AIRWidget – A Simple Arduino-based IRWidget

1. What is an IRWidget?

An IRWidget is a tool for capturing and displaying infrared remote control signals. It is used together with IRScope, a PC program that displays and analyses data from the IRWidget. It is also one of the capture devices supported by IrScrutinizer, a powerful program for capturing, generating, analyzing, importing, and exporting of infrared (IR) signals. More about these programs is given in section 4.

The IRWidget was originally designed by Kevin Timmerman in 2007, who also wrote the original version of IRScope that I currently maintain. Kevin's original design is fully described here:

IR Widget - Consumer infrared remote control capture and visualization (compendiumarcana.com)

together with links to all the details needed to build one. Building one, however, requires having the means to program a PIC chip. Ready-made IRWidgets to this design have been available for many years from Tommy Tyler, see this link:

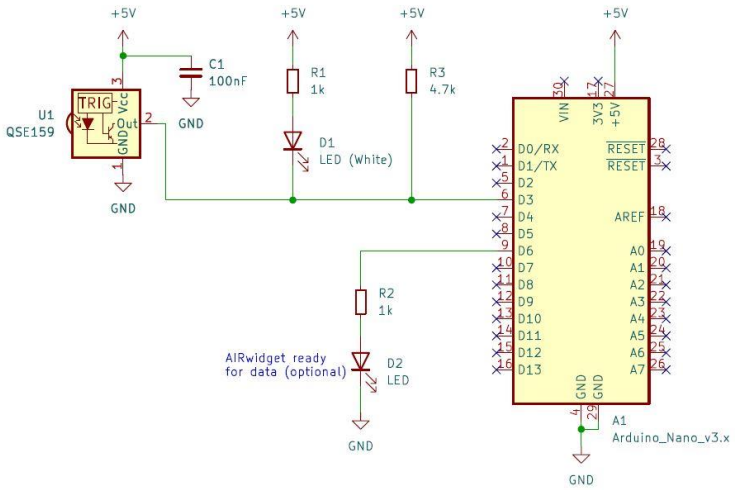
Tommy Tyler widget (weebly.com)

He currently accepts orders only from addresses within the continental United States, and is planning to retire completely from this activity by the end of this year.

2. What will replace the IRWidget?

To make access to IRWidgets more readily available, Tommy has led a project to design the AIRWidget, an Arduino-based IRWidget. This is a collaboration between Tommy, me, and Arduino forum member 6v6gt (Profile - 6v6gt - Arduino Forum). An Arduino-based solution has several advantages over the original PIC based design. It is written in C++, which is much more accessible to a wider audience than the assembler code of the PIC. Also, the Arduinos come with a pre-installed bootloader, so they can be directly programmed over USB and do not require a specialized chip programmer, like in Kevin's project.

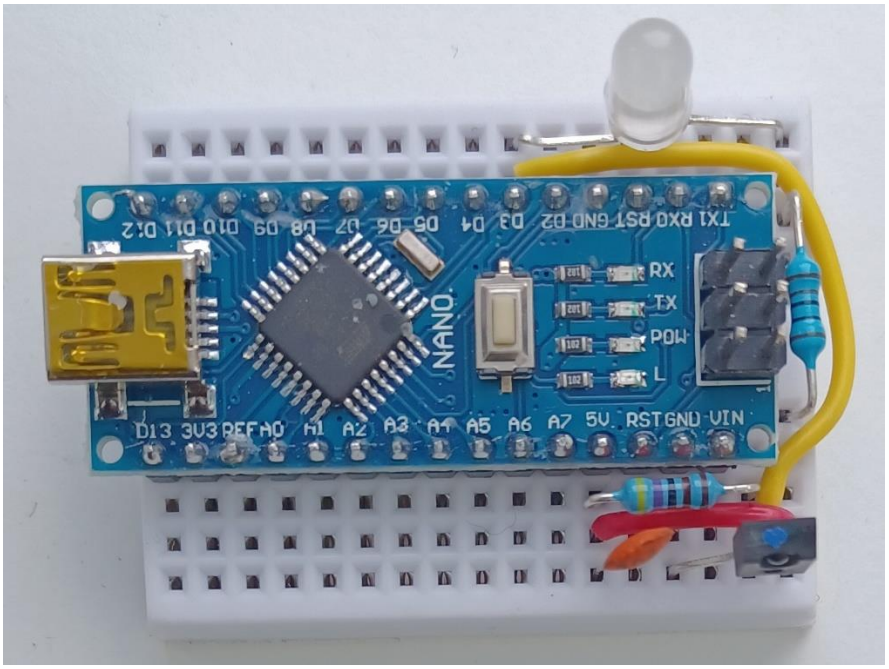
Here is the circuit diagram:



It uses few parts, all of which are readily available and can be assembled on a solderless breadboard, so avoiding the need for soldering. It can also be easily soldered in a more permanent form if desired. The

components are the Arduino Nano clone A1, the QSE159 photo sensor U1 to detect the IR signal, the LED D1 to tell you when you are receiving a signal, the 1K resistor R1 to limit the LED current, the 4.7K pull-up resistor R3 (which is in addition to internal pull-up provided by the Arduino) and the 100nF decoupling capacitor C1. The circuit diagram also shows an optional second LED D2 with limiting 1K resistor R2 that signals *AIRWidget ready* but in practice I have not found any problem with the readiness of the device. The decoupling capacitor C1 serves to limit electrical noise in the power supply and may be omitted if such noise is not a problem. Indeed, the Nano and QSE159 are the only absolutely essential components. All others are either to provide visual confirmation or to enhance reliability.

The photo sensor has three leads. Lying flat with the sensor window upwards and the leads towards you, the leads from left to right are ground, output and 5v. The LED has two leads but needs to be connected the right way round. Generally, on a new LED the leads are different lengths and the longer one is the positive lead. There is also a flat edge to the case of the LED next to the negative lead, which can distinguish them if the leads have been trimmed. Finally, it will do no harm to connect the LED the wrong way round, provided the resistor is in place, as it will not light and the connections simply need to be reversed. This picture shows mine, assembled on a small solderless breadboard. It includes the capacitor C1 but not the optional second LED D2 and its limiting resistor R2. The LED I used is white, but you can use any colour LED you prefer.



A genuine Arduino Nano uses an FTDI USB-to-Serial converter but many Nano clones, such as mine used in this picture, instead use a CH340 chip. There are also other Arduino boards besides the Nano. The majority of the testing of this design has been done with a Nano clone with a CH340 chip but it is expected that the design will work also with any Arduino board with an ATmega328P MCU chip and probably also with other USB-to-Serial converter chips. Note, however, that it will not work with the current official incarnation of the Nano, known as the Nano Every, as this uses a different MCU chip.

3. Installing the programs

When the device has been constructed, the next step is to install the Arduino IDE program in your PC and use it to upload the AIRWidget sketch into your Arduino Nano. You can get the Arduino IDE from this link:

Software | Arduino

At the time of writing, there is a version 2.0.0 of the Arduino IDE that is still under development so the latest release is the legacy version 1.8.19. I recommend using this legacy version and the instructions below are for this version.

In addition to this IDE you will need a driver for the USB-to-Serial converter of your Arduino board. If that is an FTDI chip then installing the IDE should also have installed the driver. Drivers for other converters may also install automatically, but if not then you will need to obtain the driver from the manufacturer's website, which for the CH340 is here:

CH341SER.ZIP - NanjingQinhengMicroelectronics (wch-ic.com)

You also need the AIRWidget sketch (Arduino's name for a program). This is the file AIRWidget.ino which is packaged with this document. For the IDE to upload the sketch, you need to create a folder called AIRWidget (the name of the sketch without the extension) and copy the sketch file into it. Then connect your Arduino to the PC. Note that Arduino Nano clones usually use a mini-USB connector rather than the more common micro-USB one. Open the Arduino IDE and make, or at least check, three settings under the Tools menu. Under Board select Arduino AVR boards > Arduino Nano. Under Processor there are three choices, "ATmega328P", "ATmega328P (Old Bootloader)" and "ATmega168". It should be one of the first two. Try one and if the software does not upload (as described below), try the other. I needed the Old Bootloader one despite this apparently being for Nanos from 2017 and earlier. Finally select the Port. Unless you have other USB ports connected, there should only be one choice listed.

Now go to File > Open and navigate to the downloaded sketch in its new folder, select it and press Open. The status bar should say *Compiling sketch*, then *Uploading*, and finally *Done uploading*. You can now disconnect the board and close the Arduino IDE, which you do not need again. The AIRWidget is now complete.

4. Using the AIRWidget

As mentioned in section 1, Kevin's IRWidget was designed to be used with his IRScope program. That program has been considerably enhanced since then by me and until this current project, the latest version was v2.01a. A comprehensive guide to the use of the Widget with this version can be downloaded here:

IRScope and Widget Guide (hifi-remote.com)

Due to slight differences in operation between Kevin's design and the AIRWidget, I have updated IRScope to v3.05, adding a new mode of operation that is specifically intended for the AIRWidget. This version can be downloaded here:

IRScope v3.05 (hifi-remote.com)

The guide applies almost unchanged to this version, but please note the following:

- Ignore the section on IR Widget Driver Installation as you will have already installed the driver in order to load the sketch into the Arduino.
- The guide tells you to select *IR Widget Count* in the drop-down *Hardware and Mode* box. Instead, select *Arduino Widget* which is the new entry and is the last in the drop-down list.

The remainder of the guide applies unchanged.

The original IRWidget is also supported by IrScrutinizer, a powerful program written by Bengt Martensson. This is available here:

GitHub - bengtmartensson/IrScrutinizer: IrScrutinizer is a program for IR signal analysis, decoding, generation and much more.

Select *tags* in the header line of the page that displays. The available versions are listed there together with download links. Just as IRScope has been updated to take account of differences in operation between Kevin's design and the AIRWidget, so also the support for the *IrWidget* device in IrScrutinizer is being updated. The AIRWidget will be supported by v2.4.0 and later. At the time of writing, this version is not yet available but is expected to be released shortly. There is just one issue to note if adjusting the parameters of IrScrutinizer from their default values. The value set for the *Options > Timeouts > Ending silence* parameter should be less than 500ms, required because the AIRWidget has a 500ms timeout itself. This constraint does not limit the capabilities of IrScrutinizer because the default is 300ms and there is no known IR protocol that includes a period of silence within its signals that is longer than 400ms.

Graham Dixon
4 November 2022