FAQ about the Transistor Tester (aka Component Tester)

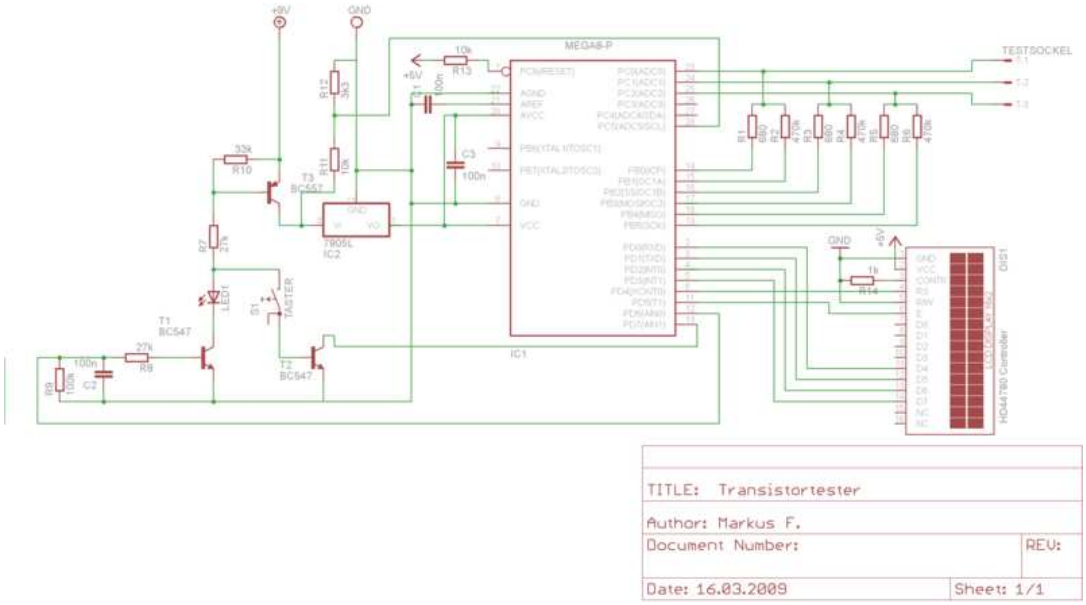
V0.2.4 by Feliciano

Foreword

After reading most of related the EEVblog thread over the years, I noticed there are several questions that pops-up every now and then, so I took the initiative of writing this FAQ with the help of the major supporters. We hope it help you get your bearings about what is this project about and what value can add to your bench.

FAQs

1. What is the "LCR and transistor tester"?
   a. Actually it's an electronics components tester. It started as an idea on the German Mikrocontroller forum. Here you can read a summary: https://www.mikrocontroller.net/articles/AVR_Transistortester#Introduction_.28English.29
   b. The basic design revolves around mimimun AVR hardware (ATmega) with a text display, for which a firmware was developed to allow identify an unknown electronic component of two or three pins connected to it, and its main characteristics.



   The key factors was to keep the price low and provide a reasonable precision (although not the same as a professional equipment costing 10x more)
   c. The original circuit could be optionally upgraded and/or expanded with several hardware options to add more features
   d. This entire project is OSHW (Open Source Hardware). The original developer, Markus Frejek stated his design was "Open Source". The time passed and he ceased releasing updates, so Karl-Heinz Kübbeler (kubi48) forked the project to add new features, and shortly Markus Reschke (madires) joined in with additional ideas and strong support. kubi48 releases the k-firmware; madires releases the m-firmware under EUPL V.1.1 License
   e. Some Asian vendors took the original circuit and/or improved/modern circuits, as well as the original/updated firmware, and started to sell "original" mods of the OSHW circuit and firmware. We refer to those unassembled or finished kits as "clones", some are even clones of other clones, and the quality varies from one vendor/batch to the other. Normally those vendors lock the chips and the designs "to avoid piracy" and don't show interest on sharing their hardware/firmware mods for further improvements. Therefore most of the times somebody buys a clone online, cannot find an update for the firmware it came with, so you can keep it as it came (as long as it's not broken there's no need to fix it, right?), or you can try to flash the OSHW firmware and gain more features and/or improvements

f. Please note the following open letter from the microcontroller.net forum to the component tester clone vendors: "*Dear Transistortester Cloners and Sellers!*

We don't mind if you produce and sell clones of the Transistortester. It provides an inexpensive great little tool for electronics enthusiasts and beginners, but PLEASE note the links to the project's webpage, source repo and documentation. You would add more value by giving users that information to be able to update the firmware and to understand all the features. If you do any modifications to the firmware, please send us a copy for the repo. And if you would send us your Transistortester clones, we would be able to keep the firmware as compatible as possible. Don't forget, this is an OSHW project!"

2. Is this really an LCR meter?
   a. The typical working principle of commercial LCR meters is to generate an AC signal (usually around 100KHz) and from the measurements calculate many of the parameters of interest for the selected connected inductor or capacitor under test
   b. Additionally, the commercial LCRs use to have a wide range of measurement, and medium-to-high accuracy
   c. On the other hand, the inexpensive component testers here explained works under a totally different principle: it applies to the device under test (DUT) a combination of DC pulses, while sampling the voltages on its terminals at given times. This way it can deduct the main parameters of resistors, diodes, several kinds of transistors, SCR, Triacs, and also for certain range of capacitors and/or inductors. Therefore, although the clone makers use the LCR buzzword, the component testers here discussed are not the comparable to an USD100 to USD200 commercial LC/LCR meter
3. What is the working principle of the transistor/component tester?
   a. As mentioned previously, it has an AVR microcontroller (there are different designs with different amounts of flash and RAM) with bidirectional I/O ports. Some of these ports are connected to known high value resistors (470K), and others are connected to known low value resistors (680Ω). By sending pulses over one pin and sampling the voltage on the same and/or other pins, the kind of component connected can be inferred, and from it, its main characteristics can be determined (under 5V and low current scenarios, which is what a microcontroller with some resistors can offer)
   b. For more technical details, Karl-Heinz has shared very detailed information on how his approach works. You can read the Chapter 5 of: https://github.com/kubi48/TransistorTester-documentation/blob/main/pdftex/english/ttester.pdf
   c. There is also a technical manual for m-firmware (with the help of Bohu): https://github.com/madires/Transistortester-Warehouse/blob/master/Documentation/English/ctester-1.44m.pdf
4. How a transistor/component tester is used?
   a. Easy: connect the battery or power adapter (preferable a linear regulated supply), and for testing a component, connect its two or three terminals, press the test button, wait, read the results. Depending on the tester you have, and the firmware installed, you may have additional functions such as signal generator, frequency measurement, etc.
   b. Warning: If you're about to measure a capacitor (especially electrolytic), make sure it's discharged. A high voltage/current can damage the protection of your equipment (if exist), and/or the MCU
   c. Calibration or Self-test note: this equipment needs to be adjusted every now and then, and/or when you flash a new firmware. After invoking the calibration or self-test, the screen will show some instructions to follow along the procedure. The details will depend on the particular hardware/firmware&version your unit is running, you can check it on the above referred user manuals if you want, and it's something like:
   d. For the k-firmware the calibration can be started by connecting the three terminals together (i.e. make a short), and press the test button (for the units with menu, it can be selected from the menu entry as well), acknowledge the confirmation prompt, wait, isolate probes, connect a film capacitor >100nF (followed by a film capacitor between 10nF and 30nF if the SamplingADC is enabled), remove it, wait for the results
   e. For the m-firmware, it's recommended to measure at least 3 times film capacitor(s) between 100nF and 2.2uF before performing the self-test. After that, the self-adjustment can be called from the main menu, you just have to follow simple on-screen instructions

f. The measurements performed during the calibration or self-tests procedure are stored in the EEPROM. These offset parameters can be shown after the self-test calibration, or from selected the menu entry for showing data. Sometimes it's needed to review these offset values to troubleshoot some odd measurements. Also, some firmware allows to save two profiles, so you can use a ZIF socket and a set of leads, for instance

5. Can this be used for (ESR) measurements on-circuit?
   a. If you measure, let's say a capacitor (remember to discharge it first), you're measuring the component and whatever other components are connected to it, so the measurements can be deceiving. Therefore, you can do the in-circuit measurement, but if the results show something weird, you are advised to remove the component and measure it again off-circuit to validate the results
6. What information is shown on screen of the transistor tester?
   a. It depends whether your unit is running the k-firmware, the m-firmware, or one of the Asian variations (MTester and such), and the version of that firmware
   b. In the case of the k-firmware, the author wrote a visual guide of its functionalities: https://github.com/kubi48/TransistorTester-documentation/blob/main/pdftex/english/ttinfo.pdf
7. Which clone I have?
   a. Kubi48 kindly gathered some pictures of the main clones, so you can look whether the one you already have look similar to one of those: https://github.com/kubi48/TransistorTester-source/blob/master/picture-link.pdf
8. Which clone do you recommend?
   a. It depends. An ATmega processor can have flash of 8KB, 16KB, 32KB, 64KB, etc. and a RAM of 512B, 1KB, 2KB, 4KB, etc. and it can have more or less ports (some of them will be used for the test terminals and the screen connections), so there is no perfect design yet unless you build one yourself from scratch and compile the firmware accordingly
   b. On the dedicated EEVblog forum thread: https://www.eevblog.com/forum/testgear/$20-lcr-esr-transistor-checker-project/ the most popular are the AY-AT (it has that name printed on the PCB) and the Hiland m644. Indman kindly generated a table of the main clones and its main features, so you can choose the one that better match your needs: https://github.com/madires/Transistortester-Warehouse/blob/master/Documentation/English/Clone-Comparison-Chart.pdf
9. Can I improve my equipment?
   a. Probably yes, either by using lower tolerance resistors (i.e. 0.1%), and/or upgrading the crystal from 8MHz to 16MHz, and/or upgrading the microcontroller for one with more memory, and/or by improving the power supply and/or voltage reference to higher precision (i.e. replacing the typical TL431 with an LM4040 or such, or removing it if your 5V regulator is a high precision one like MCP1702). This has been discussed several times on the EEVblog thread. You can search for details
   b. You can also add optional components, like a rotary encoder, or a circuit for voltage measurement, or a buffer circuit for external signals, or a TTL I/O, etc.
   c. Note: Most of the hardware changes require compiling/flashing a custom firmware
10. Can I repair my equipment?
    a. If you forgot to discharge a capacitor before measurement, you may have burnt the protection circuit (in case your circuit had one), or you may have blown your AVR. The protection elements (diodes, if present) can be removed and you can check whether the rest still works, or if you can easily find the parts and prefer to repair rather than to replace the whole unit, you can swap the microcontroller and re-program it with the k-firmware or m-firmware
    b. Note: some clones have better designs than others, and some vendors use better components than others for the same clone, and sometimes one batch of components has better quality than other. YMMV
11. Which is the latest firmware?
    a. If you have an Asian clone, most likely you have an old firmware, under the original version number or an unofficial version number (such as 2.12k for a TC-1 clone). Probably you won't find an update for that
    b. The current k-firmware version (by 2021/Q3) is 1.13k
    c. The current m-firmware version (by 2021/Q3) is 1.44m

d. Note: the two OSHW firmware branches are different (different approach, features, GUI), so is not like one is better and/or newer than the other. On most cases you can try both and decide which one works better for your liking
e. If you want to read about the differences between k-firmware and m-firmware, madires releases a README of over 50 pages inside his m-firmware archive, where some of the pages discuss the differences with the k-firmware

12. Where can I download the firmware?
    a. For Asian firmware: we don't know. Those firmware use to have nicer graphics, but in exchange of functionalities (memory trade-off). Some users have managed to read the firmware of certain clones and share it on the EEVblog forum. Indman has collected several of those, but I advice to better to ask him before using any of those as he had made modifications of some
    b. For k-firmware (or trunk): https://github.com/kubi48/TransistorTester-source/tree/master/trunk
    c. For m-firmware: https://github.com/madires/Transistortester-Warehouse/tree/master/Firmware/m-firmware
13. Is the firmware only available in English?
    a. The info on the display can be shown in several of the major languages, you just need to check the list, compile the firmware selecting on the makefile the one you want, flash, enjoy
14. How can I compile the firmware?
    a. For k-firmware: KH has created folders for the main clones with the related EEPROM and flash files (.eep and .hex). You can either directly flash those to try, or you can git the whole source, edit the related makefile according to your clone variation and/or likings, compile (make) it, and flash it
    b. For m-firmware: madires release the source (not targets) as a .tar.gz file. You have to download and decompress it, open the "clones" file and find the settings to modify for the clone you have, edit several .h directive files (do a backup of those just in case), edit the makefile (do a backup too), and compile (make) it
    c. On your PC you can do this on Windows or Linux. Remember this is written in C, so you have to install the compiling environment for your OS. In case of Windows, you can try WinAVR with the right version of the ATmega toolchain (overwriting WinAVR), you may have to overwrite the msys-0.1.dll too if you have W8, W8.1 or W10. And for Windows you will also have to install the drivers for the cable you're about to use to connect to the board. Popular cables are TL866A and USBasp (in case of the later, please disregard posts on the net telling you need to upgrade the firmware to a 2010 version, unless your cable is that old). There are other options like Arduino, etc.
    d. For more information, Flywheelz made a detailed description with the links to the Windows software: https://www.eevblog.com/forum/testgear/$20-lcr-esr-transistor-checker-project/2475/
    e. Indman also shared an archive with the Windows software: https://drive.google.com/file/d/1ioYbEuzrPkiSpJ51IamuwcZ1IU4PplbI/view
    f. And Blurpy has made a Windows guide for a popular clone he has: https://github.com/blurpy/transistor-tester
    g. If you are using linux or Mac, the k-firmware and the m-firmware has instructions (in the readme and/or pdf) on how to install the tools and compile and flash the firmware
15. How can I flash the firmware?
    a. You can do it on-circuit (if you solder an ISP or similar port) or off-circuit (if you have a socketed microcontroller). Normally the clones come without a populated ISP port, so if the port is there you just have to solder some headers for your cable, but sometimes you have to solder wires from certain pins of your AVR in order to add the ISP port
    b. After finding or compiling (make) the firmware files (.eep and .hex), having the ISP port ready and connecting the cable, you can use avrdude or equivalent (like a GUI frontend called AVRDUDESS) for flashing both files. Please remember to read about the fuses in advance and set it correctly, especially if your unit came locked.
    Warning: a wrong set of fuses can render the ISP port useless, and therefore you would need a high voltage programmer to recover your AVR

c. So, if you're only about to flash existing k-firmware's .eep and .hex files you just need to run avrdude to flash both files, and probably set the fuses (especially if your unit is locked, in which case you also have to erase it before flashing it)
   Note: Windows 10 doesn't handle correctly too long command lines, so if you're about to use avrdude on the command prompt, you may have to program the EEPROM, the flash, and set the fuses as 3 separated steps

16. Where can I find more information?
    a. You can search the whole 280+ pages of the EEVblog thread by using the print preview button at the top and searching for the keywords you're looking for. You can also use the google site search
    b. Or you can read the last 10 or so pages (as many people ask similar questions often), and if you don't find what you're after you can ask, if you ask nicely, for sure some kind person will give you a hand

Closing Thoughts

After several years this projects keeps moving forward, thanks to the developers, and the community of the EEVblog and German and Russian forums. This FAQ is not finished, yet it can help the project a little, and can be improved too. If you have comments, please contact Feliciano or indman or madires at the EEVblog.