

# Arduino UNO ComponentTester

This device is originally designed by Markus Frejek, [link](#) and software by Karl-Heinz Kuebbeler (kh\_kuebbeler@web.de). The Ardutester software is based on porting by Markus Reschke (madires@theca-tabellaria.de) and later improving by PighiXXX (info@pighixxx.com) and PaoloP.

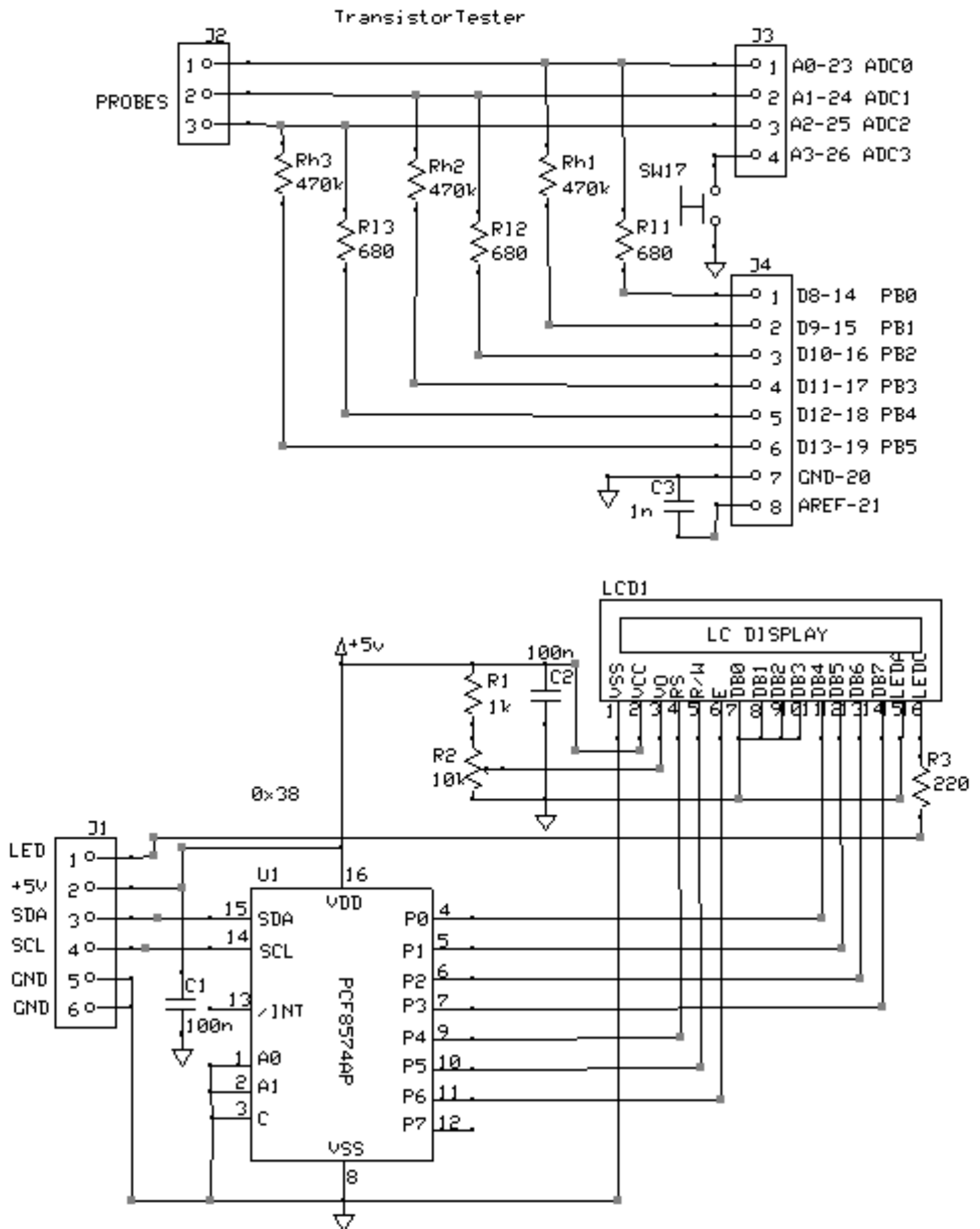
My improvements for Ardutester software are:

- displaying results into different display types for example DOGM128 display with SPI interface,
- restructuring software into header and class modules;
- some changes (for example PWM tool when SPI interface shares timer1 pin, frequency measuring function) and bug fixes;

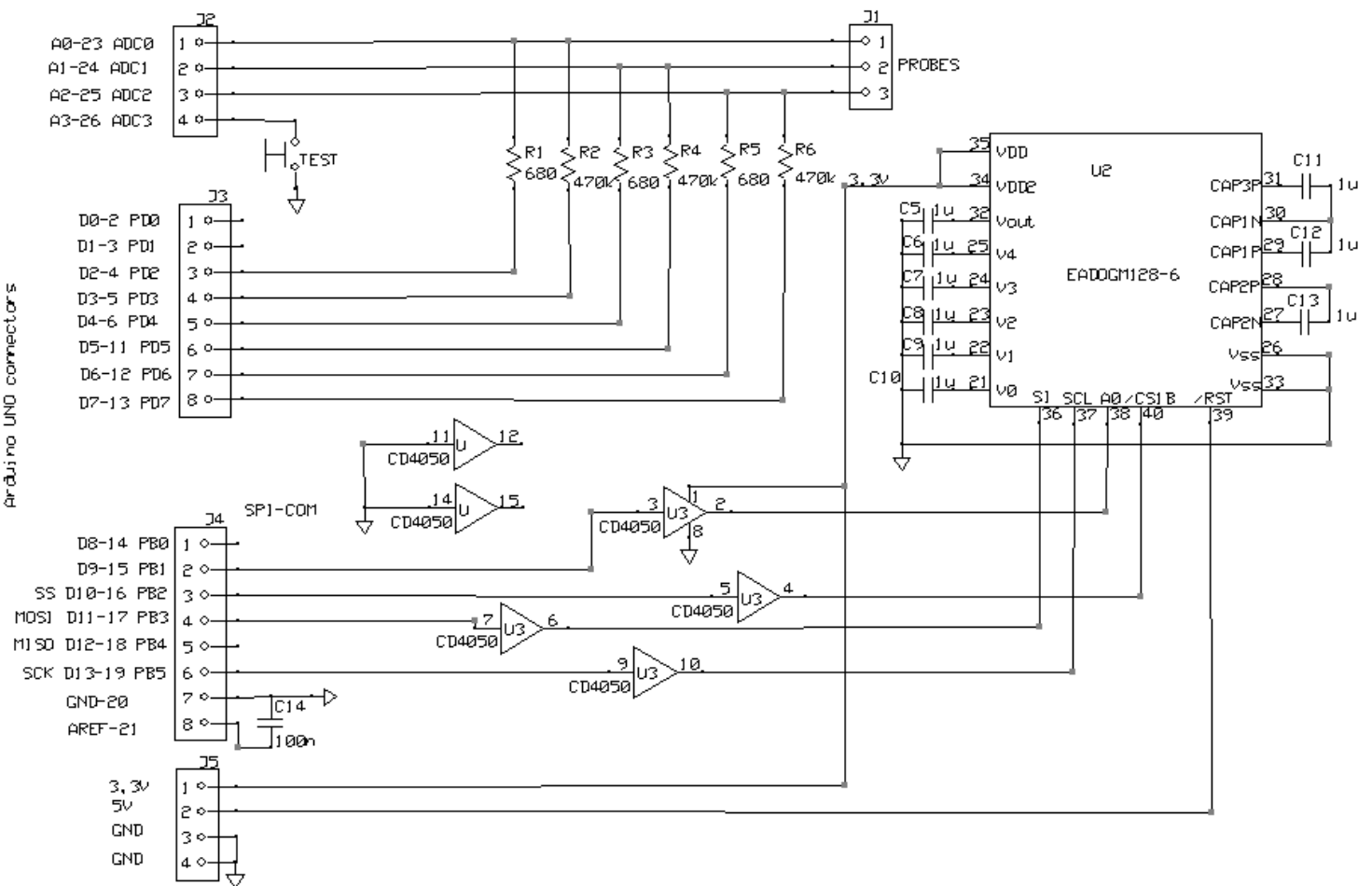
Electronic Component Tester is based on Arduino UNO board.

Electronic schematic diagram with I2C LCD Display.





## Arduino UNO connectors



[illegible]

The diagram illustrates the internal wiring of an Arduino Uno-based system. Key components and their connections include:

- Arduino Uno Connectors:**
  - J2 (ADCs):** A0-23 to ADC0, A1-24 to ADC1, A2-25 to ADC2, A3-26 to ADC3.
  - J3 (PDs):** D0-2 to PD0, D1-3 to PD1, D2-4 to PD2, D3-5 to PD3, D4-6 to PD4, D5-11 to PD5, D6-12 to PD6, D7-13 to PD7.
  - J4 (PBs):** SP1-COM to PB0, D8-14 to PB0, D9-15 to PB1, SS to PB2, D10-16 to PB2, MOSI to PB3, D11-17 to PB3, MISO to PB4, D12-18 to PB4, SCK to PB5, D13-19 to PB5, GND-20 to GND, AREF-21 to GND.
  - J5 (Power):** 3.3V, 5V, GND, GND.
  - J6 (Freq):** Freq, GND.
- Power Regulation:**
  - A 3.3V regulator (U1) and a 5V regulator (U2) are shown, both using LM78xx series regulators.
  - Resistors R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15, R16, R17, R18, R19, R20, R21, R22, R23, R24, R25, R26, R27, R28, R29, R30, R31, R32, R33, R34, R35, R36, R37, R38, R39, R40, R41, R42, R43, R44, R45, R46, R47, R48, R49, R50, R51, R52, R53, R54, R55, R56, R57, R58, R59, R60, R61, R62, R63, R64, R65, R66, R67, R68, R69, R70, R71, R72, R73, R74, R75, R76, R77, R78, R79, R80, R81, R82, R83, R84, R85, R86, R87, R88, R89, R90, R91, R92, R93, R94, R95, R96, R97, R98, R99, R100 are used for voltage division and current limiting.
  - Capacitors C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19, C20, C21, C22, C23, C24, C25, C26, C27, C28, C29, C30, C31, C32, C33, C34, C35, C36, C37, C38, C39, C40, C41, C42, C43, C44, C45, C46, C47, C48, C49, C50, C51, C52, C53, C54, C55, C56, C57, C58, C59, C60, C61, C62, C63, C64, C65, C66, C67, C68, C69, C70, C71, C72, C73, C74, C75, C76, C77, C78, C79, C80, C81, C82, C83, C84, C85, C86, C87, C88, C89, C90, C91, C92, C93, C94, C95, C96, C97, C98, C99, C100 are used for decoupling and timing.
- Display and Control:**
  - The **EAD0GM128-6** display is connected to the Arduino's data pins (D0-D7) and control pins (CS, SCL, SDA, RST).
  - The **CD4066** (U4, U5, U6) multiplexers are used to route signals between the display and the inverters.
  - The **CD4050** (U3) inverters are used to drive the LED backlight.
- LED Backlight:**
  - The LED backlight is connected to the output of the inverters (U3) and the 5V regulator.
  - Resistors R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15, R16, R17, R18, R19, R20, R21, R22, R23, R24, R25, R26, R27, R28, R29, R30, R31, R32, R33, R34, R35, R36, R37, R38, R39, R40, R41, R42, R43, R44, R45, R46, R47, R48, R49, R50, R51, R52, R53, R54, R55, R56, R57, R58, R59, R60, R61, R62, R63, R64, R65, R66, R67, R68, R69, R70, R71, R72, R73, R74, R75, R76, R77, R78, R79, R80, R81, R82, R83, R84, R85, R86, R87, R88, R89, R90, R91, R92, R93, R94, R95, R96, R97, R98, R99, R100 are used for current limiting.

### Prototype with DOGM128 LCD Display - PWM signal configuring and start:

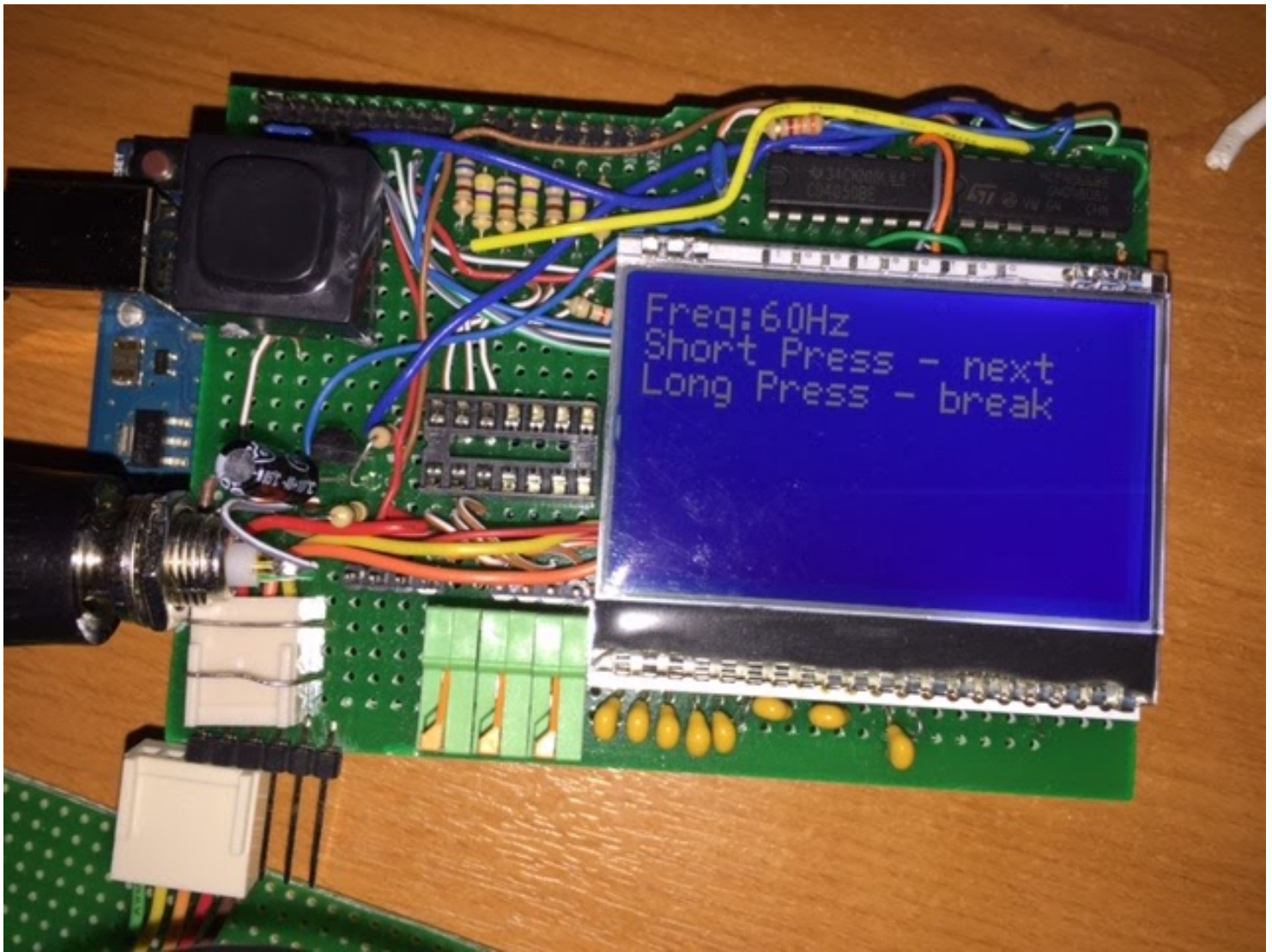


Example of testing N channel MOSFET (CD4066 not yet included):



Frequency measurement additions:





Firmware project is implemented with Arduino IDE 1.6.6 and uploaded to [here](#).

Project uses NewLiquidCrystal and U8GLIB libraries. Both libraries class files are modified little bit.

Modification to NewLiquidCrystal library header file LiquidCrystal\_I2C.h, added constructor without parameters:

```
LiquidCrystal_I2C ();
```

Second modification into LiquidCrystal\_I2C.cpp file is this constructor implementation:

```
LiquidCrystal_I2C::LiquidCrystal_I2C()  
  
{  
  
config(0x38, EN, RW, RS, D4, D5, D6, D7);  
  
}
```



Modification to U8Glib.h file was done into class U8GLIB\_DOGM128 definition - added constructor without parameters:

```
U8GLIB_DOGM128() : U8GLIB(&u8g_dev_st7565_dogm128_hw_spi, 13, 11, 10, 9, 255){}
```

### **Small user manual**

After starting device it is in component testing mode which is default mode.

Connect component with probes and short press to button.

If component is connected to probes then device tests it and displays results and graphic image with pin numbers if component was detected. If component was unknown or not connected to probes then message: "No component found!" will be displayed.

Long press to button will bring device into menu mode. Short press will display next menu choice. Long press will choose it.

When you choose PWM, device will offer frequencies starting at 100Hz. Other choices are 250, 500, 1000, 2500, 5000, 10000, 25000 Hz. Long press will choose frequency. Now you can choose duty cycle. Also Info will be displayed: Short press will increase duty cycle 5%, long press will decrease 5%. Double-click into button will start PWM. Pressing to button again will finish PWM.

When you choose Frequency Measurement, device starts measuring and displays to screen cycle count. When it finishes, it displays frequency measured at input connector. Short press to button will repeat measure cycle, Long press to button will terminate this task and device goes into tester mode. If frequency input connector is not connected to signal then device never finishes and you can break measuring with short button press.