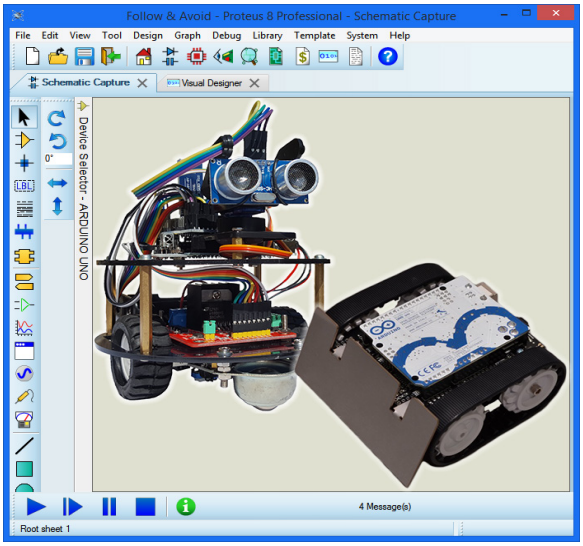
Arduino™ Turtle Simulation

Free with Visual Designer!!

- **Write turtle programs as a Flowchart or with Arduino C++**
- **Create Virtual Maps, Mazes and Playgrounds for the turtle in seconds**
- **Simulate the Turtle executing the program in the virtual Map or Maze**
- **Program the real hardware turtle at the press of a button**

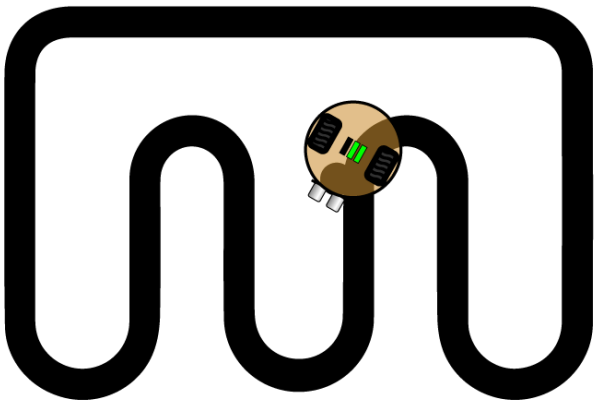
What is it?



Arduino Turtle Simulation in Proteus is a breakthrough tool for teaching and learning embedded control. It is supplied with the Proteus Visual Designer for Arduino AVR product, meaning that students can create turtle firmware either as a flowchart or as an Arduino C++ program. Visual Designer makes the task much simpler by providing high level methods for control such as forwards(int speed) and stop() for the motor drive and ping() for the range finder. This enables students to focus on the control of the turtle and watch something happening quickly.

During a simulation the turtle needs an environment in which to move while it is executing the students program. These can be created in seconds in a graphics package, such as MS Paint, with the simple rules that black lines are to be followed and anything drawn in red is to be avoided. Students can therefore be assigned the exact same challenge and they don't need real hardware or large amounts of space to test and perfect their solutions.

It is so fast and easy to change the 'landscape' that programs can then be tested against progressively more extreme environments (e.g. right angles, figure of eight, dead ends) with a few clicks of the mouse.



What can you do with it?

You can design programs to control a robot turtle at a high level and then simulate, test and debug your program against any number of virtual landscapes. The landscapes are designed to test line following, obstacle avoidance, maze solving, multi-colour logic, or collect-and-deposit challenges.

A line following challenge is perhaps the most popular of all and these can range from the simple to the surprisingly tricky. The basic idea is that the turtle is placed on (or needs to hunt for) the line and then has to follow the line around the route. Some tasks now require following one coloured line (e.g., magenta) while ignoring others.

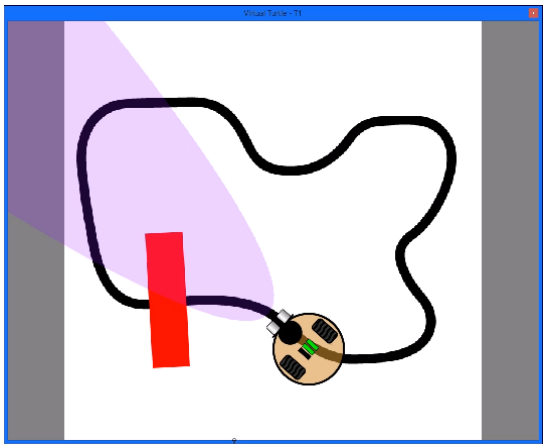
With an obstacle avoidance challenge, the robot turtle must move around obstacles that are in its way. This can also be used together with a line following challenge to force a change of direction around the circuit.

The Funduino turtle includes an ultrasonic range finder mounted on a stepper motor, making it ideal for obstacle avoidance challenges. Visual Designer includes high-level access functions such as ping() to help control the turtle, and the environment containing the obstacles can easily be created or changed in a graphics package.

A maze solver, or a maze escape, is probably the most difficult of the standard challenges for robot turtles. It can be drawn in many ways, including obstacles or simply using dead ends in an attempt to trap the turtle.

The controlling program needs a 'memory' of routes travelled as well as sufficient logic to turn and/or reverse on the line. The Zumo turtle with its gyro and compass has accurate orientation methods such as turnSouth() in Visual Designer to help with this, but it has no range finder and is best used with dead-end maze challenges. By contrast, the Funduino has a range finder to avoid obstacles but has no position encoding, so turning of the turtle is quite challenging.

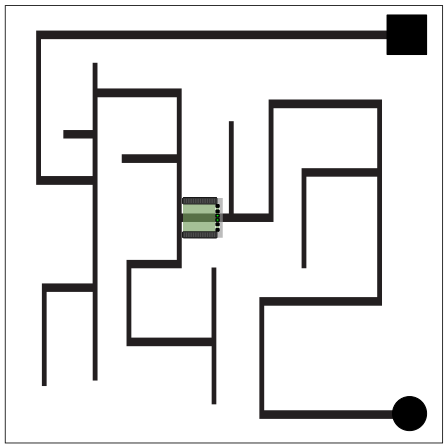
What's included?



Turtle Simulation is part of the Proteus Visual Designer for Arduino AVR product. In addition to support for the Funduino and Zumo hardware turtles, a fully virtual Proteus Turtle is provided for simulation-only challenges like multi-colour line following and collect-and-deposit. You also get generic AVR microcontroller models, simulation support for dozens of popular Arduino shields and Grove peripherals, and high-level programming using flowcharts.

The Proteus Turtle offers an ideal entry point for algorithm development and visual logic testing, with no need for physical hardware. It's especially useful for classroom or remote learning environments where access to real robots may be limited.

The Funduino robot includes an ultrasonic rangefinder on a rotatable head, making it well suited to obstacle avoidance challenges. It has three underside line sensors and a motor driver for both wheels and stepper head. However, it lacks position encoders, so turning and path accuracy are challenging in practice.



The Zumo is a more advanced robot that runs on tracks, driven by two brushed DC gearmotors. It includes six IR reflectance sensors for line following and edge detection, a piezo sounder, and a 3-axis gyroscope and compass for directional control. Visual Designer provides orientation commands like turnEast() to support this.

All three turtles are modelled in exceptional detail and run inside the Proteus simulator, which executes code at cycle-accurate instruction level. Full professional debugging tools are available so students can step through code and inspect state as needed. Once ready, the exact same code can be used to program compatible hardware directly from within Proteus.

Why use it?

- Programming for Arduino turtles is easier for students with Proteus. They can choose to code via flowchart or Arduino C++ and in either case are provided with a library of functions that help control the turtle.
- Proteus can simulate the turtle program in a virtual playground that you design. This enables early success for students who can each see their program in action on screen without needing to set up or share a cumbersome hardware setup.
- Educators can create and deploy simple graphics files (PNG) containing maps, obstacles or mazes to students. This means that each student can be assigned the exact same challenge and that work can be completed at home or in any classroom with computers.
- Class exercises can quickly be tailored to suit multiple abilities. Students can progress from flowchart to C++ programming and maps and mazes of increasing difficulty can easily be created and supplied to students.
- Programs can be fully debugged in the Proteus software. Not only can students set breakpoints and single step debug but they can see the turtle behaviour on screen as they step through their code. This promotes real understanding and helps teach valuable testing and debugging skills.
- Simulating the turtle in a virtual environment means that students can work independently, without need to share scarce hardware resources or space in the classroom. It also isolates from unwanted real-world effects; for example, a classroom full of robots using ultrasonic range sensors would result in a lot of confusion as sensors pick up random sonar bursts from other robots.
- Students can gain deeper understanding of the electronics involved by placing instrumentation (e.g. oscilloscope) on the schematic and taking measurements.
- Since Proteus simulates the exact same HEX file as the real Arduino, students can program the physical turtle at any time at the press of a button. This enables them to test whether their programs account for mechanical forces such as momentum and inertia as well as watching their algorithms come to life.

Labcenter Electronics Ltd. Head Office
The Stables, Hartlington Hall, Burnsall, BD23 6BY, United Kingdom.
Tel: (+44) 1756 753440, Fax: (+44) 1756 752857
Web: www.labcenter.com, Email: info@labcenter.com.