
Nixie Thermometer Shield for Arduino

V1.2



Safety Notes

This circuit design includes a switch-mode voltage converter which generates 170 VDC. You are responsible for the safety during the assembly and operation of this device. **DO NOT USE IF YOU DON'T KNOW HOW TO HANDLE HIGH VOLTAGES.** All assembly and safety instructions should be read carefully before the device is operated.

Disclaimer

The information in this document is provided "as is" without any warranty of any kind and is subject to change without notice. You are responsible for the safety during the assembly and operation of this device. "My Electronics" takes no responsibility for any damage to property, personal injury or death. In no event shall "My Electronics" be liable to you or any third parties for any special, indirect, or consequential, or any other damages resulting from assembly or operation of this kit. When you buy the kit you agree with the disclaimer.

All applicable UL, IEC/IEEE, VDE and local regulations must be considered.

Copyright

The schematics and source codes are open for buyers and can be modified for personal and educational purposes. The commercial usage is restricted. The distribution of the instructions, schematics, source codes and the PCB design is not allowed without permission by "My Electronics".

Product Description

The Nixie Thermometer Shield allows you to build a two-digit thermometer, including a minus sign, with your Arduino. It is a modular system consisting of a High Voltage (HV) Power Supply and a Socket Module depending on the Nixie tubes you want to use. The included temperature sensor and an optional second temperature sensor can be used to measure the in- and outside temperature. The kit can be used to getting started with Nixie tubes and 1-wire digital temperature sensors.

Features

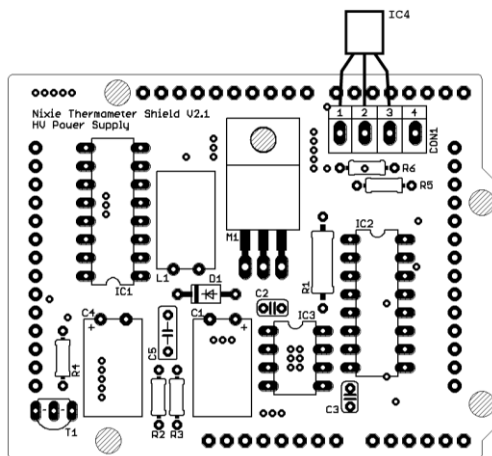
- Compatible with Arduino UNO and MEGA
- Modular system consisting of a HV Power Supply and a Socket Module for different types of Nixie tubes
- Neon lamp NE-2 as minus sign (Optional)
- Included temperature sensor (DS18B20) with a measuring range from -55°C to +85°C
- Supports up to two temperature sensors
- Very high efficiency using the MAX1771 switch-mode controller
- Easy to build, no SMD parts
- Programmable using the Arduino IDE

Assembly Instructions

To build this kit, you should know how to solder. If you have never soldered before, we recommend the [Soldering is Easy](#) tutorial.

HV Power Supply

Board Layout



Parts List

Qty.	Part	Value/Description
1	R1	0.1 Ω , 2 W
1	R2	13.3 k Ω
1	R3	1.5 M Ω
1	R4	68 k Ω
2	R5, R6	4.7 k Ω
1	D1	UF4004
1	M1	IRF630
1	L1	100 μ H, 1 A
1	C1	220 μ F, 25 V
2	C2, C3	100 nF
1	C4	4.7 μ F, 250 V
1	C5	100 nF, 250 V
2	IC1, IC2	K155ID1
1	IC3	MAX1771
1	IC4	DS18B20
1	T1	MPSA42
1	CON1	4-Pin Screw Connector
1		IC Socket DIP 8
2		IC Socket DIP 16
1		Male Header 1x6
2		Male Header 1x8
1		Male Header 1x10
2		Female Header 1x16
1		Pan Head Screw M3 + Lock Nut
1		Piece of Electric Tape

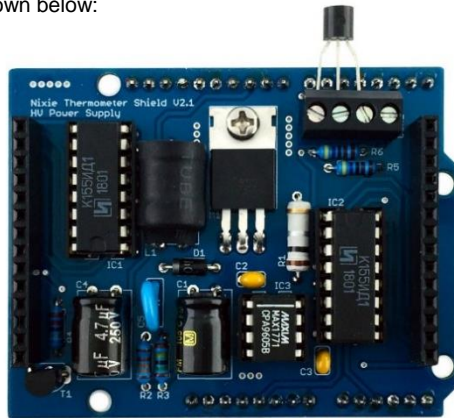
First, check if all listed parts are included in your package. We recommend to begin soldering the parts with the flattest design. Start with the resistors, followed by the capacitors C2, C3 and the diode D1. Then solder the MOSFET M1, the IC sockets and the transistor T1. Last, solder the capacitors C1, C4, C5, the inductivity L1, the screw connector CON1 and the male and female headers.

NOTE: The diode D1, the capacitors C1, C4 and the ICs need to be soldered/mounted in the right direction as indicated on the PCB.

The table below lists the required resistors and their corresponding color codes:

Part	Value	1st Band	2nd Band	3rd Band	4th Band	5th Band
R1	0.1 Ω	Brown	Black	Silver	Gold	-
R2	13.3 k Ω	Brown	Orange	Orange	Red	Brown
R3	1.5 M Ω	Brown	Green	Black	Yellow	Brown
R4	68 k Ω	Blue	Gray	Black	Red	Brown
R5	4.7 k Ω	Yellow	Violet	Black	Brown	Brown

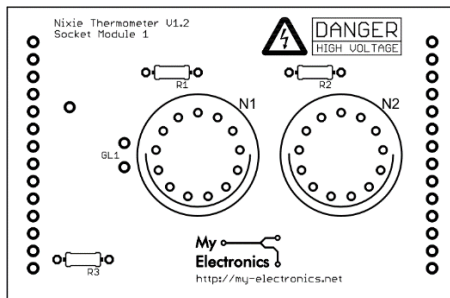
Finally, connect the temperature sensor (IC4) to the terminals 1,2, and 3 of the screw connector (CON1) with its flat side facing down (GND of IC4 needs to be connected to terminal 3). The assembled PCB is shown below:



NOTE: To prevent a short circuit of the shields left socket module header cover the USB connector of your Arduino with the piece of electrical tap included the kit and cut the pins on the backside of the shield as short as possible.

Socket Module

Board Layout


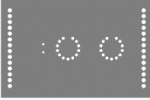
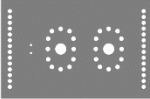





Parts List

Qty.	Part	Value/Description
2	R1-R2	See „List of Socket Modules“ on Page 6
1	R3	180 kΩ
2		Pin Header 1x16
1		Fuse Holder (Ø 5 mm) + Silver Plated Wire (50 mm)

The front side of the Nixie tubes are indicated by semicircles. To mount the Neon lamp, first cut the pins of the fuse holder with a side cutter. Then, solder a silver plated wire from the rear and bend the wires of the Neon lamp at a right angle. Finally, solder the three wires into the board (See the cover image for reference).

List of Socket Modules

Socket Module	Supported Nixie Tubes	Resistor
1 	IN-14	10 kΩ
2 	Z570/3/4M, Z5700/30/40M, ZM1080/2, ZM1134/5/6/8, B570M, F9080B, TAF1093A, TAF1317A	16 kΩ
3 	IN-12, IN-12A/B, IN15A/B, ZM1180/81/82/83/86/88, ZM1162	8.2 kΩ
4 	IN-8	8.2 kΩ
5 	IN-16	16 kΩ
6 	IN-8-2	8.2 kΩ

Value	1st Band	2nd Band	3rd Band	4th Band	5th Band
8.2 kΩ	Gray	Red	Black	Brown	Brown
10 kΩ	Brown	Black	Black	Red	Brown
16 kΩ	Brown	Blue	Black	Red	Brown
180 kΩ	Brown	Gray	Black	Orange	Brown

Operation

To use the shield, mount the HV Power Supply on top of an Arduino board and the Socket Module on top of the HV Power Supply.

IMPORTANT: Before operating the shield connect a 9 to 12 V external power supply to the Arduino board.

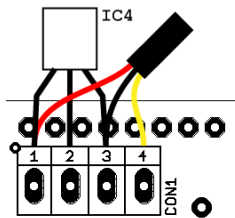
Then, connect the Arduino to your computer via USB as usual and open the sample sketch using the Arduino IDE. The sample sketch is available on our GitHub repository: [Download Sample Sketch](#)

If you are using the Arduino IDE for the first time, we recommend the [Getting Started with Arduino](#) guide.

The sample sketch requires the OneWire library to function. To install the library, in the Arduino IDE, open Sketch → Include Library → Manage Libraries... and search for “OneWire”.

If the sample sketch was uploaded without any error and the kit was assembled properly, the Nixie tubes should display the temperature measured by the temperature sensor (IC4). If the kit does not operate as expected, see the **Troubleshooting** section. The function of the sample sketch is explained in detail in the **Functional Description** section.

The shield supports up to two temperature sensors. A second temperature sensor can be connected via terminal 4 of the screw connector (CON1). A sample configuration connecting a second, wired temperature sensor is shown below:



The wired temperature sensor is not included in the kit and has to be purchased separately. It is available at our eBay store and also elsewhere online.

Wired DS18B20 Temperature Sensor



Note that, due to the heat dissipation of the other components, the temperature reading of the temperature sensor IC4 can be up to 3°C higher than expected. A more precise temperature reading can be obtained using a wired temperature sensor.

WARNING: The assembled kit generates HIGH VOLTAGES. If you buy the kit, you are fully responsible for the safety during the assembly and operation of this device.

Troubleshooting

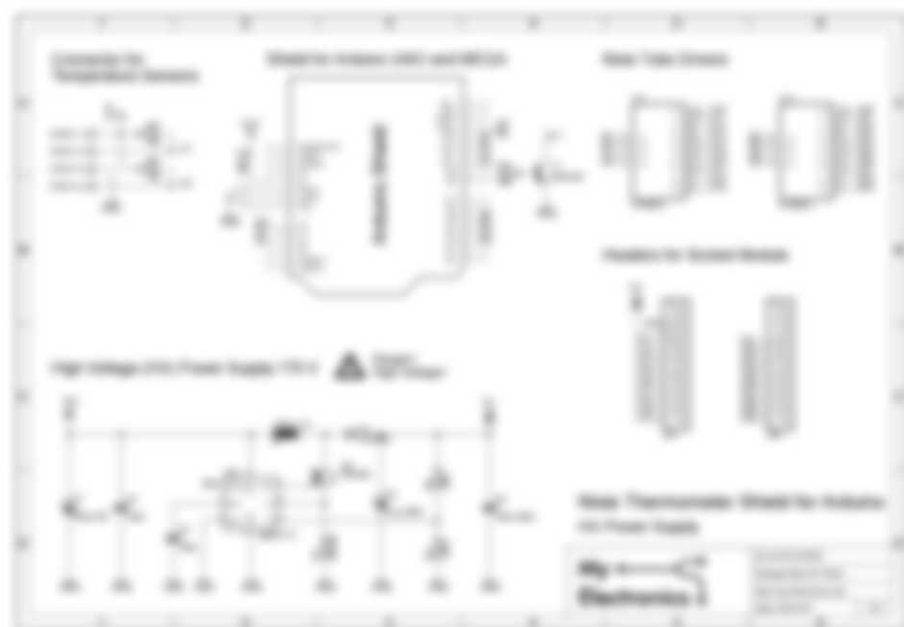
If the kit does not operate, please perform the following tests before sending a support request:

1. Check if all components are soldered properly and installed in the right position and direction
2. Check if the required external power supply is connected to your Arduino.
3. In the Arduino IDE, check if the sketch was uploaded properly.
4. In the Arduino IDE open Tools → Serial Monitor. Press the Reset button on your Arduino board and check the error messages and temperature readings.

Need help with the DIY Kit? Please send a support request with your order number or eBay ID: support@my-electronics.net

Schematics

48V Power Supply



48V LED



Functional Description

The circuit is based on a step-up switch mode power supply which generates the 170 V required to drive these tubes. There are a few ICs that can be used for step-up converters including the 9400000, 94001001 and others. Here, the 94001001 is used since because of its high efficiency. The IC is a step-up switching controller that uses a unique pulse width modulation (PWM) scheme to get a high efficiency in a variety of configurations. The employed circuit is mainly adapted from the [94001001 Datasheet](#). However, for a configuration like this, where the step-up is large, the component choice and board layout becomes important.

The key points are:

- M1 should be selected for low $R_{DS(on)}$ and Q_g
- C1 should be selected necessary I_s is 500 mA and rated $V_{DC} = 170$ V
- L1 should be rated 1 A
- C2 should be low ESR and close to IC as it must provide a high current for L1 and M1 very quickly
- The trace between the M1 and C2T pin needs to be short to allow the gate of M1 to be charged very quickly, directly affecting the efficiency
- The connection to the FB pin should be as short as possible as it is very sensitive to EMI interference
- D1 must be capable of handling at least 1 A
- C3 must be rated > 170 V
- C4, the output capacitor, should be low ESR (< 2 Ω) and rated > 170 V

From the [94001001 Datasheet](#) we get

$$R_{DS(on)} = R_{DS(on)} \left(\frac{V_{DS}}{V_{GS} - V_{GS(th)}} \right)$$

requiring $R_{DS(on)} = 10$ to 500 mΩ. To get $V_{GS} = 170$ V the resistors are selected as $R2 = 10.2$ kΩ and $R3 = 1.2$ kΩ.

$$V_{GS} = 1.0 \times \left(1 + \frac{R3}{R2} \right) = 1.0 \times \left(1 + \frac{1.2 \text{ k}\Omega}{10.2 \text{ k}\Omega} \right) = 10.7 \text{ V}$$

The specifications for, e.g., an 6B10 tube are as follows:

Supply Voltage	170 V
Operating Voltage	160 V
Cathode Current	2.5 mA

Therefore, using Ohm's law, the anode resistor should be:

$$R = \frac{170 \text{ V} - 160 \text{ V}}{2.5 \text{ mA}} = 40 \text{ k}\Omega$$

See the 'List of Buzzer Modules' for comparison.

To show the flow value two 4-DIGIT flow drivers are used (D1, D2). The D1 is a binary to decimal decoder with built-in high-voltage transistors. It takes a 5 V high-level input on D, C, B, A and outputs the corresponding digit of the flow value.

The driver takes a binary input on the pins 10, 12, 11, 13 and 5, 4, 3, 2, connected to the D, C, B, A pins of the 4-DIGIT driver. So, e.g. 1, 1, 1, 0 on the pins 10, 12, 11, 13 will display '7' on the flow tube (D1). Here 1 stands for 'LOW' (0 V) and 0 for 'HIGH' (5 V). 1, 0, 1, 0 on the pins 5, 4, 3, 2 will display '7' on the flow tube (D2). The entire logic table is shown in below.

Logic Table for the Flow Driver

	D	C	B	A	
1	1	1	1	1	0
1	1	1	1	0	1
1	1	1	0	1	2
1	1	1	0	0	3
1	1	0	1	1	4
1	1	0	1	0	5
1	1	0	0	1	6
1	1	0	0	0	7
1	0	1	1	1	8
1	0	1	1	0	9
1	0	1	0	1	10
1	0	1	0	0	11
1	0	0	1	1	12
1	0	0	1	0	13
1	0	0	0	1	14
1	0	0	0	0	15

With the 40 pin, defined as 0V in the sample sketch, the 5V Power Supply can be turned On or Off.

The sample sketch for the Flow Thermometer Shield is available on our GitHub repository: [Download Sample Sketch](#)

The sample sketch is well commented but some functions will be described in detail in the following section.

The function `void setupFlowDriver()` assembles the logic table for the flow driver. The function `void writeFlowDriver()` takes the input pins of the flow driver and the digit to be displayed as input and writes the corresponding high levels.

The transistor T1, driving the minus sign tube, is connected to pin 8 defined as 0V. HIGH will turn the flow tube On, LOW will turn it Off.

The DS18B20 (1) is a 1-wire digital temperature sensor. It reports the temperature in degree Celsius with a 0 to 12-bit precision, from -55 to 125 (±0.5) °C. The data pin of the temperature sensor is connected to A1 via terminal 2 of the screw connector.

The function `getTemperatureCelsius()` returns the temperature from a DS18B20 in degree Celsius. First, it searches for the connected sensor and gets its address. Then it sends the correct command block and reads the scratch pad. To read the scratch pad, the command `0x00` is sent and 9 bytes of data are received. The first two bytes corresponds to the LSB and MSB of the temperature data represented as a 16-bit two's complement binary (See Fig. 8 and Page 6 in the [DS18B20 Datasheet](#)). The temperature in degree Celsius is then calculated as follows:

```
int sensor = DS18B20(1);  
float temperature = sensor.getTempC();
```

Suppose that `MSB = 0111 1111` and `LSB = 0110 0000`. They will be combined to the 16-bit binary `0111 1111 0110 0000` by the integer left shift operator `<<` and the integer right operator `>>` and converted to signed integer value `-100` using two's complement. Finally, the factor `0.0625` converts between the sensor's internal 12-bit values and the temperature in degree Celsius leading to a temperature of `-100/0.0625 = -16 °C`.

© 2020 My Electronics

<http://my-electronics.net>