UnoLib documentation

How to start

version 11/10/2025

UnoLib is an open-source Pascal library designed for the Arduino Uno platform (ATmega328p microcontroller). It is a translation of a subset of the standard Arduino library, adapted as needed for a Free Pascal environment. Setting up the Free Pascal Compiler (FPC) for AVR microcontrollers, which is required for UnoLib, there are two primary options:

- Install the AVRPascal IDE with build-in latest UnoLib release,
- Install and configure the FPC AVR cross-compiler using tools like FPCUPdeluxe.

1. Setup using AVRPascal IDE

The AVRPascal IDE is the recommended method for new users. It bundles the Free Pascal Compiler (FPC) for AVR, the necessary Binutils, and the avrdude upload tool, configuring the system automatically. In this case download and run the AVRPascal installer, available for Windows, Linux and MacOS systems. This all-in-one package automatically configures the necessary FPC cross-compiler, Binutils, and the avrdude uploader. To start programming with the AVRPascal IDE:

- Open a new project or example code
- Select your board (e.g., Arduino Uno).
- Click Compile and Upload.

AVRPascal IDE is available at the author’s webpage:

https://akarwowski.pl/index.php?page=electronics&lang=en

2. Setup Free Pascal Compiler (FPC) for AVR and Lazarus

This method is recommended for advanced users, and requires detailed manual configuration of the compiler, Binutils paths, and the Lazarus IDE settings, offering full control of installation process. For a comprehensive, step-by-step guide on both installation methods, including how to build the cross-compiler with FPCUPdeluxe, please refer to the dedicated documentation:

https://wiki.freepascal.org/AVR_Embedded_Tutorial_-_Entry_Lazarus_and_Arduino

3. Library modules and examples

The library contains most of the core of the standard Arduino library translated into Pascal, as well as additional modules not present in RTL of Free Pascal for AVR. Library modules included:

- analog.pas - support for analog pins
- defs.pas - definitions of constants, bit manipulations, port support
- dht.pas - support for DHT11/22 sensors
- digital.pas - support for digital pins
- ds1302rtc.pas - support for DS1302 real time clock
- fix16.pas - support for fixed point numbers
- float32.pas - support for floating point numbers, basic arithmetic and trigonometric operations (in collaboration with @Dzandaa)
- hardwareserial.pas - support for serial communication
- i2c.pas - support for I2C communication bus (by @Dzandaa, many thanks to @ccrause)
- liquidcrystal.pas - support for LCD
- pulse.pas - routines for reading a pulse on a pin (additional function written by @Dzandaa returns length of the pulse in milliseconds)
- stringutils.pas - string conversion routines (by @Dzandaa)
- timer.pas - time-related routines
- tone.pas - square wave tone routines

The library modules are located in the "lib" directory.

Documentation is placed in "docs" directory:

- analog.pdf - documentation for analog.pas unit
- defs.pdf - documentation for defs.pas unit
- dht.pdf - documentation for dht.pas unit
- digital.pdf - documentation for digital.pas unit
- ds1302rtc.pdf - documentation for ds1302rtc.pas unit
- fix16.pdf - documentation for fix16.pas unit
- float32.pdf - documentation for float32.pas unit
- hardwareserial.pdf - documentation for hardwareserial.pas unit
- liquidcrystal.pdf - documentation for liquidcrystal.pas unit
- pulse.pdf - documentation for pulse.pas unit
- stringutils.pdf - documentation for stringutils.pas unit
- timer.pdf - documentation for timer.pas unit
- tone.pdf - documentation for tone.pas unit

The "examples" directory contains simple sample programs using UnoLib modules:

- DS137ZN_RTC_Test.pas - Real Time Clock test using I2C (by @Dzandaa)
- HMC5883L_Magnetometer_Test.pas - HMC5883L Magnetometer test using I2C (by @Dzandaa)

- I2CScan.pas - I2C bus scan (by @Dzandaa)
- pcf8591t_ACDC_Read.pas - AC/DC read test (by @Dzandaa)
- pcf8591t_ACDC_Write.pas - AC/DC write test (by @Dzandaa)
- TestBlink.pas - turns on and off the built-in LED
- TestBlinkWithoutDelay.pas - turns on and off the built-in LED using Millis
- TestDHT11.pas - displays temperature and humidity from a DHT11 sensor on an LCD display
- TestDigital.pas - turns on and off the built-in LED based on the button state
- TestHC-SR04.pas - example of using HC-SR04 ultrasonic sensor
- TestLCAutoscroll.pas - scrolls text on the LCD display
- TestLCBlink.pas - displays the text "hello, world!" on the LCD display
- TestLCChars.pas - displays non-standard characters on the LCD display
- TestLCCursor.pas - turns the cursor on and off on the LCD display
- TestLCDisplay.pas - displays and turns off the text "hello, world!"
- TestLCSerialDisplay.pas - displays characters taken from the serial port on the LCD display
- TestLCTextDirection.pas - changes the direction of text display on the LCD display
- TestLM35.pas - displays the temperature value from the LM35 sensor on the LCD display
- TestSerial.pas - sends and receives data via the serial port
- TestTone.pas - plays a melody using _tone routine

Some extra code is placed in "extras" directory:

- ccrause_blink_leonardo.pas - blink example for Arduino Leonardo based on code written by @ccrause
- ccrause_delay.pas - delay module based on code written by @ccrause
- ccrause_test_leonardo.pas - serial communication example for Arduino Leonardo based on code written by @ccrause
- cdc.pas - CDC support for Arduino Leonardo
- debug_leonardo.pas - simple LED-based debug code
- usb.pas - USB support for Arduino Leonardo
- wdt.pas - watchdog timer routines

Lazarus test projects are placed in "tests" directory:

- conversion - routines for numeric and string conversion and data preparation for CORDIC algorithm (by @Dzandaa with collaboration of @ackarwow)
- float32 - test programs for testing TFloat32 type both on Lazarus and on Arduino side (based on serial port communication program by @Dzandaa)
- serial - serial port communication test (by @Dzandaa)

Before compiling a given program, make sure that the library modules listed in the uses section have already been compiled.