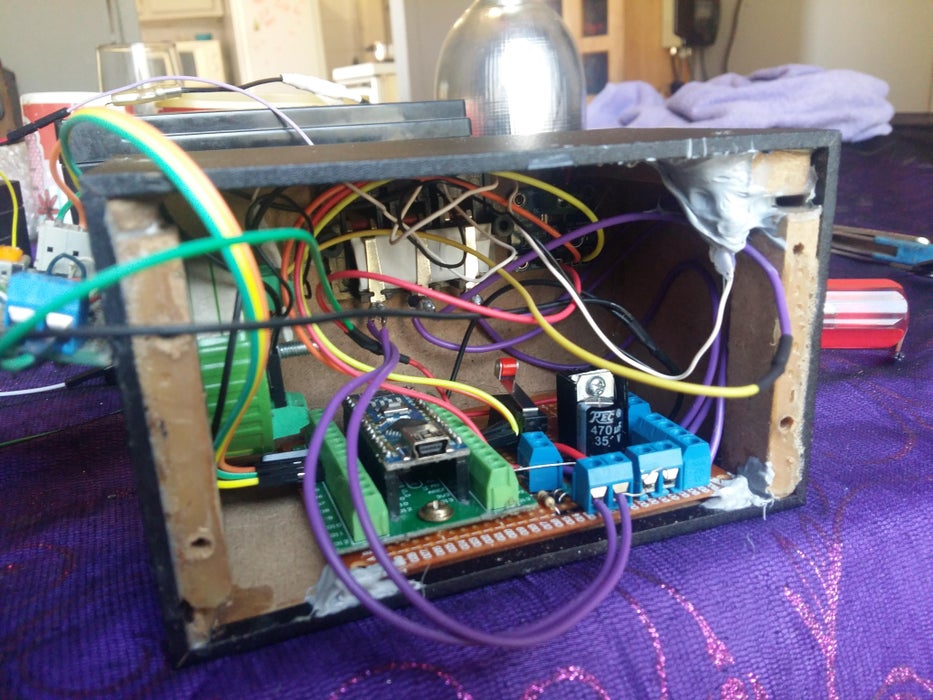
**Step 5: Some More Photos**

[](https://content.instructables.com/FQN/GW1O/KHJ4P7TP/FQNGW1OKHJ4P7TP.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=fed9c53118d8c95b66e282cb9f14ced0)

Amplifier speaker terminal. LED driven by the current generator (RTD). Analog card wiring will replace the LED.

Terminal on the far left is for supply input. Terminals on the right are for instrument input.

**Step 6: Fitting In**

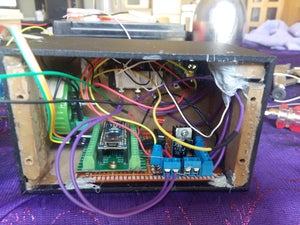
[](https://content.instructables.com/F0X/RTF0/KHJ4P7T8/F0XRTF0KHJ4P7T8.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=bcafde192c642efa1f95a4ce728c96e1)

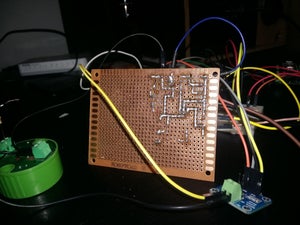
Everything seems to fit. I used silicone to temporarily hold some stuff together. The trim pot is siliconed to the top right. A small hole was predrilled. I can adjust the current from the top of the box.

**Step 7: Just Photos**

[](https://content.instructables.com/FCJ/FXCG/KHJ4P7TO/FCJFXCGKHJ4P7TO.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=7630032ed1620bc09d7e027b88c9189a)

[](https://content.instructables.com/F9Z/S86F/KHJ4P7TN/F9ZS86FKHJ4P7TN.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=dc5ff79b453c62572f4f2424a88beae3)

[](https://content.instructables.com/F1X/294V/KHJ4P7T9/F1X294VKHJ4P7T9.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=e94d8e348c118daa00daa2a9cdb653ad)

[](https://content.instructables.com/FSO/AWEG/KHJ4P7T0/FSOAWEGKHJ4P7T0.jpg?auto=webp&frame=1&width=1024&height=1024&fit=bounds&md=35359bc7981c1fd15848eb85e98f9c26)

**Step 8: Final Words**

I have tested this device's output with an Allan Bradley PLC. The results was very good. I got full range.  
I have also tested this device with 4-20mA pressure sensor that has a built in LCD display. Again results was very good. My readings was off by a couple of decimals.

I write my arduino code in tabs. In PLC's they are called sub routines. Makes debugging easier for my.

Attached are text files of those tabs.

**Attachments**

* [download {{ file.name }}**Calibrate.ino**](https://content.instructables.com/FX7/DL85/KHJ4P8CN/FX7DL85KHJ4P8CN.ino)

[Download](https://content.instructables.com/FX7/DL85/KHJ4P8CN/FX7DL85KHJ4P8CN.ino)

* [download {{ file.name }}**Current\_Generator\_Final.ino**](https://content.instructables.com/FMW/N4GQ/KHJ4P8CO/FMWN4GQKHJ4P8CO.ino)

[Download](https://content.instructables.com/FMW/N4GQ/KHJ4P8CO/FMWN4GQKHJ4P8CO.ino)

* [download {{ file.name }}**Interval.ino**](https://content.instructables.com/FPM/YO3F/KHJ4P8CP/FPMYO3FKHJ4P8CP.ino)

[Download](https://content.instructables.com/FPM/YO3F/KHJ4P8CP/FPMYO3FKHJ4P8CP.ino)

* [download {{ file.name }}**LCD\_Print.ino**](https://content.instructables.com/F0O/FP26/KHJ4P8CQ/F0OFP26KHJ4P8CQ.ino)

[Download](https://content.instructables.com/F0O/FP26/KHJ4P8CQ/F0OFP26KHJ4P8CQ.ino)

* [download {{ file.name }}**Save\_to\_eeprom.ino**](https://content.instructables.com/F1W/NHGS/KHJ4P8CR/F1WNHGSKHJ4P8CR.ino)

[Download](https://content.instructables.com/F1W/NHGS/KHJ4P8CR/F1WNHGSKHJ4P8CR.ino)

* [download {{ file.name }}**Adafruit\_INA219-master.rar**](https://content.instructables.com/F4Q/GANV/KLDUOJEX/F4QGANVKLDUOJEX.rar)

[Download](https://content.instructables.com/F4Q/GANV/KLDUOJEX/F4QGANVKLDUOJEX.rar)

* [download {{ file.name }}**eRCaGuy\_NewAnalogRead.rar**](https://content.instructables.com/FKQ/PZL8/KLDUOJFR/FKQPZL8KLDUOJFR.rar)