Simulator for Arduino

Table of contents

Introduction

Simulator for Arduino is an Arduino Simulator which may be downloaded or purchased from www.virtronics.com.au. The Free version has a ten minute trial, a thirty day trial period (or 300 second until unlocked) then a 30 second delay on opening sketches and is code limited to 150 lines. The Pro Version has no limitations.

The benefits and features of an Arduino Simulator are:

- Speed up Arduino designs
- Try out other Arduino boards first and see the pin-out configurations
- Demonstrate a project to a potential customer remotely
- The ability to educate and demonstrate the inner workings of an Arduino sketch
- Run a sketch without the hardware, or prior to purchasing hardware
- Debug a sketch, step through or run with a breakpoint and see the variables change
- Find out number overruns such as setting a byte to 256, assigning a new value to a const variable, or use library routines without including the library
- Graphically view L:CD and colour screens
- Trace and Error logs
- Improved Serial Monitor which allows an interface between the Simulator and the real world
- Simple logic analyser to view digital pins graphically
- Automated testing (Unit Testing) to check that sketches will provide a certain output
- Variable watch which shows only the last variable changed or selected variables
- Variable insight which allows for variables values to be viewed by clicking on them in the Sketch Window (ensure Menu Option is turned on)
- Design a new board inside the Simulator using the Hardware > Save Settings and Load Settings items.
- Many more features and continuous improvements with regular updates

The SIMFORARDUINO Pro Version license is licenced to one user only for use on up to 3 computers or to 100,1000 or 10000 computers for SiteWide Educational licences at a discount.

Press F1 inside the Simulator to open context sensitive help.

What's new

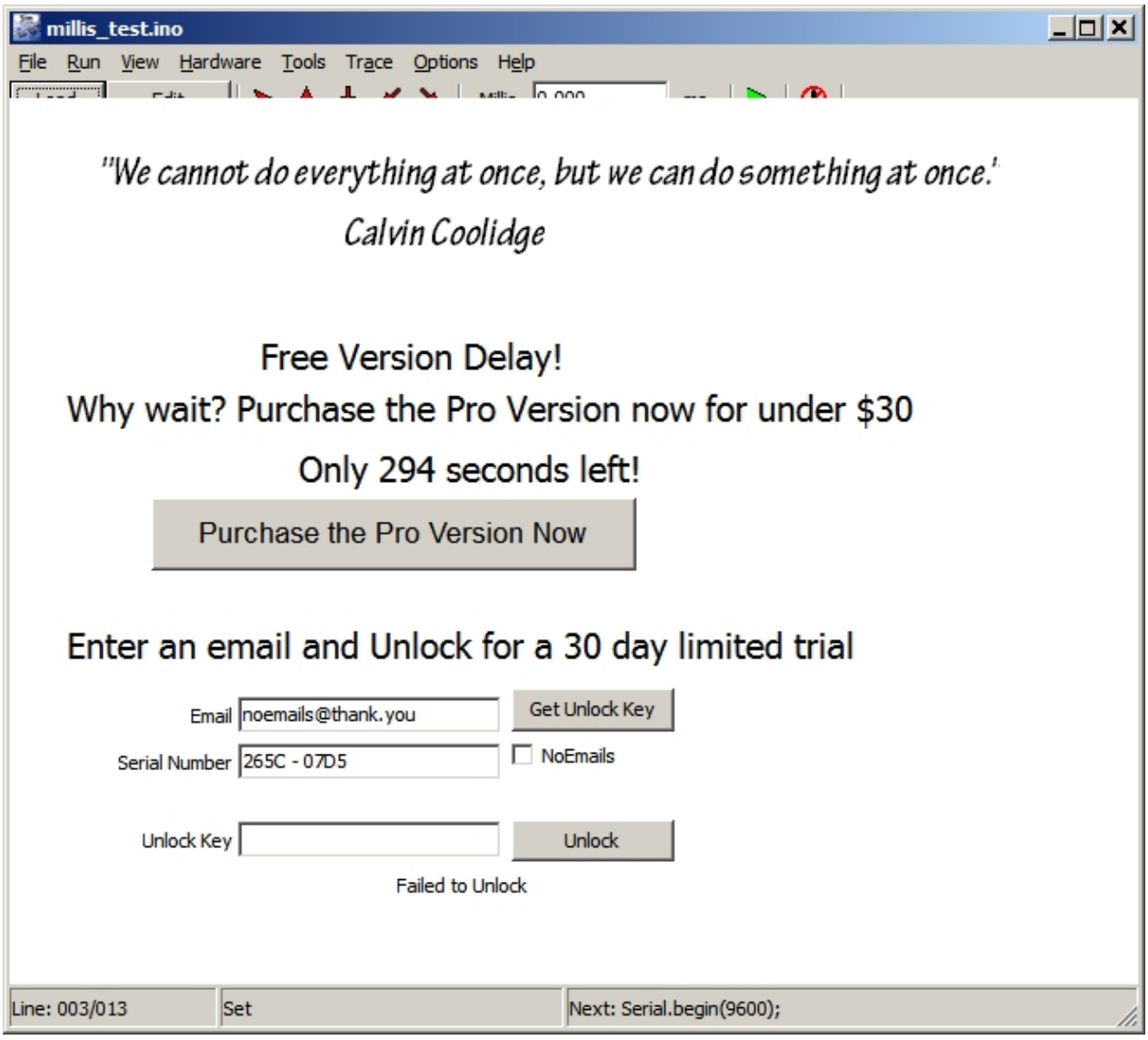
Version 0.99E - Dec-2014

1. Add adjustable syntax colours
2. Move settings into separate folders in registry
3. Add timer interrupt
4. ms goes red and pressing triggers timer interrupt
5. Add 4 breakpoints
6. Allow multi char arrays
7. Fixup char arrays - convert to strings
8. Improve return function
9. Add in undock variables
10. Add in LED support using timer interrupts
11. Addin 4 breakpoint conditions
12. Improve breakpoint hidden command // breakpoint1(12,3,i=0)
13. Improve library directory finding
14. Allow arrays to be part shrunk in variables

15. Fixup line numbers to start at 1 where possible

Free Version Differences

The Free Version is identical to the Pro Version, apart from the addition of a timer delay window and a Code Limit of 150 lines. The Free version is supplied as a demo version to allow users to trial the Simulator for Arduino program. The logic and operation of the Simulator for Arduino program is identical apart from the timer delay and code limit. The Pro Version License comes with the latest Simulator version and Upgrades until the end of the Calendar year.



Simulator or Arduino

Simulator or Arduino is a good question and *Simulator and Arduino* or even *Arduino and then Simulator* is a good answer. The Simulator is not a replacement for Arduino but an add-on product to the Arduino development kit and has been designed as a training/demo/debugging tool. It has not been designed as a development or compiling tool. We fully support the Arduino project and encourage new users to buy and try a real Arduino kit. For first time users, please refer to the book "Getting Started with Arduino" by Massimo Banzi.

To develop code for Arduino, a good starting point is to use the Arduino Uno together with the Arduino IDE. For larger projects, the Arduino Mega or Due may be more suitable. This Simulator has not been designed to check for syntax errors, and we recommend only using the Simulator for debugging only after the Arduino sketch compiles in the Arduino IDE.

We recommend using Official Arduino kits since these are well designed and support for the Arduino project leads to more cool new development boards and IDE upgrades.

Getting Started

Getting Started.

Download the zip file, extract the setup.exe file. When finished, run this program to install the Simulator for Arduino program. For the Free Version, there is a ten minute trial period then an unlock process. To Unlock the Simulator, click on the Get Unlock Key button and then copy the Unlock Key back to the Simulator and clicking Unlock. Please email us if there are any issues with this process.

System requirements

The System Requirements are a PC Pentium II or later.

The Simulator for Arduino program will run on the following systems:

WINDOWS
- Windows 2000
- Windows XP
- Windows Vista
- Windows 7
- Windows 8 and 8.1

MAC
- Windows using Parallels
- Windows using VMWare VM systems

LINUX
- Wine on Linux (windows emulator)

Untested on Windows 98,ME and 95. If the Simulator does run on any of these systems, please let us know by sending an email to support@virtronics.com.au

Getting help

For help, please email support@virtronics.com.au and please attach the sketch, or click the email button inside any Error Message. Replies sometimes can take a few weeks.

Also, please make use of the forum (http://forum.virtronics.com.au/) which may be a little quiet sometimes. There are already many questions and answers on the forum and threads can be subscribed to in order to be emailed about new replies.

Before emailing, please check that the sketch compiles in the Arduino IDE first.

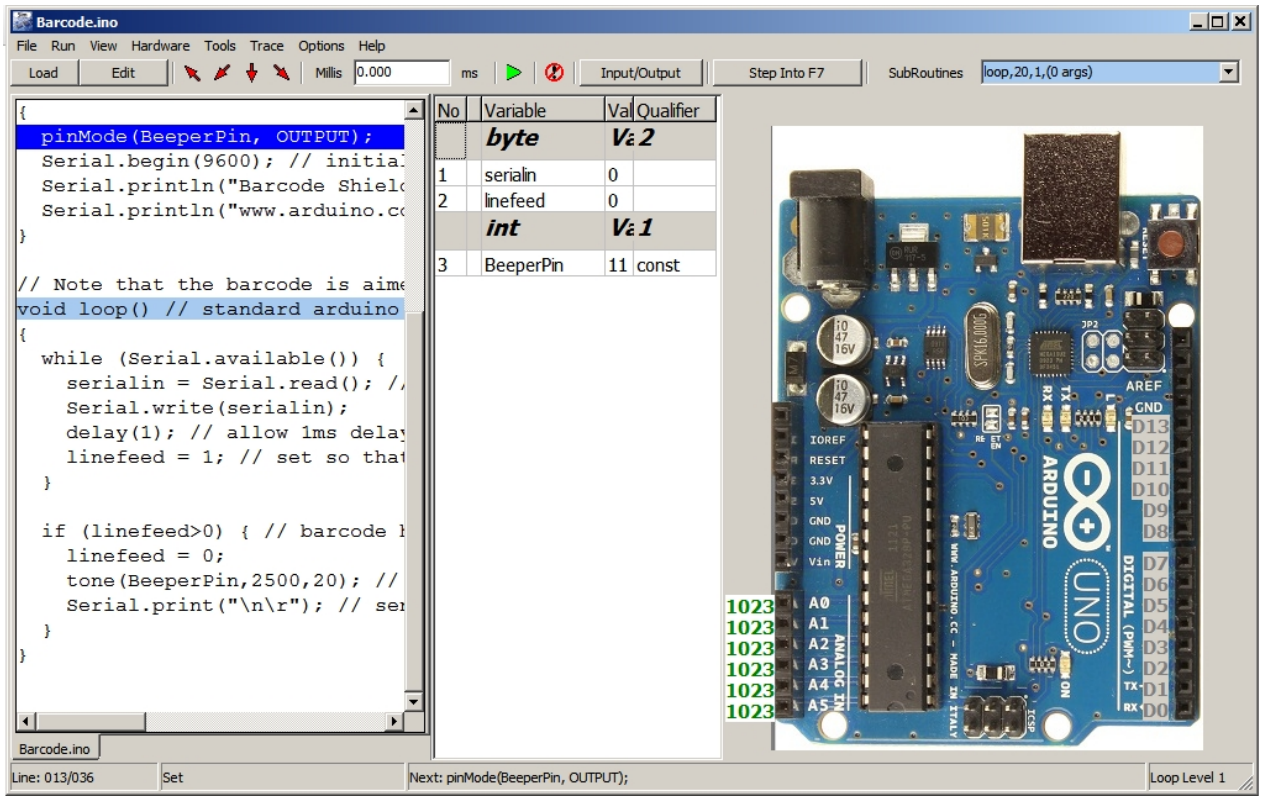
Using the IDE

The Simulator for Arduino IDE has several sections. These are:

- The Menu system which contains options and settings
- The Shortcut Toolbar provides single click action for commonly used operations
- The Program or Sketch window
- The Variables area
- The Arduino Area

Note that the order of the Program window, Variables are and Arduino picture may be swapped by Selecting View|Arduino on Left. The preferred method is to have the Program Window on the left so that the work flow is input,data and output. The Arduino picture may be hidden by selecting Options > Show Arduino and clicking on it to uncheck this option.

When the Simulator for Arduino IDE is resized, the Variables window will disappear first since this is seen as the least important window. If the main screen is made smaller, the Program window will then disappear and the Arduino Screen will always be on top. This allows for sketches to be simulated as they would appear for real. There is also a Minimize button designed just for this purpose.



Program Window

The Program window area shows the program listing, and the cursor will always be in the vertical center of the screen. The Program Window allows for the program to be stepped through. When selected, the following keys may used to run the program:

- F7 key may be used to step through the program and
- F8 can be used to step over subroutines while still executing all the code inside the subroutine.
- F9 may be used to run the program
- Shift F8 may be used to step out of a routine
- F2 may be used to force a hard reset
- F3 may be used to open the Find window to search for text
- F6 may be used to edit the sketch
- See the Shortcut Toolbar for more info on the step and run icons

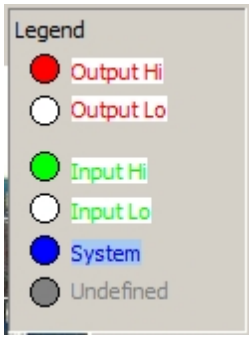
```
// Saved by Simulator for Arduino V0.92
// Simulate Uno
This is a test /* Text */
/* Text */ This is a test
This is a test /* Te
xt */ This is a test
xt */ This is a test /* Te */
int y = 23 ;

char x = 1;
int y = 23 ;
long fd = 34;
String r ="1234";
#define dghs 3456

void setup() {
 pinMode(13,OUTPUT) ;
 pinMode(12,INPUT) ; digitalWrite(12,HIGH) ;
 Serial.begin(9600) ;
}

void loop() {
 y = digitalRead(12) ;
 Serial.print(y) ;
 digitalWrite(13,y) ;
 if (y==HIGH) Serial.println("Hi") ;
 if (y==LOW)  Serial.println("Lo") ;
 delay(500) ;
}
```

A new line may be selected by right clicking anywhere in the Program or Sketch window and selecting Set Next Statement.

The Statusbar shows the Line number, Last Line number and the next code to be executed. If the AutoStep time is set to less than 2ms, the Status-Bar will not update until Run or AutoStep is stopped.

Special Simulator hidden code may be used to setup a particular hardware configuration, and this allows the setup configuration to be saved inside the comments of the sketch. See Hidden code for more detail.

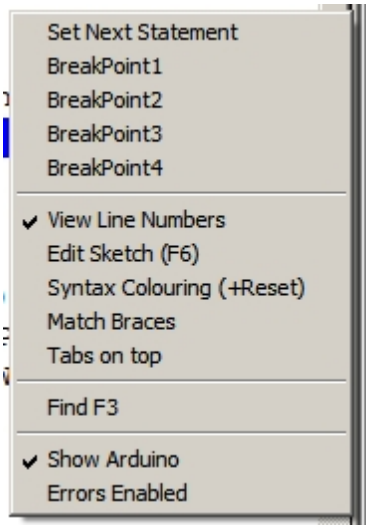
Legend



The Legend may be turned on by selecting View | Legend

The Legend shows the colours and states of the Digital output. The AnalogWrite value of a digital pin will be displayed to the left of the digital pin when the analogWrite() routine is used.

Program Popup Menu



The Program Area Popup menu has these options:

- Set Next Statement - change the program line
- Breakpoint1-4 - right click to add a breakpoint in red or clear it. Left click on the breakpoint to see which number it is. Also refer to breakpoint condition in the Edit Window.

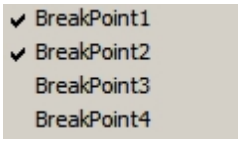
- View Line Numbers - turn on the line numbers in the Program Window - this is useful for checking subroutines
- Edit Sketch - allows the sketch to be edited in the Edit Window
- Syntax Colouring - show comments as light gray, #if hash commands in medium grey, definitions in green, constants in Olive and subroutines as blue
- Match Braces - find the opening or closing bracket for the current line
- Tabs On Top - places the tab with the filename at the top if checked or else the bottom

- Find - open the Find window to find text inside the sketch

- Show Arduino - uncheck to hide the Arduino picture and allow more screen area for your super sketch
- Errors Enabled - turn off to prevent any annoying errors, and hope for the best. Please note this must be manually turned on again to show any sketch errors.

BreakPoint

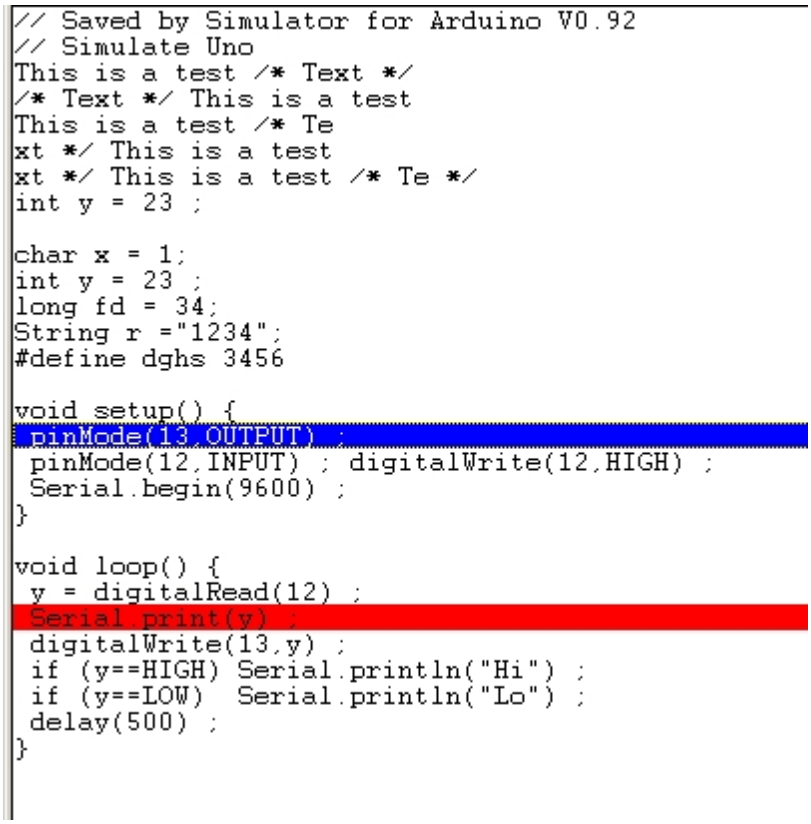
A breakpoint can be set by right clicking anywhere in the program window.



The pop-up menu has an option which can be clicked to add or clear the breakpoint. Please note that a breakpoint condition can be added in the Edit Window to only break at the red line on a certain condition.

If a BreakPoint is added, the breakpoint line will be shown in red. This is commonly used with the AutoStep or F9 command. The breakpoint can also be saved inside a comment using Hidden Code
the syntax
// breakpoint(20)
will set a breakpoint on line 20. If a breakpoint is set on a new line, the previous breakpoint will be cleared. To turn off a breakpoint, select the same line and select Breakpoint.

Left click on a set Breakpoint to find out which number it is. Right click to clear it.



Bare Minimum

Bare Minimum will load the bare minimum needed for a new sketch. Select Run|Edit Sketch and the press the rightmost button to load the Bare Minimum to start writing a sketch. The text will appear as shown below. Modify and then click the button Save File As to save the sketch to a new filename.

```
// Simulate(Uno) link sketch to Arduino board here

void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Loading a Sketch

To load a sketch, select File|Load Sketch, press the Load Sketch button at the top left or press F4. Navigate to the correct folder and select the *.ino file (or *.pde for previous Arduino versions). C files can now be simulated since the C and Arduino syntax is compatible.

Running a Sketch

After a sketch has been loaded, the first executable line of code will be highlighted. Now press the Step Into button or the down arrow. Alternately, the F7 single step or F8 button may be pressed. The F9 button allows the program to be autorun with a 5ms interval (adjustable) between steps. This 5ms time can be adjusted in

the Edit Window and for training, this can be set to 500ms to demonstrate in slow motion the operation of a sketch.

The middle section of the screen displays the variables grid, and these are organised into Bytes, Chars, Int(eger)s, Long, Strings and Defines. Any value in the grid can be changed by selecting it and typing in a new number.

Hidden code

Special Simulator hidden code may be used to setup a particular hardware configuration, and this allows the setup configuration to be saved inside the comments of the sketch.

This hidden code can be one fo four formats:

- Simulate(Uno) - Simulate the particular Arduino board - please note that Simulate(Uno_LCD.txt) will also work if the text file is found
- breakpoint(line,tab, condition) - will setup a breakpoint at the line on the correct Tab, (leftmost=1) with the condition. If the condition is left blank, it will not be set
- SimulateLCD(x,y,width,height,pixel width,pixel space, backlight color, on color, off color) will setup a LCD area on the Arduino picture. The colors are 24bit in a BGR format where 255=Red.The pixel width should be no more than 5)
- SerialIn(data) will load the Serial Port input data with the data inside the parentheses
- Stimulus(stim.sti) will load a Stimulus file. Stimulus actions are shown in the statusbar after the Next line data and will happen at set times. View the _Sim Test\v98g\test.sti file for a sample

Sample Stimulus file

```
20,pinMode(1,OUTPUT); // at 20us make digPin1 an output
50,pinMode(5,OUTPUT); // at 50us make digPin5 an output
100,digitalWrite(5,HIGH); // at 100us turn on digPin5
```

Note that there is also a virtual library <Graphics.h> which allows for colour screen rendering. See the Graphics.ino demo sketch under _Sim Test\v0.98 checks

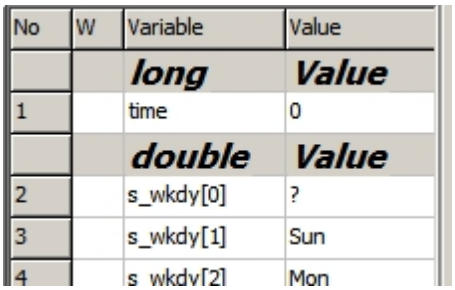
Variables Area

The Variables area shows the status of all the variables used in program and their current value. The Value may be changed by clicking in the relevant cell and changing the value.

Double click in the first four columns to auto-resize the variable area.

Double Click in the Qualifier column or right click and select Watch to turn on Watch mode. This allows only selected variables to be watched.

The Qualifier column typically shows if the value is a const, unsigned, static, volatile or pointer type (Note that Pointers are not fully supported). The qualifier number in a heading row (shown by the gray shading) shows how many variables of that type have been defined.



No	W	Variable	Value
		long	***Value***
1		time	0
		double	***Value***
2		s_wkdy[0]	?
3		s_wkdy[1]	Sun
4		s_wkdy[2]	Mon

When returning from a subroutine, the stack will be returned the the original state, and any variables defined in the subroutine or passed as arguments will be cleaned or removed.

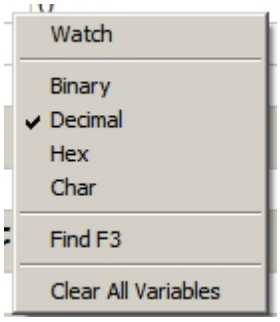
Right Click in the Variables area to bring up the Pop-up Menu.

The second column is the watch column, and clicking in this column will show a w. The right click Pop-up menu can then be selected to show only Watched variables. This is useful when only a few variables are relevant amongst several hundred. The gray area in the W column may be clicked to show or hide all the variables for that type. The last variable changed will always be shown for the last program step.

Arrays can be folded by clicking in the second or Watch column and a + sign will show that the Array has been folded. When folded, only the first element of each array will be shown and for character arrays, the string value of the whole array can be easily viewed. Alternately, the array may be folded at any element of the array and in this case, the elements above will be shown, but the elements below will be hidden. Click the + again to unfold the array

Clicking in the Qualifier column will enable or disable the watch, and the W in the second column indicates if the Watch is on (W) or off (blank). The last variable changed will always be shown.

Variables Popup Menu



The Variables Area Popup menu has four options:

- Watch - only display variables which have a w or + in the second column of the grid

- Bin - display Variable value in binary
- Decimal - display Variables value in decimal
- Hex - display Variables value in hexadecimal
- Char - display Variables value in Character format and hexadecimal

- Find - open the Find window to find text inside the variables area

- Clear All Variables - useful for header files which have a #ifdef FILE_H at the beginning. Clearing the variables, then allows the header file to be stepped through. NOTE: doing this in the middle of stepping through a sketch will cause errors since the Simulator will no longer be able to find or set variables.

Arduino Area

The Arduino picture area allows for the digital inputs and outputs to be displayed along with the Analog input and output values. The digital pins are defined up to D53 for the Mega and D13 for the Uno.

The Analog Pins are defined as A0 to A5 for the Uno with A0 equal to 14. Other boards are setup according to the Arduino configuration.

The Simulator also allows for custom boards to be setup.

Arduino Popup Menu

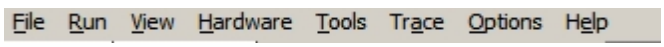
The Arduino Area Popup menu has these options:

- Arduino on Left - display the picture of the Arduino on the left and the Program window on the right
- Legend - display the digital pin legend on the top right of the Arduino picture
- Show Volts - this option allows for the analog number to be displayed as volts in the format x.xx (accurate to 0.01V)
- Minimize - turn off the Tool bar and menu to make the Arduino picture as minimal as possible. the program can be resized to then only show the Arduino picture and the Title (which shows the sketch name).

- Load Sketch - load a sketch into the program window - useful in minimize mode
- Input/Output - show the input/output window - useful in minimize mode

- Uno - set the Arduino board to a Uno type
- Due
- Papilio
- Esplora
- Mega - set the Arduino board to a Mega type
- Mega 75% - same as above but with a small picture to fit most laptop screens
- Leonardo
- LilyPad - wearable Arduino
- Nano
- Other - configure your own board

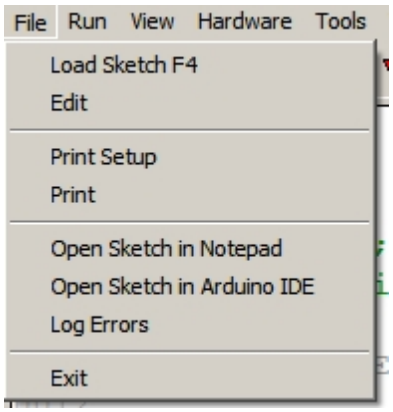
Menu System



The Menu system has eight sub-menus. These menus allow the Simulator to be customised.

Note that right clicking in the toolbar will bring up a Language popup menu.

File Menu



The File Menu Items are:

- Load Sketch F4 - Load a Sketch into the Program Window
- Edit Sketch F6 - Edit a Sketch in the Edit Window

- Print Setup - open the Printer Setup Dialog Box
- Print - print a scaled picture of the program

- Open Sketch in Arduino - this opens the sketch in the Arduino IDE if Arduino is the default program to open sketches. If this does not happen, right click a .ino sketch, select Open With and then Choose Default Program and set to Arduino.exe
- Log Errors - all errors will be logged to the ErrorLog + date.txt file Note that Errors here mean program and simulator errors

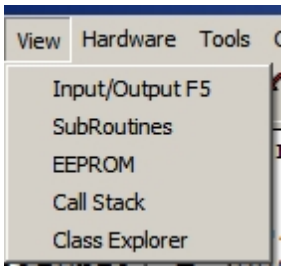
- Exit - close the program

Run Menu

The Run Menu Items are:

- Hard Reset - reset the program normally
- Reset - reset the program to the start or Setup routine and clears all variables
- Step Into - step one line through the program, and step into any subroutines
- Step Over - step one line and step over any subroutines running the code inside the subroutine
- Step Out of - Step out of a subroutine
- Run - perform a step once every time period which is 5ms by default -see Shortcut toolbar
- Soft Reset - a new function to reset the sketch without needing to reload all the #include files

View Menu

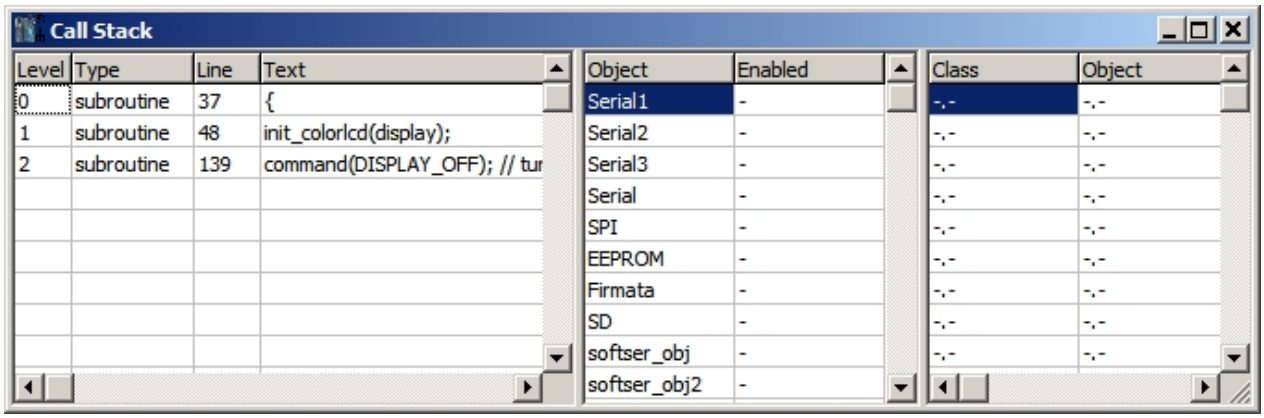


The View Menu Items are:

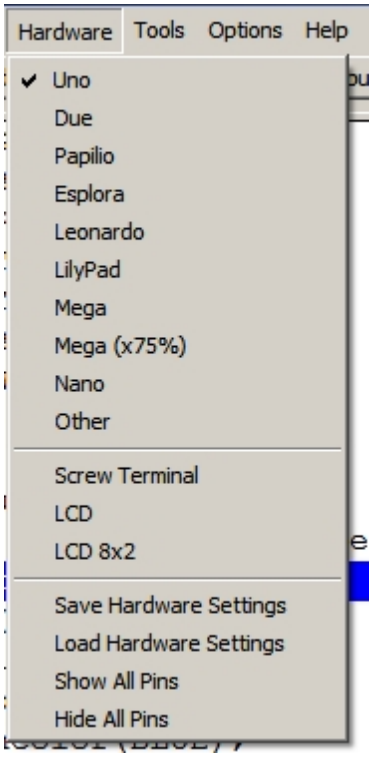
- Simulation data - open or Close the Simulation input/output window
- Subroutines - View the Subroutines and Line Numbers - double click on any subroutine to see it in the sketch window without changing the program counter
- EEPROM - View the EEPROM data
- Open the Call Stack Window
- Open the Class Explorer - view class objects with their values

The Call Stack Window is very important in debugging since it shows the origin of the current loop which may be a subroutine, while loop or any other condition. The loop level is show at the right in the Status bar.

The middle area is the standard Arduino library objects, while the right area is for user defined C++ objects.



Hardware



The Hardware Menu Items are:

- Uno - display the Board picture as an Arduino Uno board
- Due - display the Board picture as the ARM-based Arduino Due
- Papilio - display the Board picture as the Papilio One FPGA devkit
- Leonardo- display the Board picture as an Arduino Leonardo board
- LilyPad - display the Board picture as an Arduino LilyPad board
- Mega - display the Board picture as an Arduino Mega board
- Mega (x75%) - a smaller version for laptops
- Nano - display the Board picture as an Arduino Nano board
- Other - open up any picture file preferably 350 pixels wide to overlay on the Arduino picture

- Screw Terminal - Show the Virtronics custom Screw Terminal board
- LCD - Show the Virtronics custom LCD board with a 16x2 LCD fitted
- LCD 8x2 - - Show the Virtronics custom LCD board with a 8x2 LCD fitted

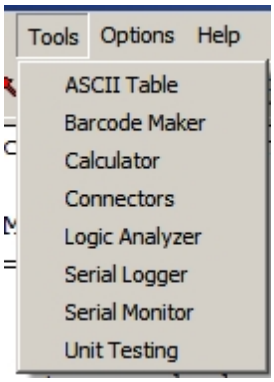
- Save Hardware Settings - save hardware settings to a .txt file which can be edited later
- Load Hardware Settings - load hardware settings from a .txt file
- Show All - Show all the pins, pin names and analog values for the Arduino board
- Hide All - Hide all the pins, pin names and analog values for the Arduino board

The hardware settings are saved to a text file with the format shown below. Clicking anywhere on Arduino picture will show the X,Y coorindates in the Status Bar and also save them to the clipboard.

```
Picture(Uno,350,506) // pix,width,height
resetSw(345,105) // Reset Switch x,y
onLed(273,454) // power led x,y
txLed(276,256)// transmit led x,y
rxLed(260,256)// receive led x,y
d13Led(309,256)// d13 led x,y
crystal(192,160)// crystal x,y
usb(250,32) // usb x,y
```

maxPin(13,5,19)// maxDigitalPin,maxAnalogPin,maxAllPins
externalInterrupts(2,3) // External Interrupts int0pin int1pin ...
LCD_Enabled(-29,-16,200,200,0,0,0,0,0)// x,y,w,h,pixel w, p-p,blit_color,oncolor,offcolor width=6*lcdx*p-p height=10*lcdy*p-p - for graphic screen
digitalPin0(367,490,340,490,396,490) // pin x,y pin label x,y analogValue x,y
...
analogPin0(45,406,71,406,2,406) // analog label x,y pin x,y analogValue x,y
...

Tools



There Menu Tools are:

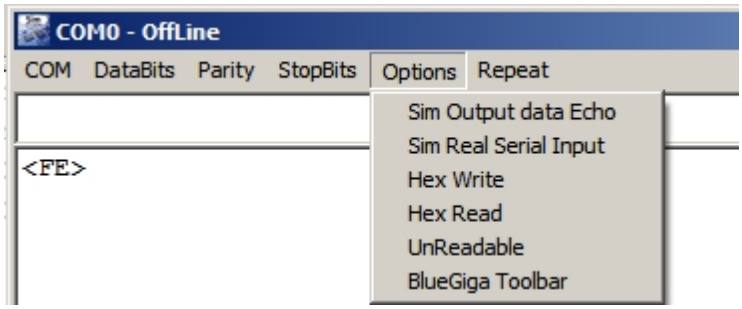
- ASCII Table - view the full ASCII table with character, hex and binary representations. Useful to most programmers and now viewable with large Text.
- Barcode Maker - the ability to set and print various barcodes - useful for the Virtronics Barcode Scanner Shield
- Calculator - a simple programming calculator for adding and multiplying hex numbers
- Connectors - view some common electrical connectors such as the serial D9 connector, USB typeA, USB mini and DC jack
- Logic Analyzer - view digital pins 0-13 graphically using the trace - automatically activates the File|Trace option
- Serial Logger - show Serial data graphically from the real World or from the Simulator
- Serial Monitor - an improved Serial monitor with a classic Bluegiga toolbar Note the output can be directed to a virtual serial port
- Automated Testing - the ability to automatically test series of sketched or commands. Very useful.

In all these Tools (except Automated Testing), the keyboard commands such as F7 to single step and F8 to step one line at a time will work.

Serial Monitor

The Serial Monitor is provided to be able to view real Serial Data.

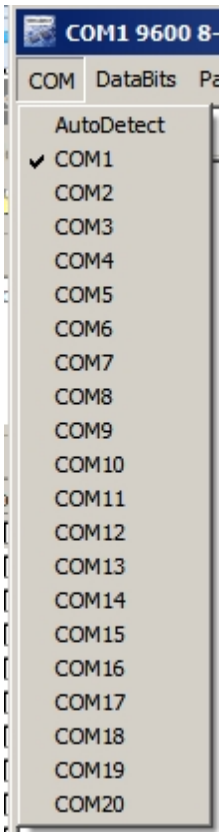
The Serial Data will be transmitted from the Simulator out the designated Serial Port.



The Options are:

- Sim Output data Echo - allows for data transmitted using the Serial.write or Serial.print commands to be Echo in the transmit box
- Sim Real Serial Input - great feature to allow incoming serial data to be ported to the Serial Indata in the Input/Output window
- Hex Write - send outgoing data as two hex characters followed by a space
- Hex Read - display incoming data in Hex format
- Unreadable - show unreadable characters as their character representation otherwise display as <hh> where hh is the hex code
- BlueGiga Toolbar - for those using classic Bluegiga device such as the ones from ESDN

In the COM Menu item, select AutoDetect to view only the available Com ports.



Unit Testing

Automated testing is an extensive page setup to automatically test sketches against set outputs. This helps with testing the Simulator, and while it takes a lot longer to setup a test script, in the long term, this will save much time.

At the bottom are 4 buttons labelled T1 to T4 and new. These are for routine tests which are all text files found in the _Sim_Test folder:

- T1 is for the Examples from 1.Basic up to and including ArduinoISP
- T2 is for the rest of the Official Arduino examples using the standard libraries such as SPI, Wire and Ethernet
- T3 is for bug fixes and known previous issues
- T4 is for a Simulator-oriented test script test each function as it is listed inside the Simulator source code
- New shows the latest test scripts added

Each line in a script file conforms to the following basic syntax:

test(function or variable, expected integer result expected float result, expected string result)
NA means not applicable or No test

Extra single letter functions are:

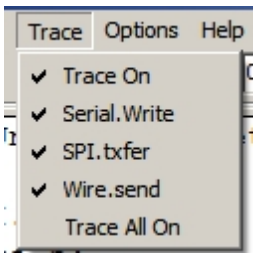
- + for single step
- @for read data from Input/Output port using the integer result as the line number (SoftSerial,Serial,SPI,EthernetClient,EthernetServer,Ethernet, Wire,LCD, KeyBoard,File & Udp)
- ! load a file
- ~digPin - read a digital pin state with the pin set by the integer result and check the colour is the same as the expect float result read as an integer

The analog(x,val,NA,NA) can only be used once at the start of the script

Examples(Int,Float,Str,Output)
analog(0,123,NA,NA)

```
// ***** 0 BASIC *****
test(!01.Basics\AnalogReadSerial.ino,NA,NA,NA) //  1.1 AnalogReadSerial
test(+++sensorValue,123,NA,NA)
test(++@Serial,0,NA,123)

test(!01.Basics\Blink.ino,NA,NA,NA) // // 1.2 Blink.ino
test(+~digPin,13,0xffffff,NA)
test(++~digPin,13,255,NA)
test(+millis() ,1000,NA,NA)
test(+~digPin,13,0xffffff,NA)
test(+millis() ,2000,NA,NA)
```

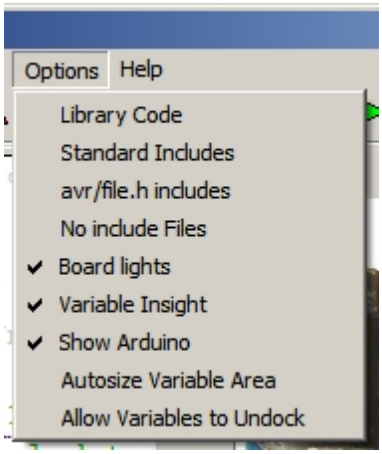
Trace



- Trace On - save all program steps to a text file with the filename based on the current date. the digital pins state is also saved for the Logic Analyzer
- Serial.Write - trace into all the output bits of each serial.write instruction and add time to the millis function depending on the baud rate
- SPI.txfer - trace into the output bits of each 8 bit transfer and add 24us for each transfer (3us for each bit)
- Wire.send - trace the output bits of each I2C output byte
- Trace All On - turn on all the options above or turn off if checked

The middle three options are specially designed for the Tool > Logic Analyzer and allow the digitalPin changes to be view in real time similar to the way an oscilloscope would display the voltage levels

Options



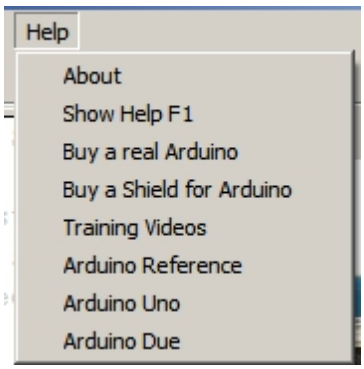
The Option items are:

- Library Code - open and display the actual code for a library such as <SPI.H> - note the Library director should be setup to point to the right directory first - see here for more info.
- Standard Include - open and display the actual code for header files such as <stdint.h> and <inttypes.h>
- avr.file.h - open and display the actual code for any avr/file.h files.
- No Include Files - stop those annoying open file dialogs - may not allow classes to be used.
- Board Lights - Uncheck to be able to run graphic colour screen sketches without the board leds activating
- Variable Insight - Uncheck to turn off the sketch window hover function to see variable value during stepping
- Show Arduino - Uncheck to hide the Arduino picture
- Autosize Variable Area - automatically resize the variable area when reset or a new sketch is loaded
- Allow Variables to Undock - Checking this option allows the variables area to then be dragged out as a separate undocked window. If the variables area window disappears or is closed using the X, select this option again to make the variables area re-appear,

Please note that having any of the options turned on can cause many errors since these files use high level C++ syntax structures so these options should be used with care.

On the positive side, these libraries allows for library code to be easily inspected. This can be very useful when adding a colour display from an online store such as Adafruit and then being able to open the Adafruit_GFX and Adafruti_SSD1206 library to see how the SPI port drives the colour display.

Help



The Help Menu items are :

- About - Show the Program version
- Show Help F1 - show this help - Note: this can also be activated by pressing F1

- Buy a real Arduino - open the Buy Arduino page
- Buy a Shield for Arduino - open the Shields for Arduino page
- Training Videos - watch all the Youtube videos related to the Simulator and Shields
- Arduino Reference - display the Arduino language reference page
- Arduino Uno - display the Arduino Uno Page
- Arduino Due - display the Arduino Due Page
- Unlock Key - view the Unlock Key and licence info (in free version only)

The Help | About screen shows the current software version.

The F (for Free version) after the version number indicates the Simulator is identical to the Pro Version but has a delay timer window any time a new sketch is loaded or edited. The Free version has been provided as demo version to allow the Simulator for Arduino program to be evaluated prior to purchase. It is anticipated that after a week of use, it will be more economically viable to purchase the Pro Version than continue to run the Free Version.

Another limitation is a Code limit of 150 lines which will be displayed on the Delay Timer screen.

Dialog Windows

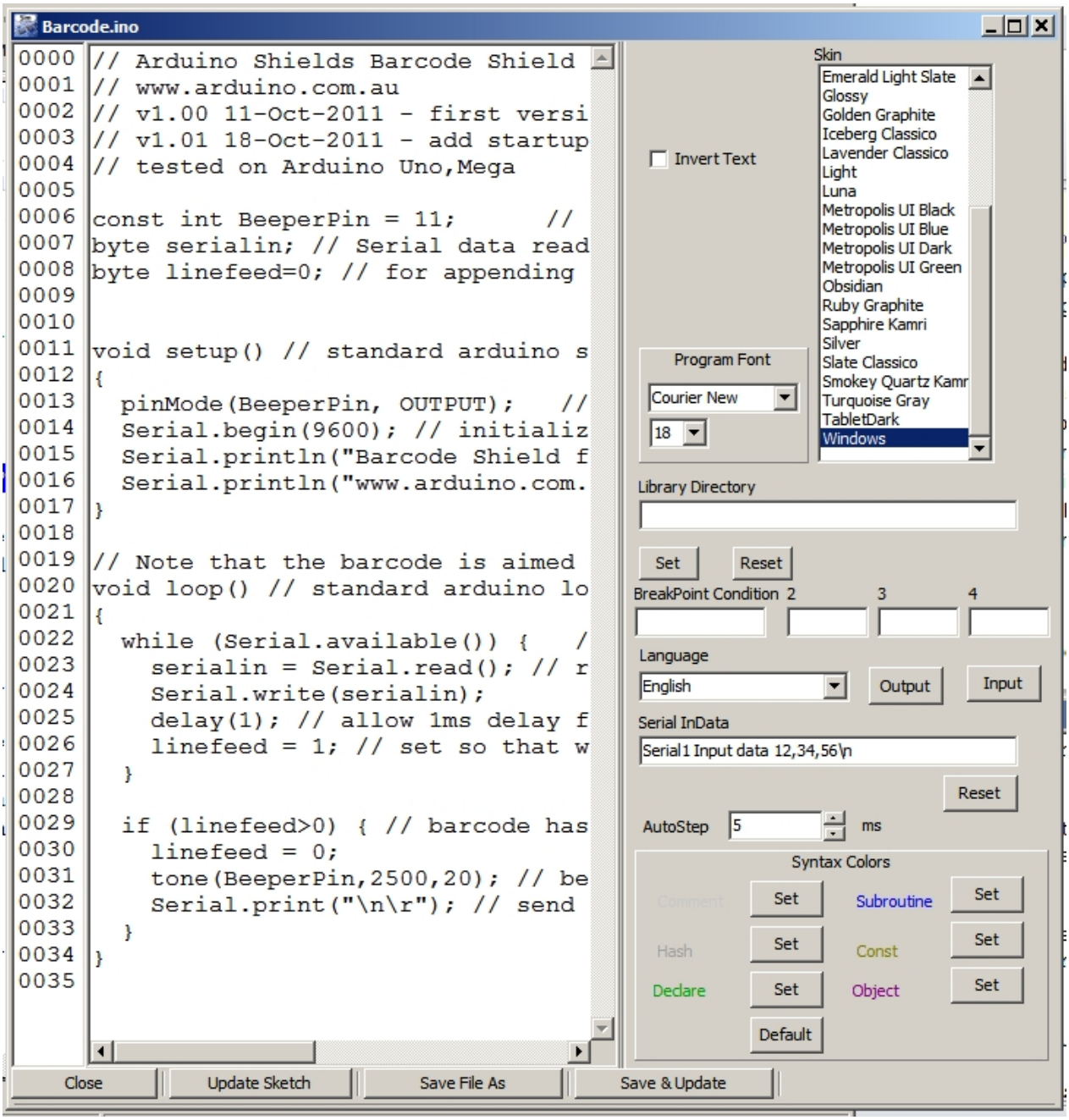
The Simulator has several dialog windows which can display more information when activated. These are:

- The Edit Window
- The Input/Output Window
- View Subroutines
- Error Message

Edit Window

The Edit Sketch Window allows for sketches to be edited and updated or reloaded into the Program window

- Close - closes this window and resumes simulating the program
- Update Sketch - save this edited sketch back to the program window
- Save File As - save this sketch to a new .ino file
- SaveUpdate - save the sketch under the same name and update the sketch in the program window
- Bare Minimum - load the bare minimum code skeleton - see here for more info
- Sketch Font type - changes the font type of the main program window
- Sketch Font size - changes the font size of the main program window
- Skin - this will change and save the skin for the whole Simulator program. The default skin is Windows.
- Library Directory - set for using custom libraries or click Reset to set the Library directory to the default value
- Language - change between English, French, Italian and Other for titles only - also Input and output the language text
- BreakPoint condition - set to break on a certain condition at the red line. The condition can be any expression and can even be a formula - such as i+=10.
- Serial InData - provides the ability to Change the Default Serial - Also available as a hidden code command - Reset to set default value of 123456ABCDEF\n
- AutoStep - the delay between single steps in the run mode
- Syntax Colours - allows the user to adjust and save the syntax colours - Default will reset all colours



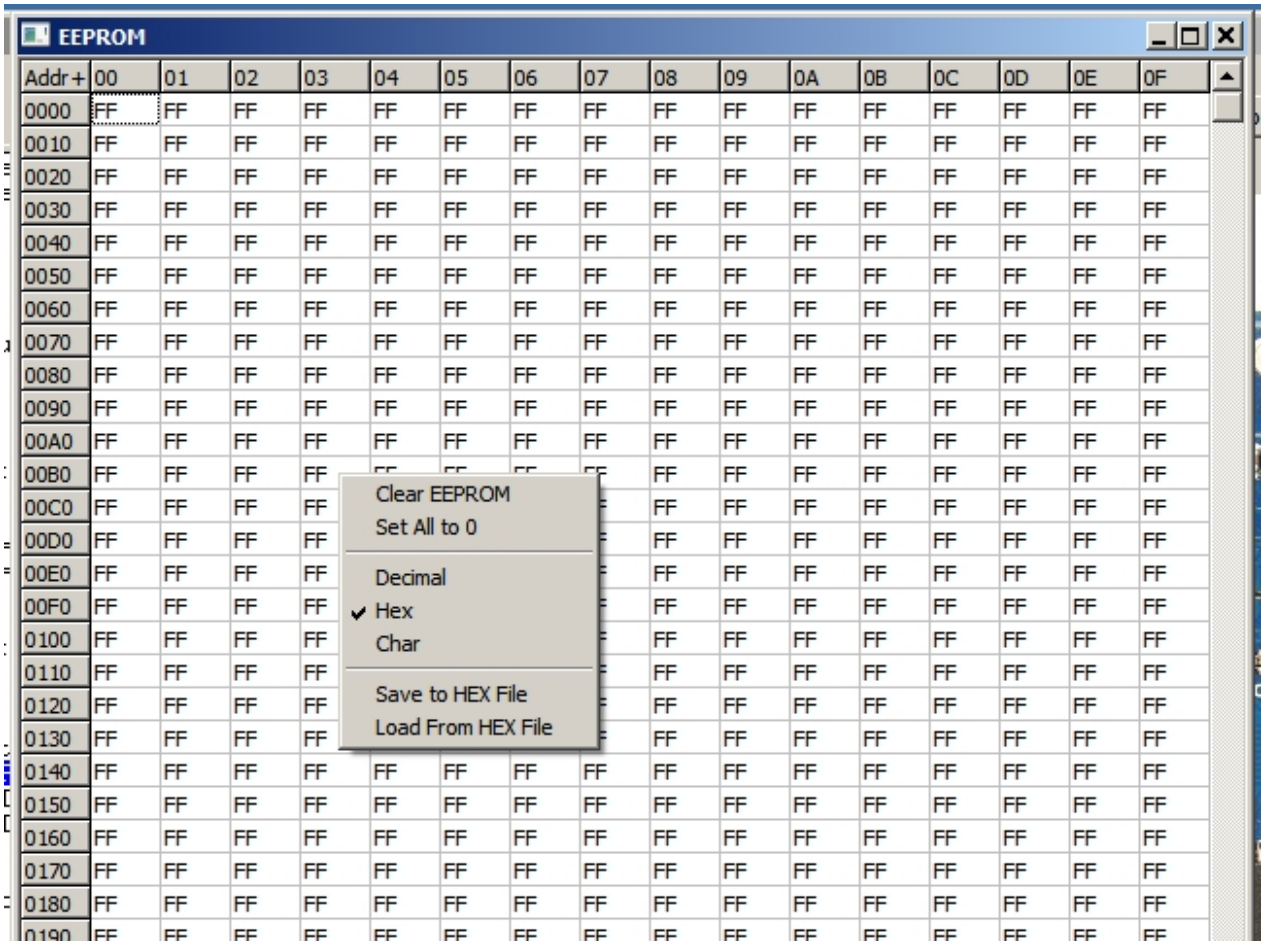
EEPROM Window

The EEPROM window allows for viewing and modifying of the EEPROM memory.

The EEPROM Window also has the added function of being able to view a full hex file contents, the ability to view the contents in different formats (eg Decimal, Hex or Char) and being able to change and save the file. there are many hexfile editors available on the internet and this feature is just a small part of the Simulator

Right click to bring up the popup menu which has several options:

- Clear EEPROM - set all EEPROM memory to FF
- Set All to 0 - all EEPROM memory to 00
- Decimal - display Variables value in decimal
- Hex - display Variables value in hexadecimal
- Char - display Variables value in Character format and hexadecimal
- Save to Hex File - save the EEPROM memory to a hex file
- Load from Hex File - load the EEPROM memory from a hex file



Input/Output Window

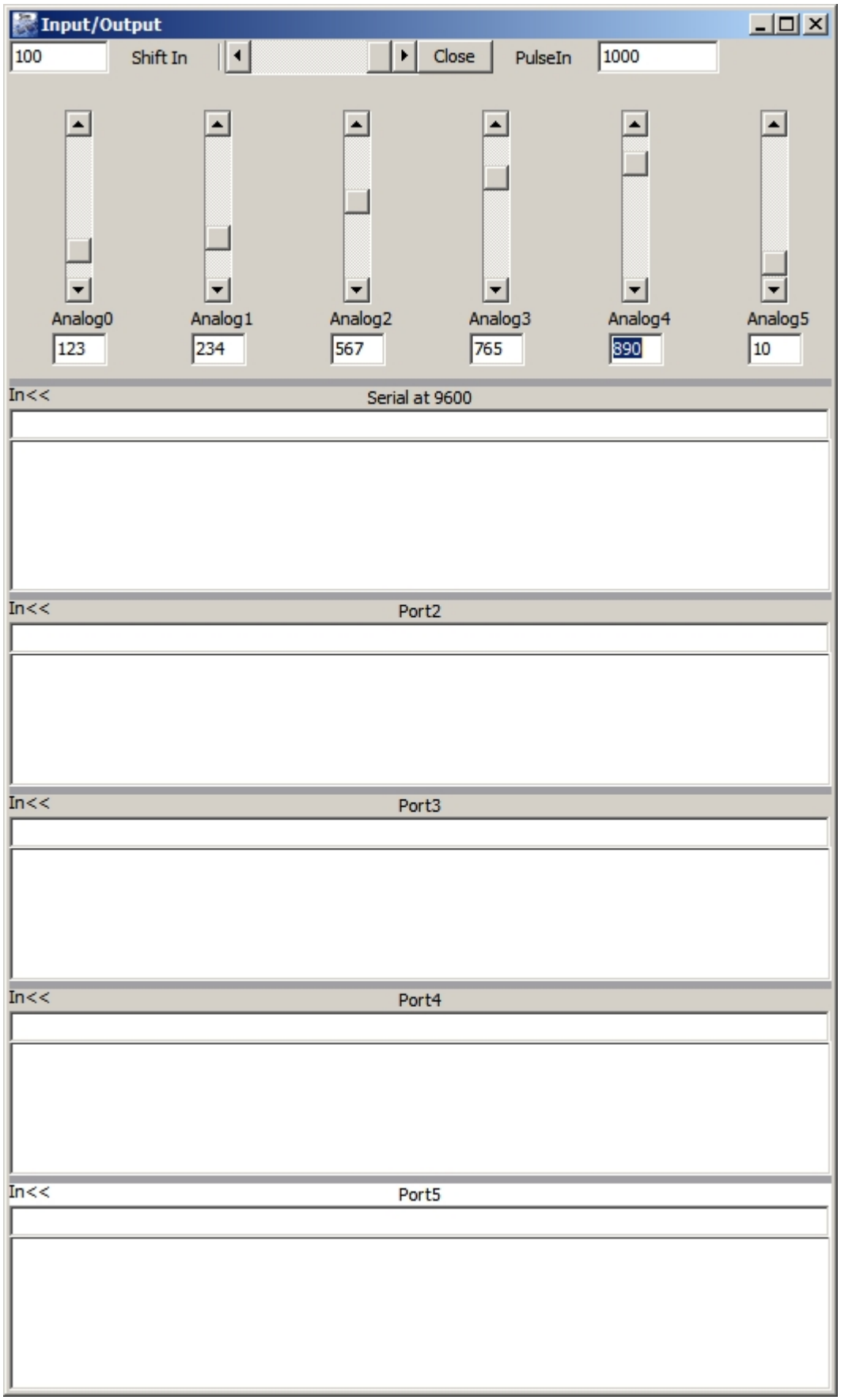
The Simulation Window allows for data to be simulated for the six analog inputs for the Uno or 16 analog inputs for the Mega. The window may be resized to be smaller, and the scrollbars and text boxes will automatically resize to suit.

The ShiftIn and PulseIn allow the return values for these functions to be set.

The Analog Value may be changed by moving the respective scrollbar, or entering a new value in the white Edit box.

Input and Output data is displayed in five ports. Each port has its own line for input data and a box for output data. Each Port can be expanded or contracted to suit the display and resized.

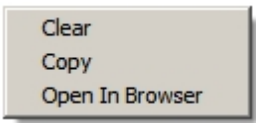
The ports are assigned in the order they are setup.



Simulation Popup Menu

The Simulation Window Popup Menu has two items:

- Show All Ports - to view the five ports and reset the Splitter bars if needed
- Hide Unused Ports - close all ports except the opened ports

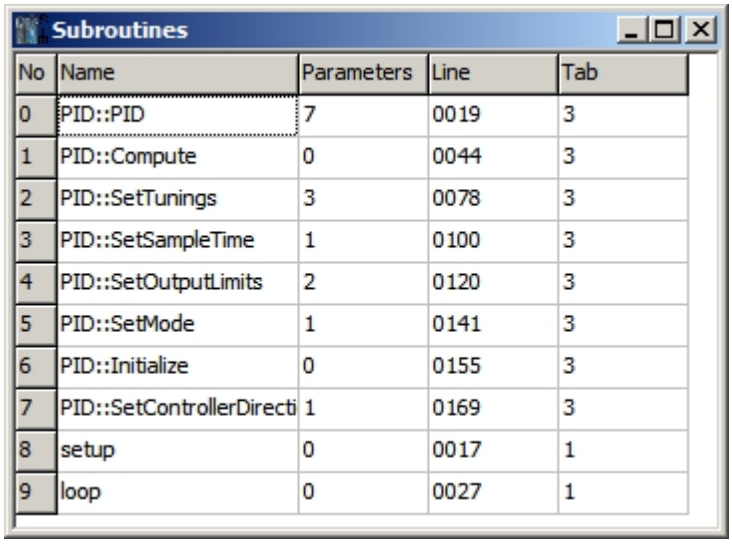


Inside each port is another Popup window with three options:

- Clear - clear the Port screen
- Copy - copy the data to the clipboard
- Open in Browser - open the Port data in the default web browser, useful for sketches using Ethernet

View Subroutines

The View Subroutines window allows for the Subroutines to be viewed along with the number of parameters, the line number and the tab. The line numbers may be viewed in the program window by selecting Line Numbers



No	Name	Parameters	Line	Tab
0	PID::PID	7	0019	3
1	PID::Compute	0	0044	3
2	PID::SetTunings	3	0078	3
3	PID::SetSampleTime	1	0100	3
4	PID::SetOutputLimits	2	0120	3
5	PID::SetMode	1	0141	3
6	PID::Initialize	0	0155	3
7	PID::SetControllerDirecti	1	0169	3
8	setup	0	0017	1
9	loop	0	0027	1

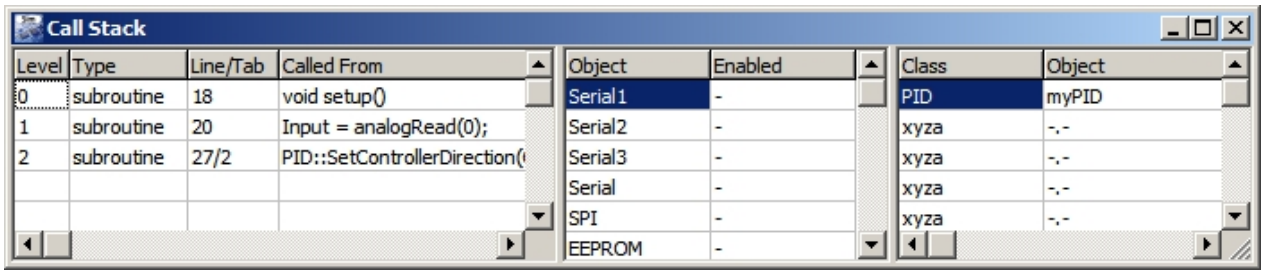
There is now a new Combo Box on the Shortcut toolbar which allows for all the subroutines to be viewed and shown in the Program Window without changing the program counter.

Call Stack

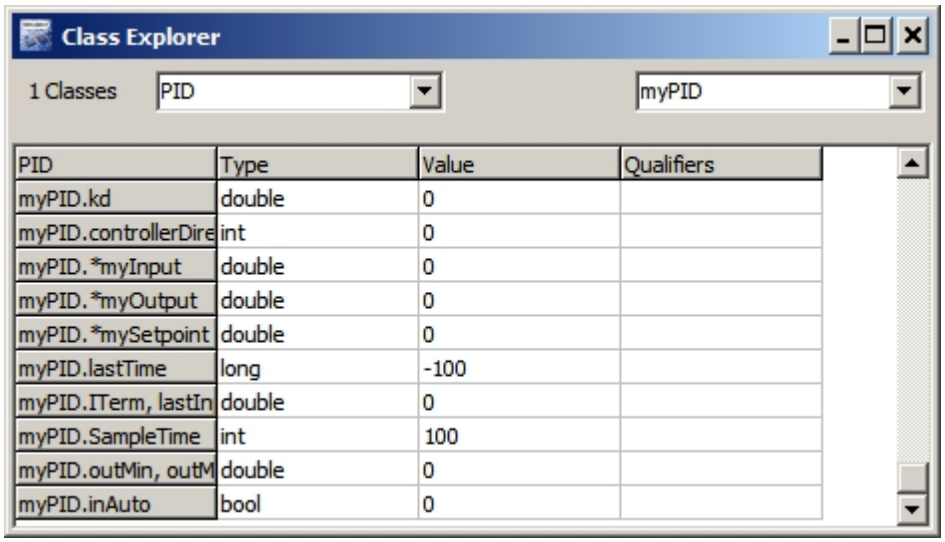
The Call Stack displays the different levels of sketch loops in the left grid

The middle grid shows the Arduino objects, names and Enabled status.

The right grid shows classes found and objects derived from these classes.

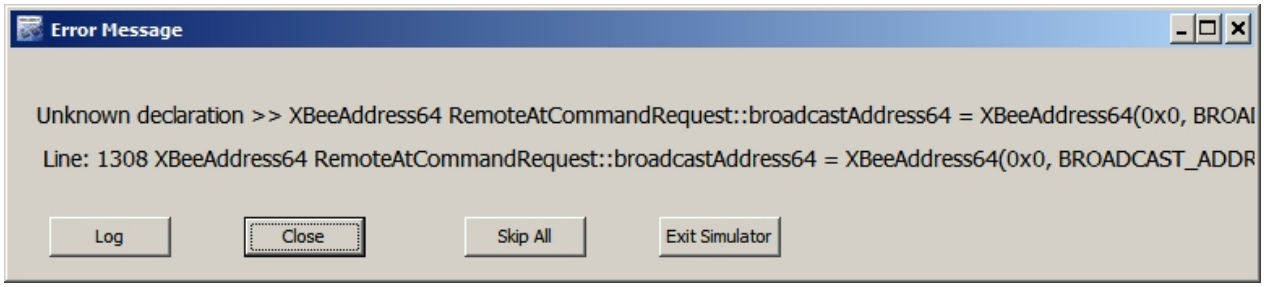


Class Explorer



The Class Explorer allows for all the class variables to be viewed and inspected. These may only be modified in the Variable Area at present.

Error Message



The Error message screen displays any code which the Simulator cannot process or which the Simulator decides if an illegal operation such as:

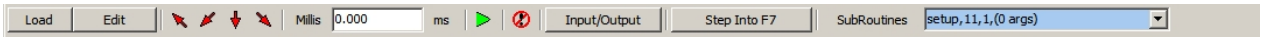
- the above operation, outside the setup() and loop() routines
- setting a digital pin which is set to input
- changing a const(ant) value
- setting a variable to a value outside its limits
- addressing an array element before the start or past the end of the array

The File|Log Errors option allows these errors to be logged to an ErrorLog +date.txt file.

The Log button will only be shown if the Error could be related to internal Simulator logic. If the Error message starts with Sketch Error, the Simulator has guessed that it is more likely the error is in the Sketch. If this is not the case, please email the sketch to support@virtronics.com.au

Shortcut Toolbar

The Shortcut Toolbar provides single click actions for commonly used operations. Right Click on this toolbar to show the Language popup menu which allows easy switching between English, French, and Italian(and other languages when requested).



Button Functions:

- Load - loads a sketch from the last used folder into the Simulator
- Edit - Edit a sketch

- Left up arrow - F2 Reset
- Left Down Arrow - Shift F8 Step out of subroutines excluding setup and loop
- Down Arrow - F8 Step Over - single step jumping over any subroutines
- Right Down Arrow - F7 Single Step and Step into any subroutines

- Millis - the run time in milliseconds (NOTE: ms not seconds). This value can be changed at any time. Each program step takes around 1us

- Run - F9 AutoStep the program one instruction every xx ms The ms value can be edited. This time is the delay between line execution and is not the actual speed of the program since each line may take 1-100ms to execute. If the AutoStep time period is set to 1ms, the variables, blue program traceline and statusbar will not update to increase speed.
- Abort - abort if the program is stuck in any loop

- Simulation - show or close the Simulation input/output window
- Step Into (F7) - this button allows for the program to be single stepped

- SubRoutines - contains a list of subroutines - select a subroutine from the list to jump and highlight in blue that subroutine first line without affecting the program counter

The Milli seconds variable is displayed in a text box. This value can be adjusted by typing in a new value. Each line of program execution takes 1us or 0.001. The commands Delay and delayMicroseconds will add the accurate number of milliseconds or microseconds to the Millis Text box.

Shortcut keys

Several keyboard shortcuts have been added. These are

- F2 for Reset
- F3 for Find
- F4 for Load a sketch
- F5 to open the Simulation Window
- F6 - Edit a sketch in the Edit Window
- F7 to Step Into
- F8 to Step Over
- Shift F8 to Step out of a subroutine
- F9 to Run or AutoStep
- Ctrl-F for Find (same as F3)
- F11 for Soft Reset

Example Sketches

Several sketches have been included for testing the Simulator. These are the Official Arduino examples in the folders 01 Basics to 19 Wire.

In addition there are:
_Sim_Samples - Samples for the latest Youtube videos
_Sim_Shields - sketches to run on the custom Virtronics Shields for Arduino
_Sim_Test - test sketches and sketches sent in for error checking and fixing
_SIM_Extra - sketches with some new tryout features and sketches from the Videos

Libraries - this folder has some test Library sketches and various header files

In a typical install, the sample sketches will be found in this folder:
C:\Program Files\Virtronics

or for Windows 64 bit
C:\Program Files (x86)\Virtronics

Test Sketches

The test sketches are made up of issues with the Simulator and are used to do quick checks on patch versions of the Simulator. The major releases of the Simulator are fully tested against most of the sample sketches.

The test Sketches can be found in the _SIM_Test subfolder.

Language Processing

Keywords

The following words are defined as keywords and should not be redefined in a sketch:

defined(ARDUINO)	1
defined(__AVR__)	1
defined(__AVR_ATMEGA168__)	1 for Uno or 0 for Mega
defined(NUM_ANALOG_INPUTS)	6 for Uno or 16 for Mega

FALSE	0
TRUE	1
LOW	0
HIGH	1
ARDUINO	100
TOTAL_ANALOG_PINSX	// unused - default return value of 1
TOTAL_PORTS	14
INPUT	0
OUTPUT	1
INPUT_PULLUP	2
TOTAL_PINSX	20
SD_CARD_TYPE_SD1	1
SD_CARD_TYPE_SD2	2
SD_CARD_TYPE_SD3	3
PI	3.14159265358979323846264338327 95
HALF_PI	1.57079632679489661923132169163 9
TWO_PI	6.28318530717958647692528676655 9
DEG_TO_RAD	0.017453292519943295769236907684886
RAD_TO_DEG	57.295779513082320876798154814105
NUM_ANALOG_INPUTS	6 for Uno or 16 for Mega

TCC	0
NULL	0
DAC0	66
DAC1	67
ARDUINO	100
__AVR__	1
__AVR_ATmega328P__	1 or 0
__AVR_ATMEGA168__	0
NUM_ANALOG_INPUTS	6 or 16
__AVR_ARCH__	100
__AVR_ENHANCED__	0
false	0
true	1
__GNUC__	4
__GNUC_MINOR__	3
__INT_MAX__	6
RASPBERRY_PI	0
WL_IDLE_STATUS	0
WL_CONNECTED	1
WL_NO_SHIELD	-1
MAX_SOCK_NUM	4

PINA-L (no PINI)
DDRA-L (no DDRI)
PORTA-L (no PORTI)
defined(...)

Custom Libraries

The Simulator now has the ability to load custom libraries and single step through the code. Please note that pointers are not supported and this is the typical reason for a library not to run.

To use a custom library, there are two methods. The first is to copy the header file and cpp file to the same directory as the sketch, and then the Simulator will find these and load them in separate tabs.

The second method which is the preferred method is to use the library directory (.\Libraries) and then the Simulator will automatically find the library header and source file and load them in separate tabs in the Program Window. This method is better since the Simulator will find the files in the same place as the Arduino IDE, and will save the effort of copying the files. The Library Directory can be setup by pressing F6 to load the Edit Sketch page and at the right hand side is an Edit Box which allows the Library directory to be Set or Reset.

Millis

The Millis functions returns the number of milliseconds the program has been running for. In the Shortcut toolbar, the Millis edit box shows the value of milliseconds as a floating point number. The micro function will output this value * 1000.

Please note that the Simulator is not a real-time Simulator and runs around 1000 to 10000 times slower than a real Arduino.

Troubleshooting

- Menu has disappeared => resize the screen or right click on the Arduino picture and uncheck Minimize
- The Language is wrong => Click Edit (second from left top button) and select language and change back to the selected language

- An error has come up => click the Log button to log the error to Quality Central or Email to email the error with the sketch
- I have too many errors and just want to run my sketch - right click in the Sketch Window and uncheck Errors Enabled

Virtual Serial Ports

There a big demand for using the Simulator with a Virtual Serial Port. The best virtual serial port we found is:

FREE Virtual Serial Port Driver - http://www.virtualserialport.com/ Note: creates two serial ports which is useful to send data from one port to another
(NOTE: for 14 days only and then a $99 licence fee applies)

Registry Settings

Disclaimer: Do not use a registry editor to edit the registry directly unless you have no alternative. The registry editors bypass the standard safeguards provided by administrative tools. These safeguards prevent you from entering conflicting settings or settings that are likely to degrade performance or damage your system. Editing the registry directly can have serious, unexpected consequences that can prevent the system from starting and require that you reinstall Windows .

All the Simulator settings are saved in the registry under the key
HKEY_CURRENT_USER\SOFTWARE\SimForArduino x.xx where x.xx is the version number

The Registry settings can be viewed by running the RegEdit program or a similar utility.

Version Info

Version 0.99E - Dec-2014
1. Add adjustable syntax colours
2. Move settings into separate folders in registry
3. Add timer interrupt
4. ms goes red and pressing triggers timer interrupt
5. Add 4 breakpoints
6. Allow multi char arrays
7. Fixup char arrays - convert to strings
8. Improve return function
9. Add in undock variables
10. Add in LED support using timer interrupts
11. Addin 4 breakpoint conditions
12. Improve breakpoint hidden command // breakpoint1(12,3,i=0)
13. Improve library directory finding
14. Allow arrays to be part shrunk in variables
15. Fixup line numbers to start at 1 where possible

Version 0.99D - Sept-2014
1. Improve Logic Analyzer to have continuous time base
2. Add Read and Println to Logic analyzer - improve zoom feature
3. Add in 3 more breakpoints
4. Improve Find to search through all tabs
5. Add replace to Edit Sketch window
6. Addin ten more tabs to 30 tabs maximum
7. Fixup two or more declarations on same line
8. Allow for structs within structures or unions

9. Add Option to Autosize variable area or not
10. Fixup bug with setup where index is not centred vertically
11. Add GSM functions and test
12. Add Wifi Instructions and test
13. Fixup Keyboard bug which flagged error during Unit testing
14. Check for recursive Setup call and don't crash
15. Increase maximum loop level to 300

Version 0.99C - June-2014
1. Improve Trace function to record all digital pins from 0-53
2. Improve Logic Analyzer to add timebase, pins14-53, wheel scroll zoom, mouse hover and green line on sketch line
3. Fixup Serial Monitor COM port not being saved
4. Move autostep ms to Edit page since not that relevant
5. Add reset to Set library on Edit page
6. Variable area double click now resizes, row height resizes with font
7. Add some more skins
8. fixup int i=digitalRead(x)
9. maxMin(tempC1) causing errors
10. Add email and file attach in Error message
11. fixup while in call stack
12. multi-expr lines
13. interrupt returns
14. add 10 minute startup free trial period
15. fixup redeclare PIN,PORT,DDR

v0.99 - 31-Oct 2013
1. Add Connectors
2. Add Class Explorer
3. Add Show Arduino option
4. Allow breakpiont to override previous
5. add double click to Show Subroutines
6. add many more unit tests
7. fixup rollover for variables
8. fixup array failures
9. add char s1[] = 'a','b';
10. fixup for loop issue
11. save Input/Output box heights
12. fix (unsigned long) cast
13. allow multi expression per line
14. allow two and three functions in one expression
15. fix Serial Monitor Auto Detect

v0.98 09 November 2012
1. Add Clear, Copy and Open in browser window to Input/output Ports
2. Add error message for floating point comma issue with brief instruction to fix
3. increase servos to 12
4. Add some Ethernet UDP support
5. Add Screw Terminal shield
6. Add Due
7. Add two LCD shields 16x2 and 8x2
8. Fix some else issues
9. Move A0 to 14 for Uno and 54 for Mega
10. Improve string processing - strings get saved to temporary array to stop "quote mark issues
11. Enable multi line processing
12. Fix a breakpoint issue
13. Add support for SerialIn, Breakpoint(Line,tab, condition), and SimulateLCD
14. Fixup up themes
15. Add Edit and Language in case Language is set to ?

v0.97 03 August 2012
1. Add CallStack window
2. Improve EEPROM dramatically - add save/load clear and fill
3. Add F3 Find to Sketch, variables and edit
4. Add Leonardo and Mega 75%
5. Add ON/OFF for objects (in call stack window)
6. Add inline Simulate(Uno) and Breakpoint commands
7. Add load/save language
8. fix a = b*c+d
9. add ?: and log10 operator
10. fix cos(a+b)+d
11. Add save and load hardware settings
12. Check all menu functions and fix Run F9
13. Remove back step
14. Fix break
15. Improve looping if/else/while

v0.96 22 April
1. Make Input/Output ports open as 1,2,3,4,5 so LCD, SPI and I2C can be used together
2. Start implement custom libraries - add four test samples sketchs
3. Add breakpoint condition and library directory to Edit Sketch screen
4. Add two more fonts
5. Add divide by zero check
6. Add recursive checks and max include checks
7. Stop input.output window resizing when reset is click
8. Improve minimize mode and add lights to LEDs
9. Add finger to reset button, pulse to crystal and USB plug
10. Add Structure, two dim arrays (without = yet) and strcmp
11. Add class and abject processing for up to 10 objects
12. Convert /t and .n to tab and linefeeds (tabs do not quite line up yet)
13. Add PIN, PORT and DDR processing
14 Add many more keywords
15. Remove Option Delay and Tone

v0.95 27th Feb
1. Run/Stop button changes state when Abort is pressed or error is found
2. Add syntax colouring where possible
3. Improved the Uno/Mega selection - Uno picture will only shows pins 0-13
4. Added a font selection for the program listbox
5. BUG - Fixed reset issue with Line Numbers on - caused by attempted parse of commented line
6. String addition not recognised - "This" + "That" prints as is
7. LCD Scroll goes past characters allowed
8. Match Brace option Added
9. Add Windows Themes - still to test if it works on a separate PC
10. Improve variable area to have individual +- for folding arrays
11. Improve error processing so sketch errors are different from possible Simulator errors
12. if/While loops need to check for commented lines
13. Include file directory sometimes loses the directory (possible issue)
14. Increase digital pins to 53
15. Add a button area to the reset switch and crystal

v0.94 28th January 2012
v0.93 6th January 2012
v0.92 - 30 December 2011
v0.91 - 16 December 2011
v0.80 - v0.90 1-Dec to 16-Dec-11 first test releases

Credits

This project has been a collaboration with many people.

Credit and thanks must go to these people for all their help: Adrian Wells, Marco Stuurman, Zeljko Frankovic, Pete Lunt, Mark Grass Sr, Serge Desjardins, Peter Brouggy, Halam Rose, Larry Vatland, Raimondas Butauskas, Jason Snow, J-Pierre Romanogli, Hamilton Elliott, Graeme Caie, Michael Moore, Filipe Oliveira, Leon Rozengarten, Robert Lopez,Mauro Abbattista,Todd Radack,Alain Herben, George Vrynios, Neko San, Mauro Abbattista, Alain Herben, Victor Aguilar, David Williams, Janis Gkatzaras, Rodrigo Amaya, Ed Ross, David Cox, Shane Stenton, Freddie Snijman, Andres j. Ogayar, Anxionnaz Yannick, Jeandaniel Planterose, Donald Dempsey, Enrique Condes, Peter Brown, Mladen Bruck, Jesse Carneiro, Nigel Woodford, Ken Jensen, Koen Victor,Damir Kudeljan,Maxwell Yun, Douglas Hendricks, Marco Baitelli, Howard Bassen, David Griffey, Patrick Beier, Fokko Dusseljee and few hundred thousand more we will add when there is some spare time.