

The aim of this short document is to get your first program to run on the EZSBC1.

## Conventions

Programs are displayed in this font:

```
REPEAT
  OUTD 41,0
  WAIT ontime
  OUTD 41,1
  WAIT offtime
UNTIL INKEY > 0
```

Keystrokes and Keys to press are shown as *Ctrl-W* and it should be understood to mean press the Ctrl key followed by the W key and letting go of the keys in the reverse order.

### ***For the Impatient***

- Install FTDI USB Virtual Comm port drivers.
- Install TeraTerm or another terminal emulator.
- Configure the terminal emulator for VT100 emulation
- Set the terminal for 57600,8N1 communication
- Connect the USB cable and find the new serial port
- Connect the terminal emulator to the new serial port
- Press any key
- Skip ahead to the **Enter the Program** section

## Concepts

To distinguish between different parts of the controller we need to use the same terminology. EZSBC1 refers to the entire module, all the components on the PC board, the pins. Control BASIC or sometimes just BASIC refers to the language used to program the EZSBC1. It is likely to be slightly different from other BASIC dialects that you may be familiar with. The details of Control BASIC and its keywords and reserved words are described the Control BASIC Language Reference. All programs executed on the EZSBC1 are written in Control BASIC.

Input and Output can be very confusing because the direction depends on the viewpoint of the author or reader. In this document direction is described from the view of the EZSBC1. Pretend you are sitting on the EZSBC1 then IN or Input is towards you and OUT or Output is away from you (and the EZSBC1). When a signal is described as an Output then the EZSBC1 will drive the pin in some way. It will source some current and try to force a voltage on the pin. The EZSBC1 is in control of the state of the pin.

When a signal or a pin is described as an Input the EZSBC1 will try to 'read' or 'evaluate' the pin. In most cases the pin will approximate an open circuit, the EZSBC1 will draw little or no current from the pin or signal. Something other than the EZSBC1 should drive the pin or control the voltage on the pin.

When a BASIC program is not running on the EZSBC1 then it is waiting for user input. A program is still running on the EZSBC1 and it is referred to as the EZmon. It is a "monitor program", a program which is loaded onto a Single Board Computer (SBC) at the time of manufacture. It provides a few features and controls over the SBC. It provides a functionality that is always there and can be relied upon. It makes it easier to develop

software or programs on the Single Board Computer. Later in this document the features of EZmon will be described in some detail in the **Operation** section. Output to and from EZmon occurs via USB that emulates a serial port and will appear on the PC as a serial port. Before we can connect the EZSBC1 to the PC we need to get some drivers installed and some software configured.

## FTDI Driver Installation

The EZSBC1 requires USB drivers for Windows and Apple PC's. On Linux systems it is recognized directly by built in drivers.

Point your favorite web browser to this page: <http://www.ftdichip.com/FTDrivers.htm>. Towards the bottom of the page you will find a link that says 'VCP or D2XX)'. Click on the VCP link to go to the Virtual COM port drivers. You should end up on this page: <http://www.ftdichip.com/Drivers/VCP.htm>. In the very right hand 'Comments' for the Windows drivers there is a note "Available as [setup executable](#)". If you are using Windows then download this file, it is the easiest to use. When I downloaded it the file name was CDM20824\_Setup.exe. Run this downloaded program to install the drivers before connecting the EZSBC1 to the computer. You should get this window (or something similar).



Press the 'Run' button to continue. Some windows may pop up depending on the exact version of Windows being used. On Windows XP a window briefly appears and closes. The drivers are now installed.

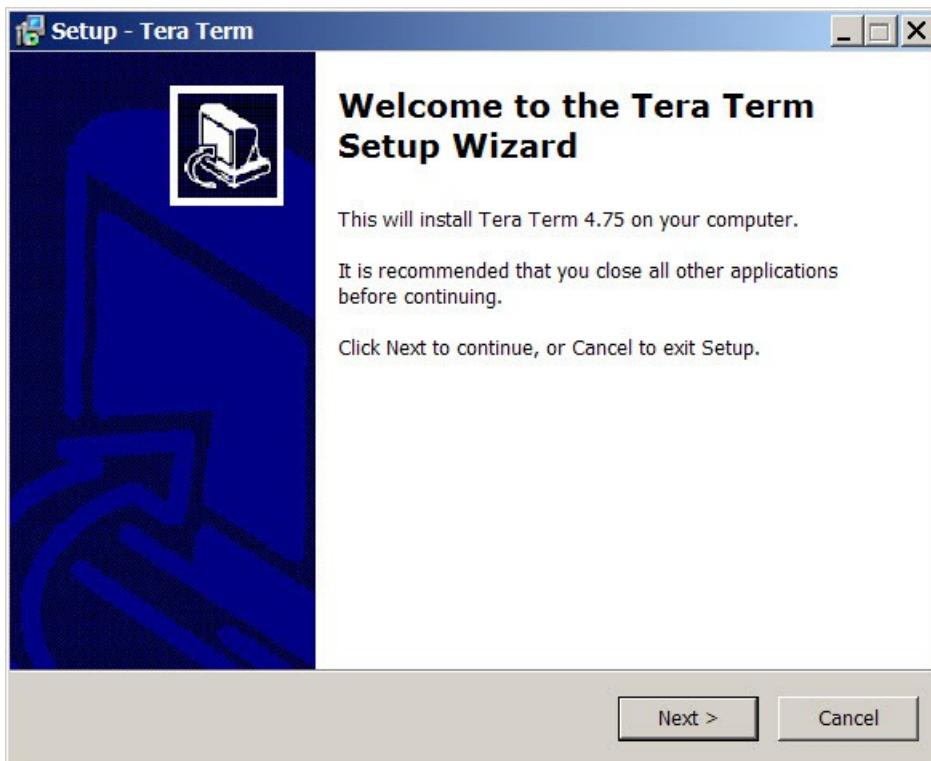
## Tera Term Installation

On a Windows PC, point your favorite web browser here: <http://en.sourceforge.jp/projects/ttssh2/>. There is normally a big green button on the page with the word 'Download' in it. Press this button and a new page should open with a [teraterm-4.75.exe](#) and [teraterm-4.75.zip](#) link on it. (The version numbers may be higher.) Download the teraterm-x.xx.exe file to your local machine. Double click the executable to install Tera Term.

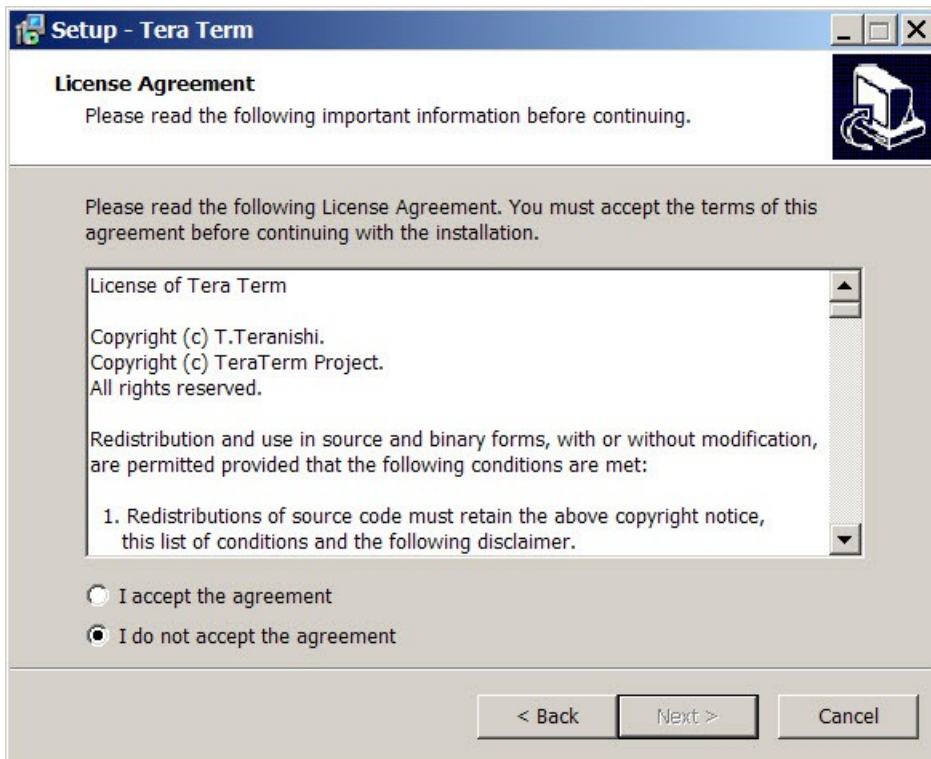
You should get the normal Windows installation dialog box as shown below.



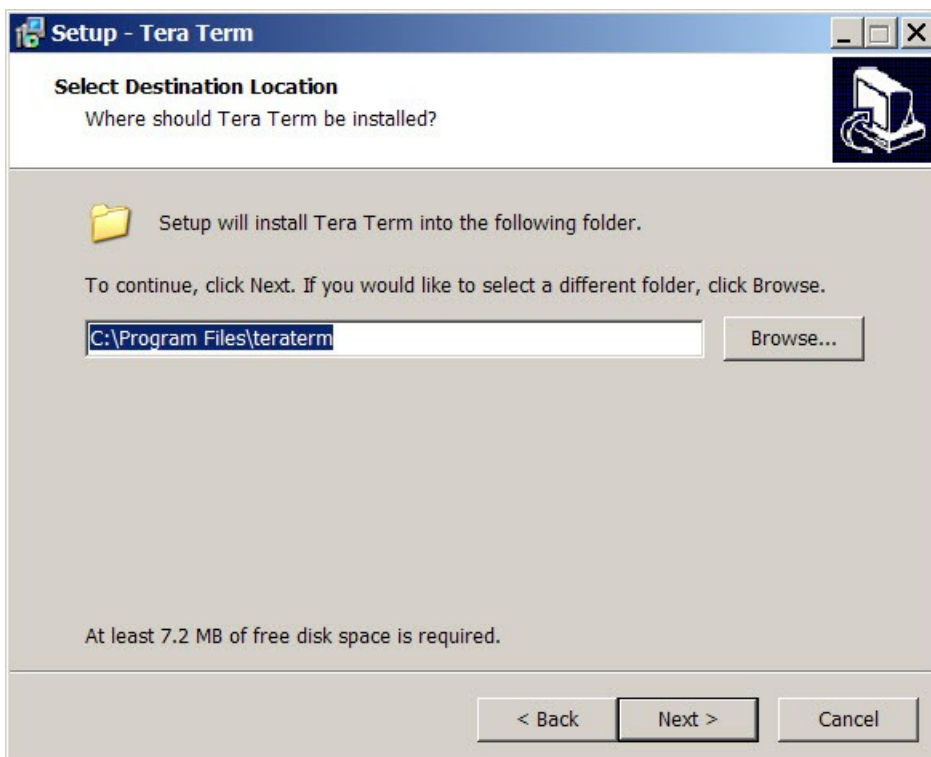
Click on the 'Run' button to Continue. A new window will pop up



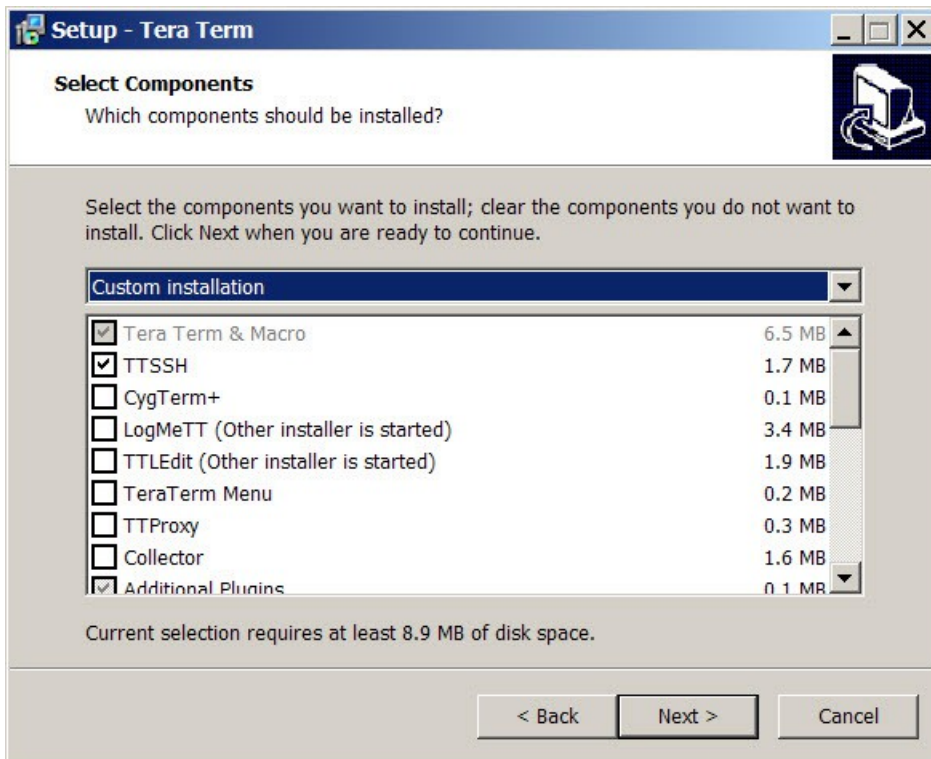
Click on the 'Next' button to continue and you should see the License Agreement appear.



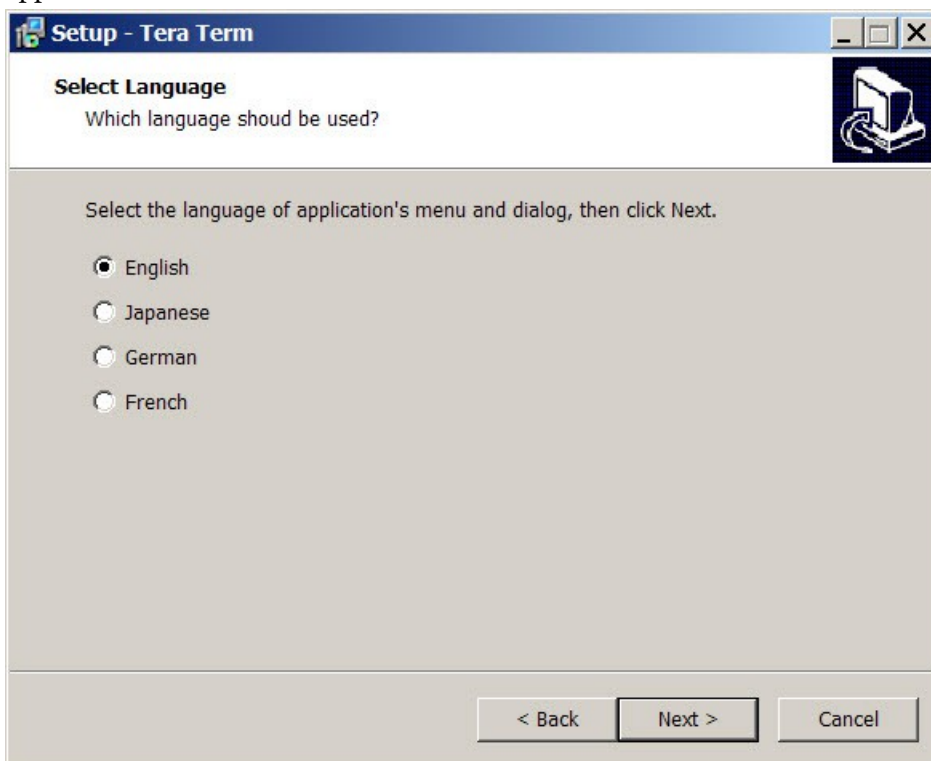
Select the 'I accept the agreement' radio button to make the 'Next' button appear. Press the 'Next >' button and the Destination dialog should appear.



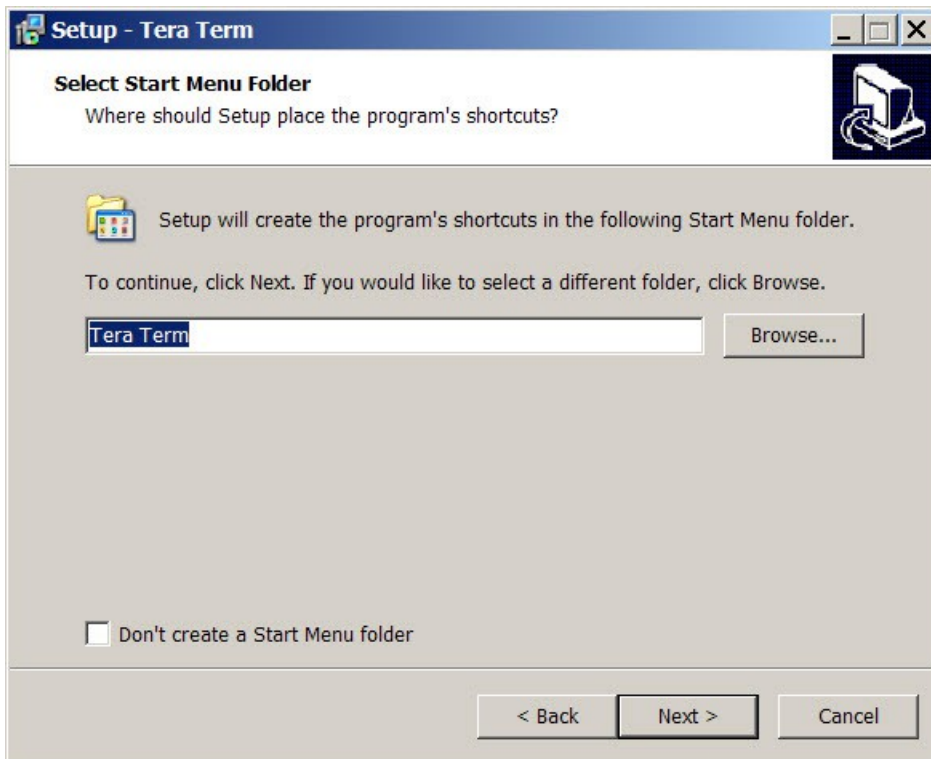
Click on the 'Next' button to install at the default location and the File Association dialog should appear.



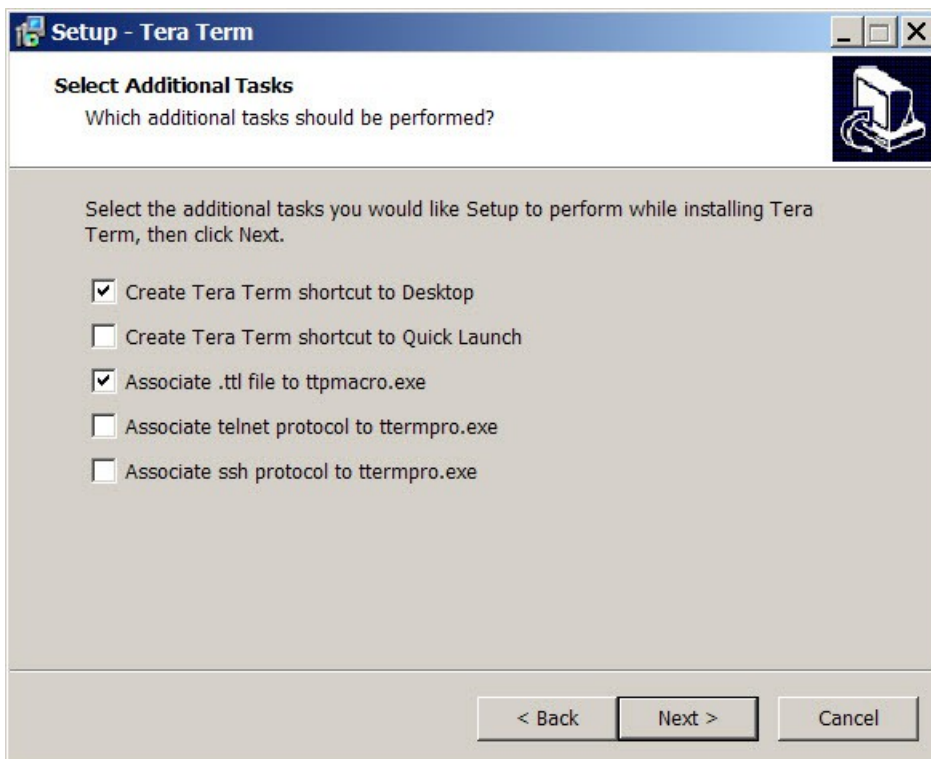
Leave the default options unchanged and press the 'Next >' button. The Language Selection dialog should appear.



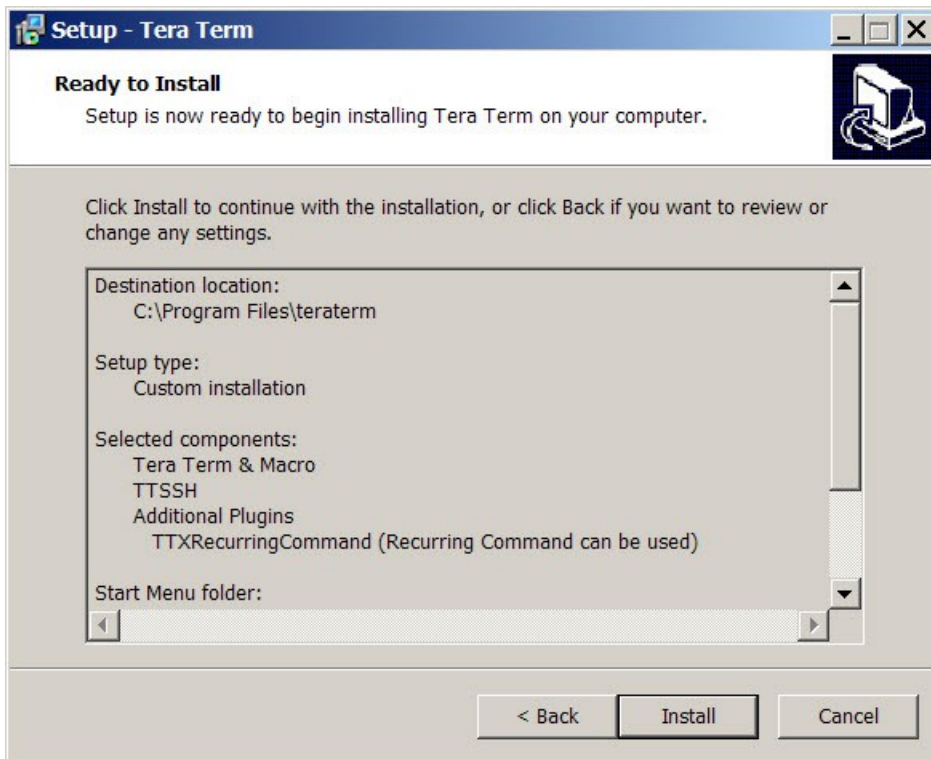
You can select any option except Japanese with English being the preferred selection to follow the documentation. Press the 'Next >' button and the Start menu folder should appear.



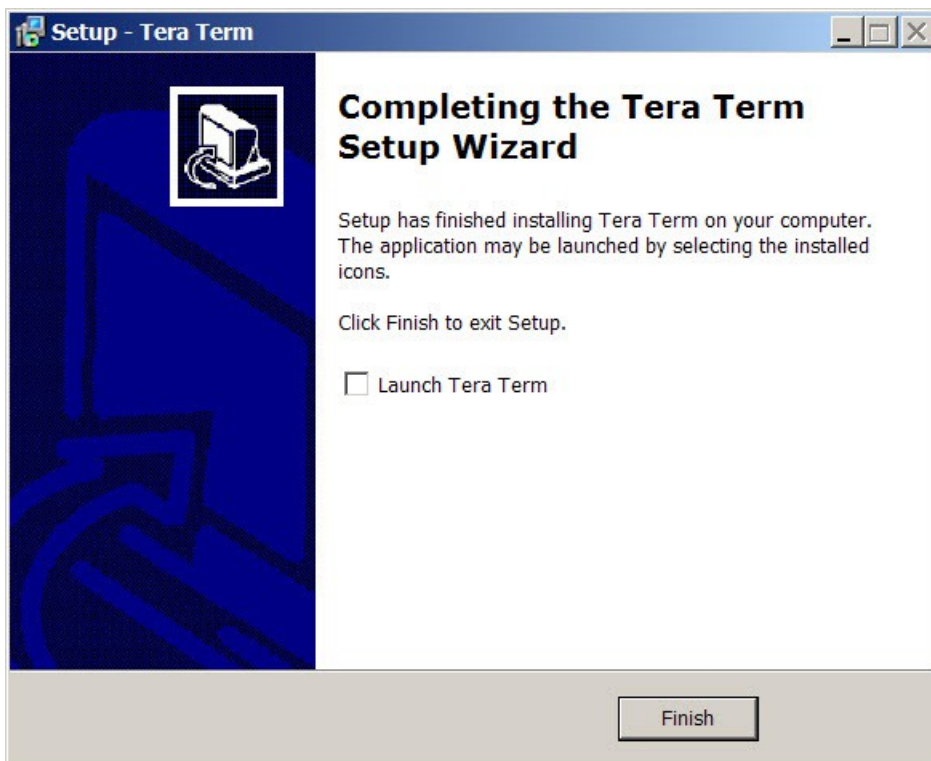
Leave the default unchanged and press the 'Next >' button. The Additional Tasks dialog should appear.



Press the 'Next >' button and the Ready to Install dialog should appear.

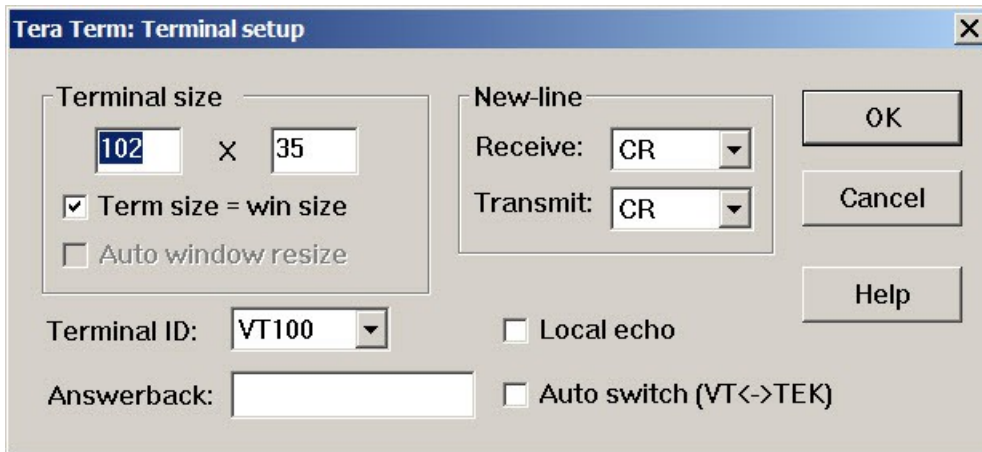


Press the 'Install' button to complete the installation.

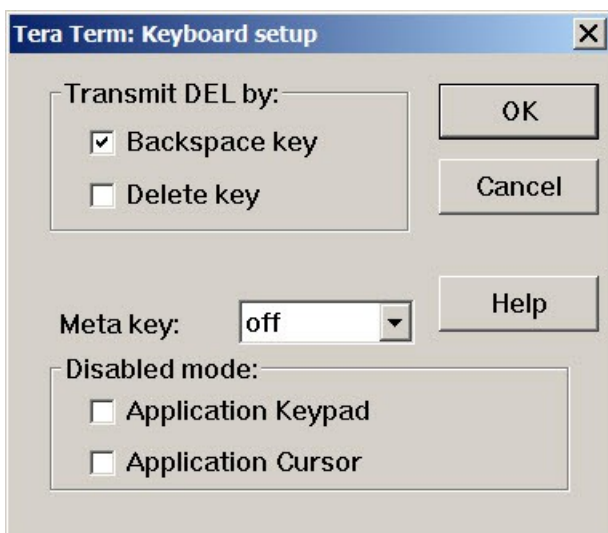


Select the 'Launch TeraTerm' tickbox and press the 'Finish' button. Tera Term should start up and display a blank screen or a pop-up with a warning message about a serial port that it unavailable. Dismiss the message box if it appears.

On the Setup Menu select the Terminal option and set the dialog box as in the screen capture shown below.

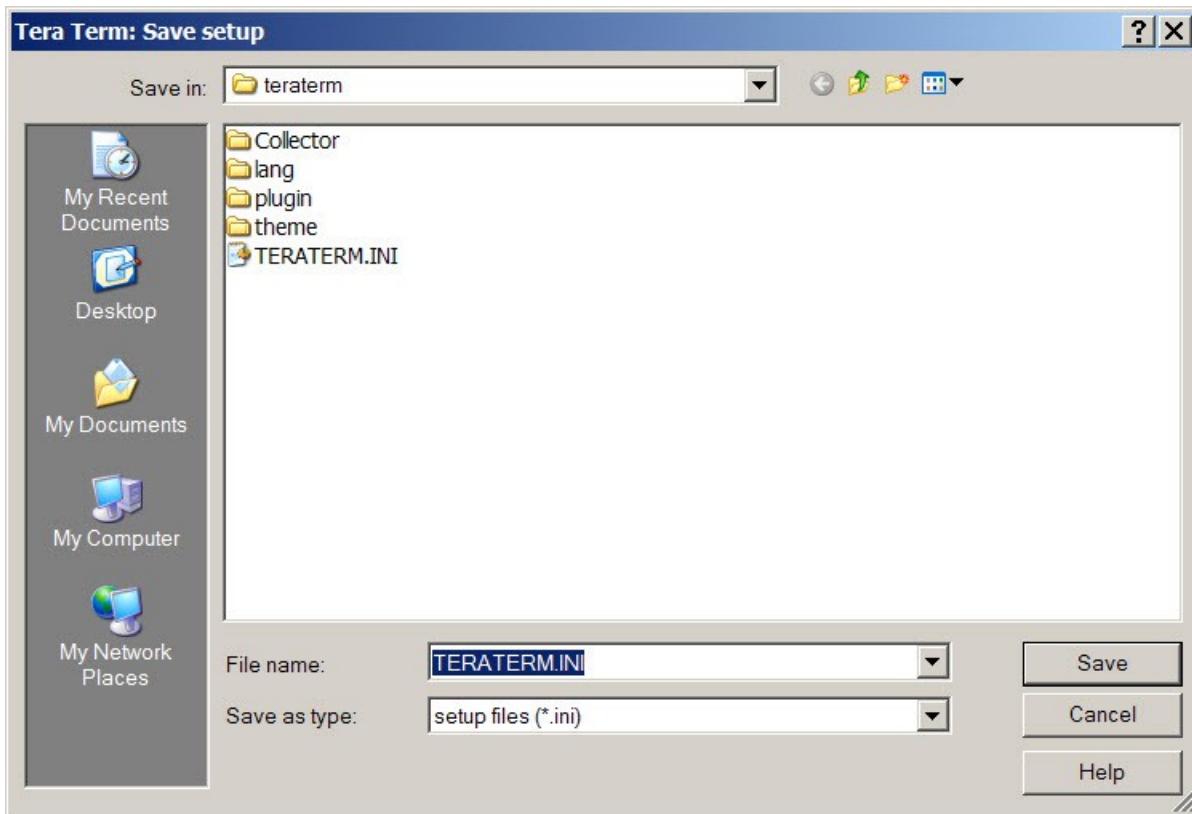


On the Setup Menu select the Keyboard option and set the dialog box as follows and press OK.

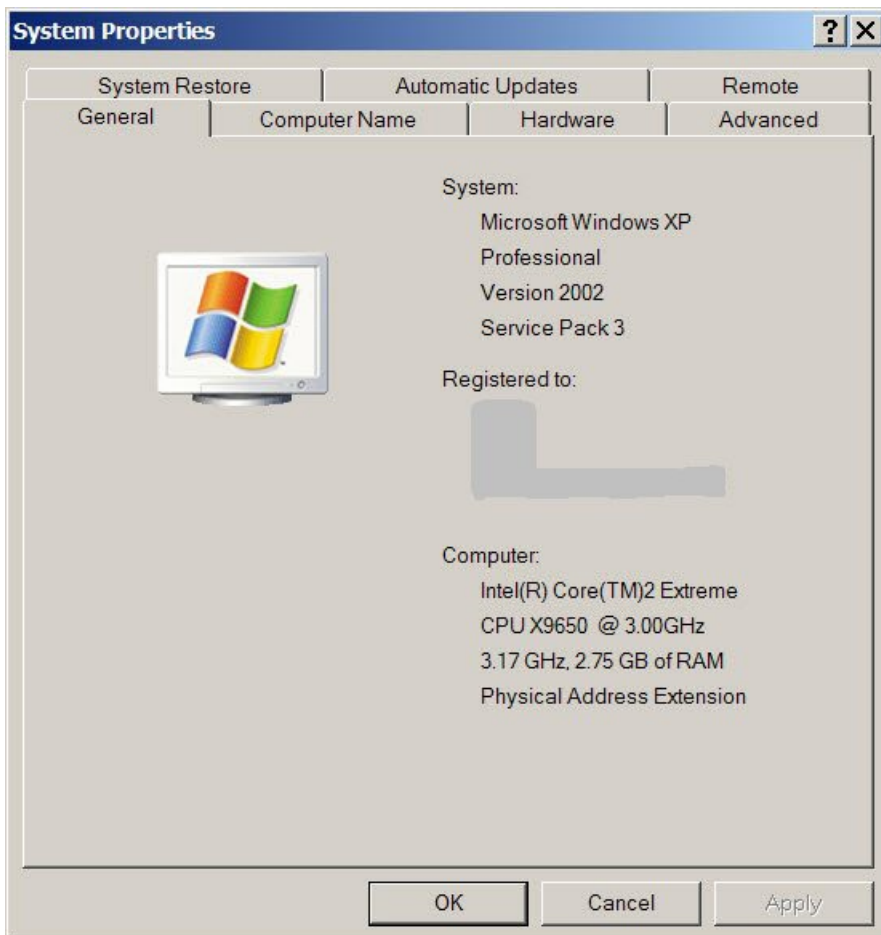


On the Setup Menu select the Save Setup option and save the setup as the default as follows

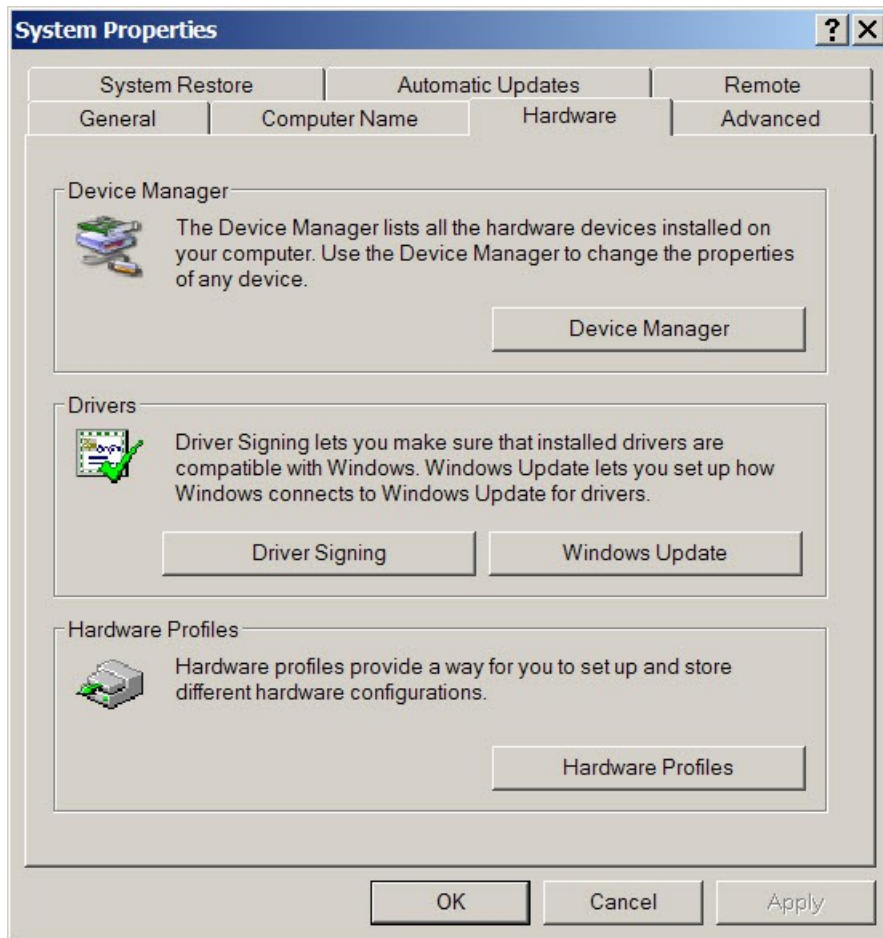




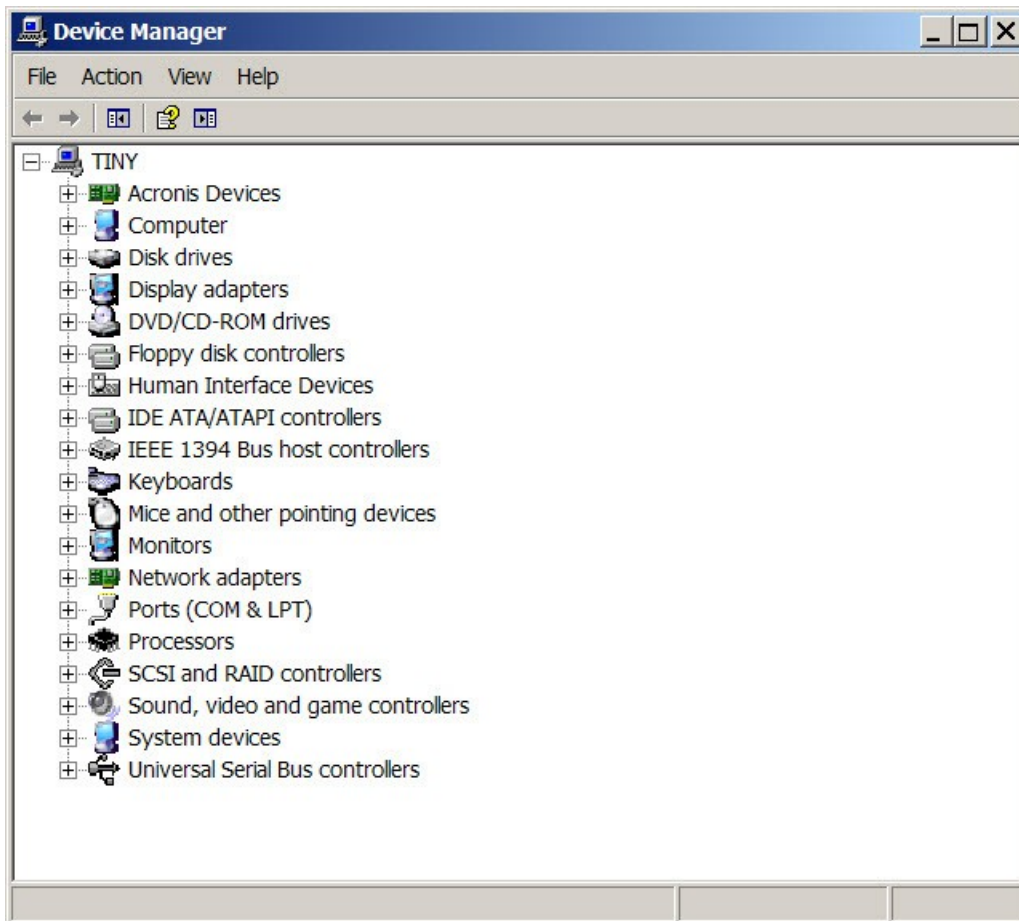
Open the System (Properties) dialog from the Control Panel



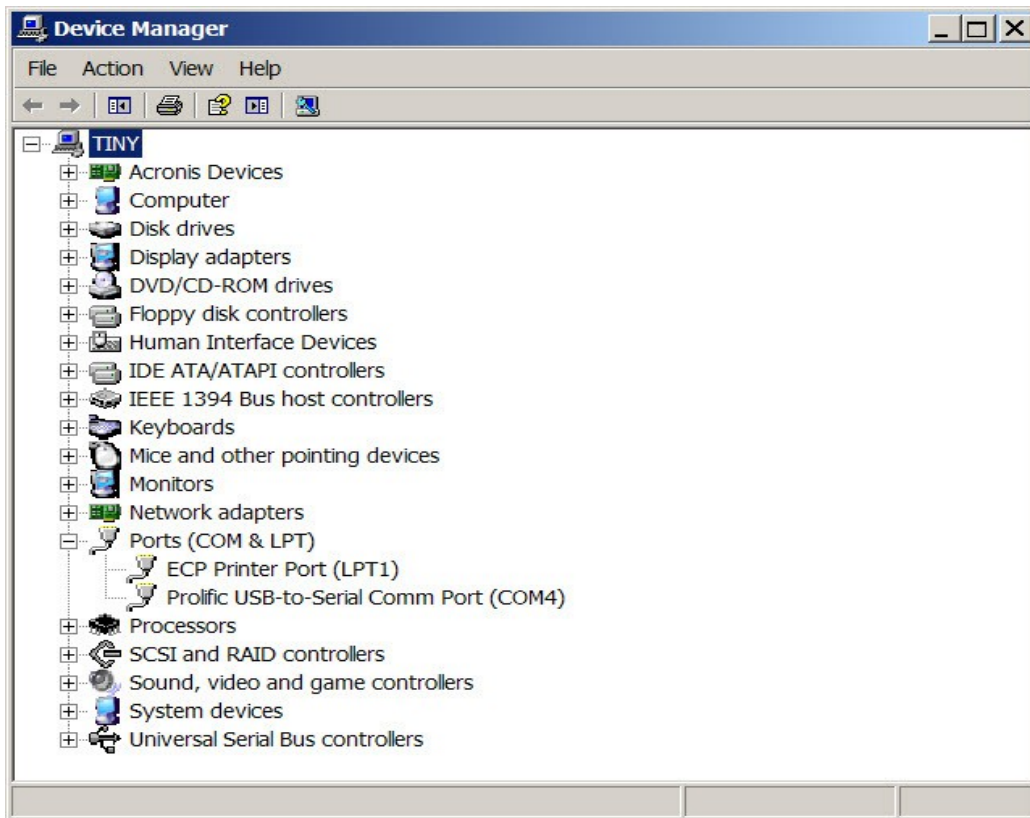
Click on the Hardware Tab



and now open the Device Manager



and expand the Ports section. You should see something like this (or it may be empty). Make note of the Serial Port COM numbers that exist. On the screen capture it is COM4.

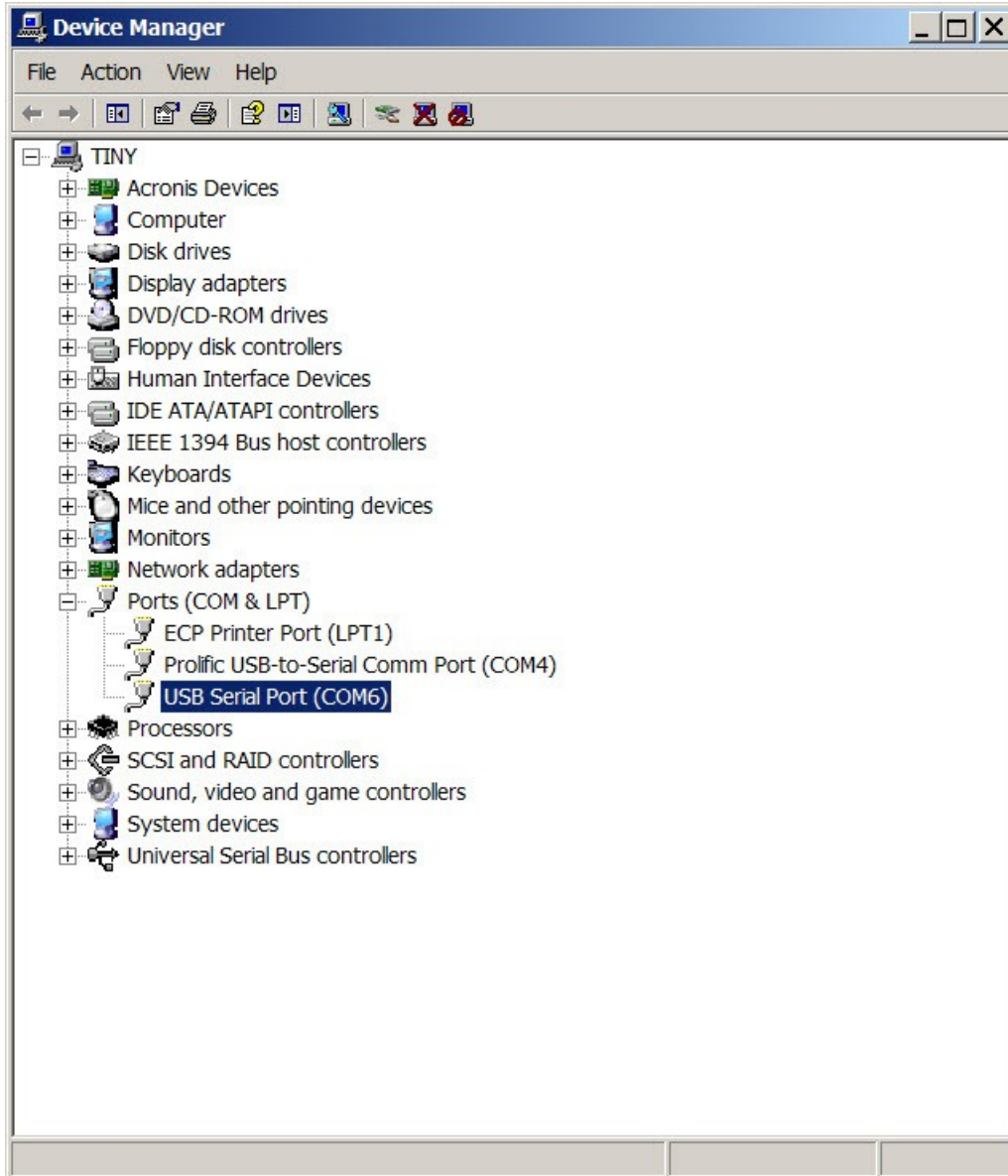


Now watch the Device Manager screen and plug the EZSBC1 into the USB cable. there will be some activity on the Device Manager and after a few seconds it should stabilize like the screen capture below and you may be prompted to restart Windows. If the yellow tick mark is present, save any unsaved work and restart the computer. If the tick mark is not present, skip over the next instruction to restart the machine.



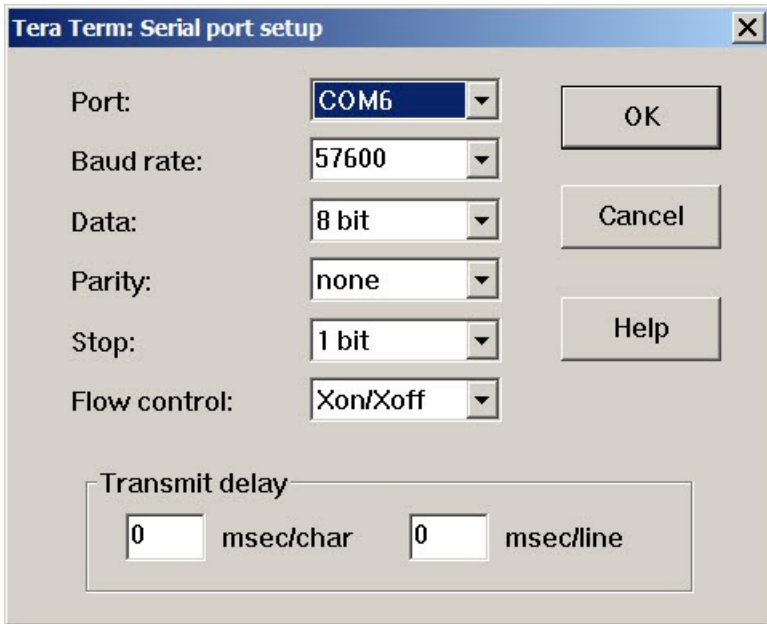


After the restart, go back to the Device Manager screen and expand the Ports tab again.

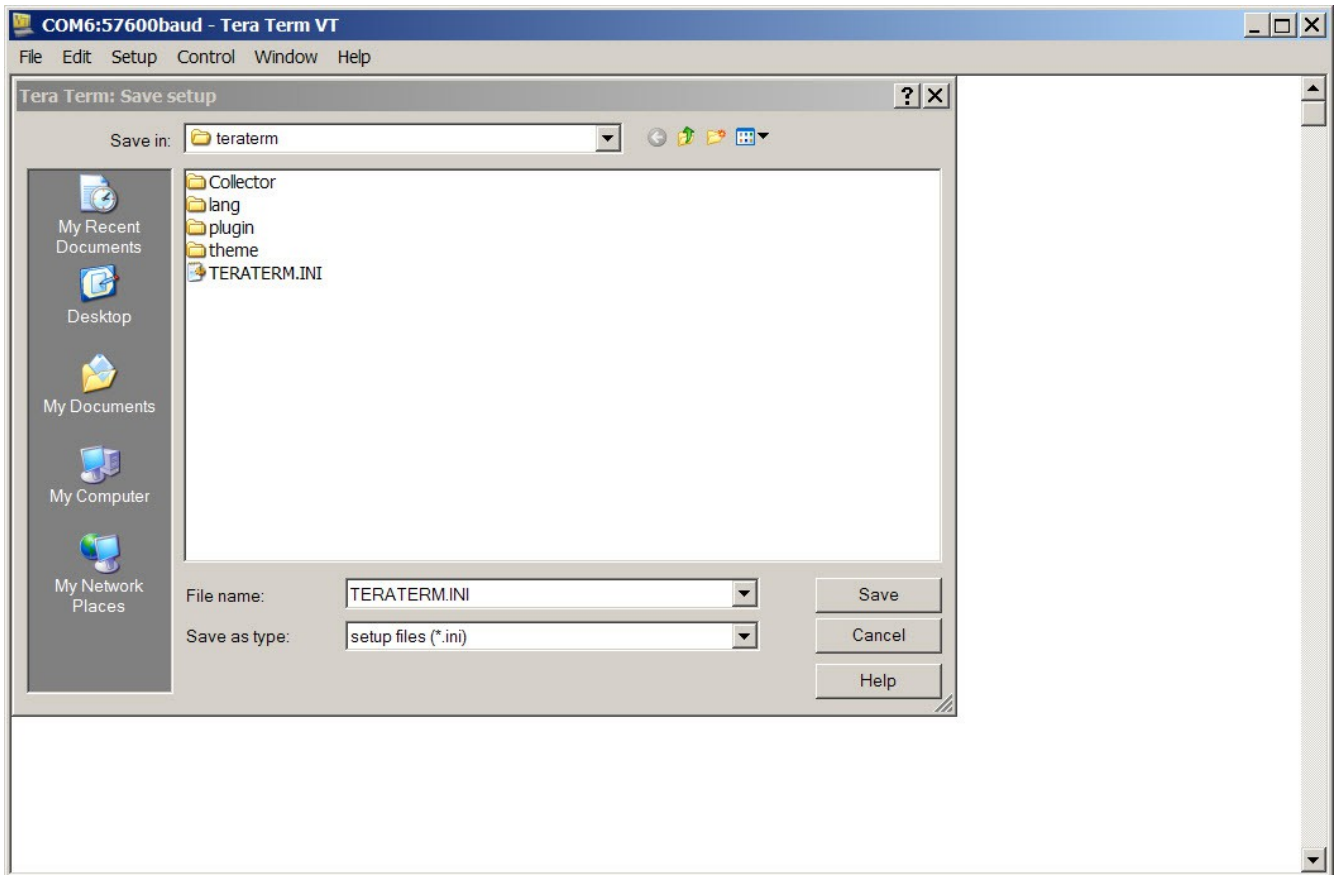


There should now be a new Serial Port with a COM number, lets say COM6. (If you get a very high COM number such as COM55, don't be alarmed everything will still work. See Appendix A for a method to set the COM number down to some low number.)

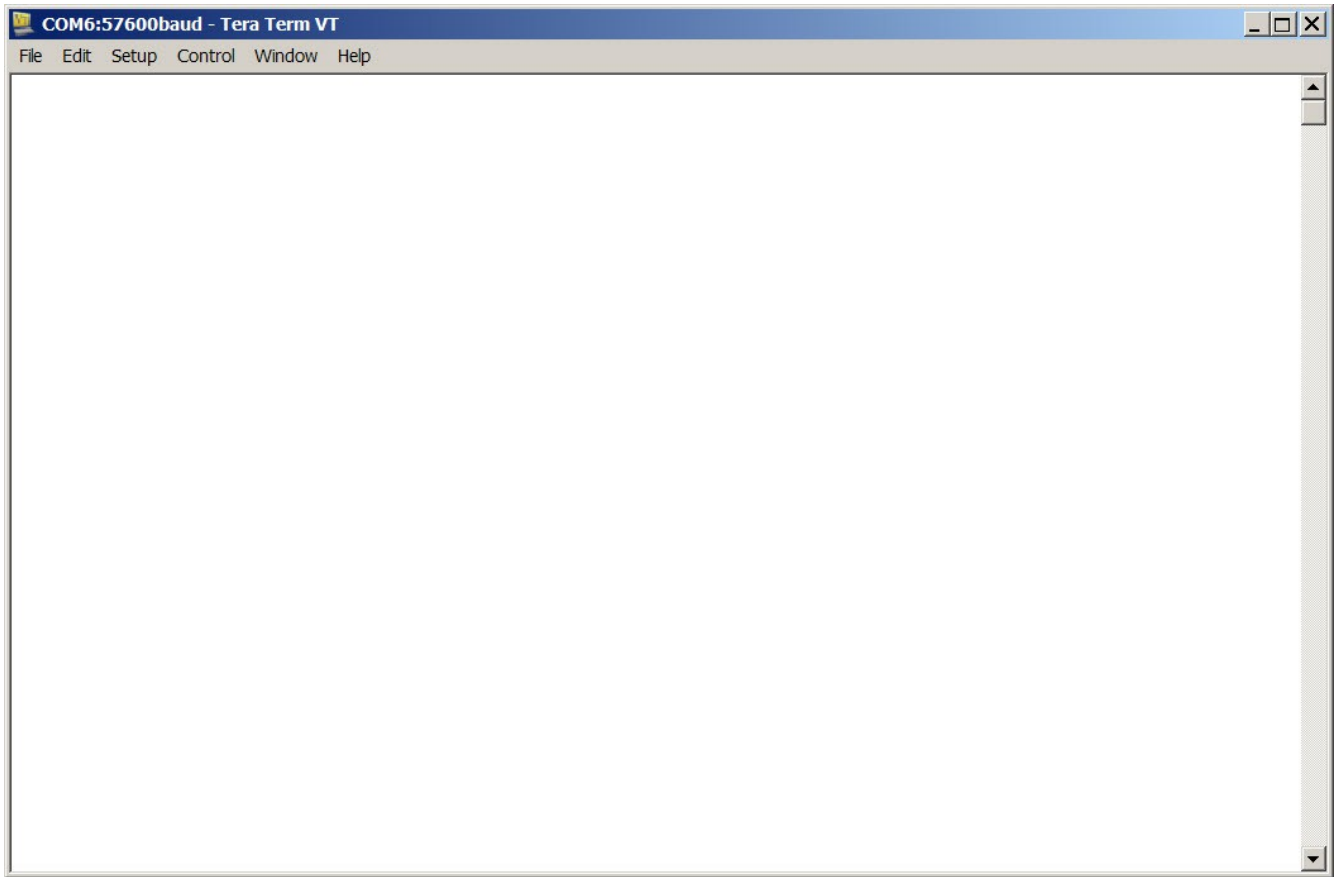
Open TeraTerm and on the Setup menu select Serial Port. On the dialog box select the correct COM port for example COM6 and set the other parameters as follows:



On the Setup Menu select 'Save setup...' and leave the defaults as they appear. Press the 'Save' button and the settings will now be restored when Tera Term opens again.



The configuration of TeraTerm is now complete. Close any open dialog boxes and you should see this:

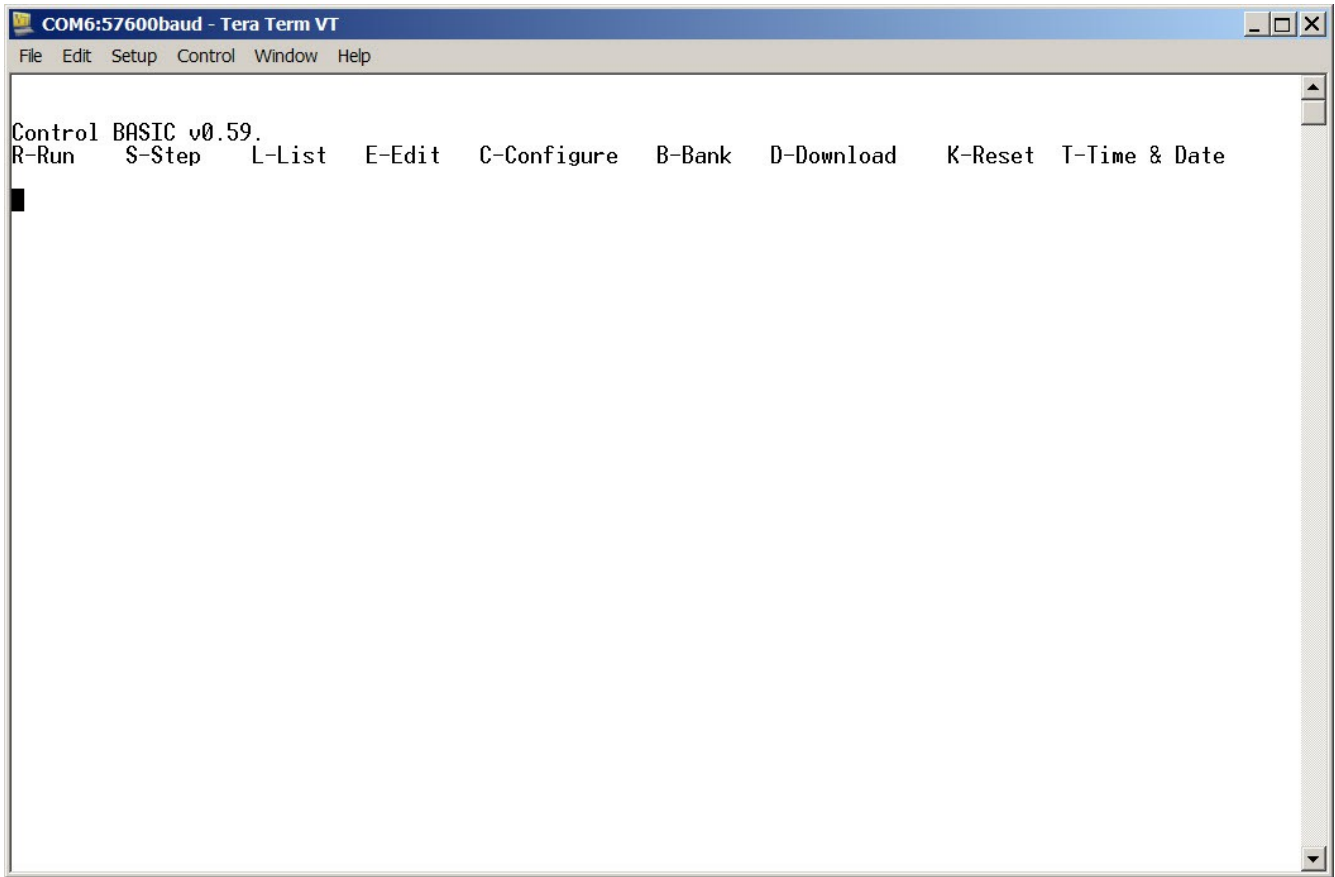


Note the title bar shows the Baud rate, com port and VT. Congratulations, you have correctly configured TeraTerm and the FTDI drivers and the EZSBC is ready for use.

On Linux, use minicom configured for VT100 emulation.

Press any key on the keyboard and you should see EZmon respond.





Press the Reset button on the EZSBC and you should see

```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help

Control BASIC v0.59.
R-Run  S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date
Cold Start

Control BASIC v0.59.
R-Run  S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date
█
```

Note the 'Cold Start' text. This indicates that the EZSBC1 was reset with the Reset button (or the K command from the terminal).

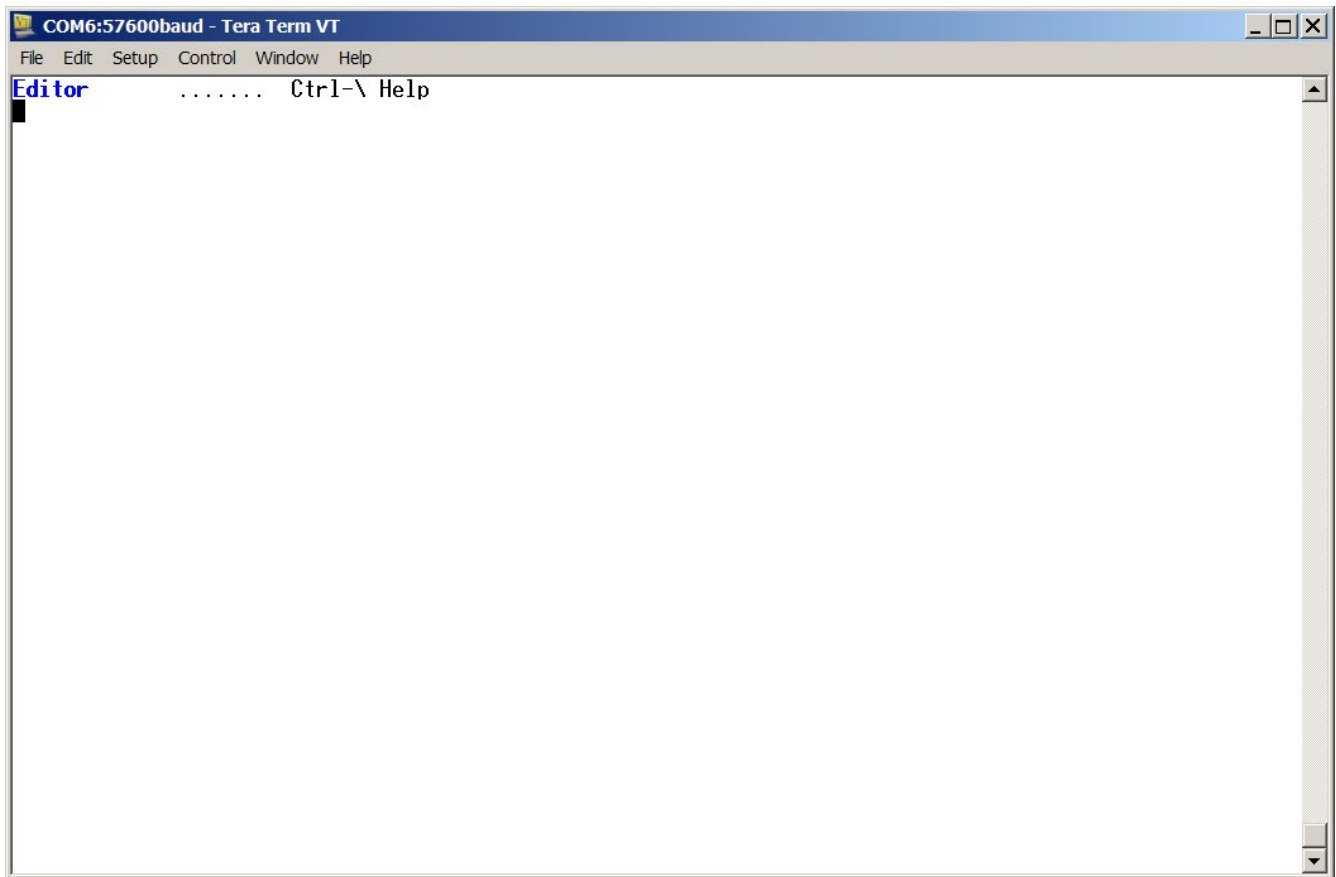
You are now ready to enter and run your first program.

## Enter a Program

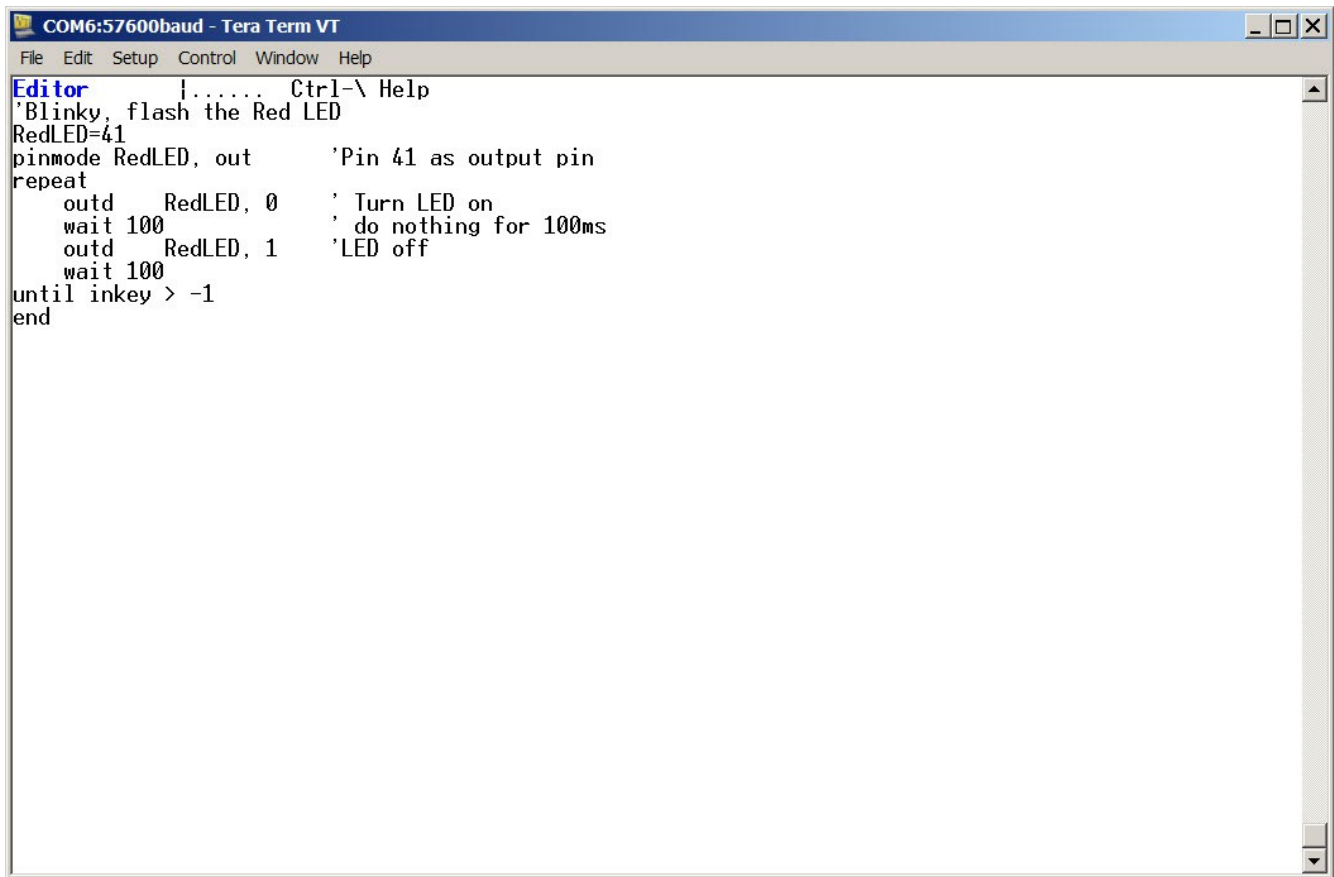
### ***Blink A LED***

Traditionally blinking an LED is the first program on an embedded controller, much like 'Hello World.' in a desktop programming language.

Type 'e' to enter the editor and type the code listed below. If the screen does not appear as in the screen capture below then another program has been loaded earlier. Skip to the section on 'Deleting an Entire Program' and follow the instructions before returning here.



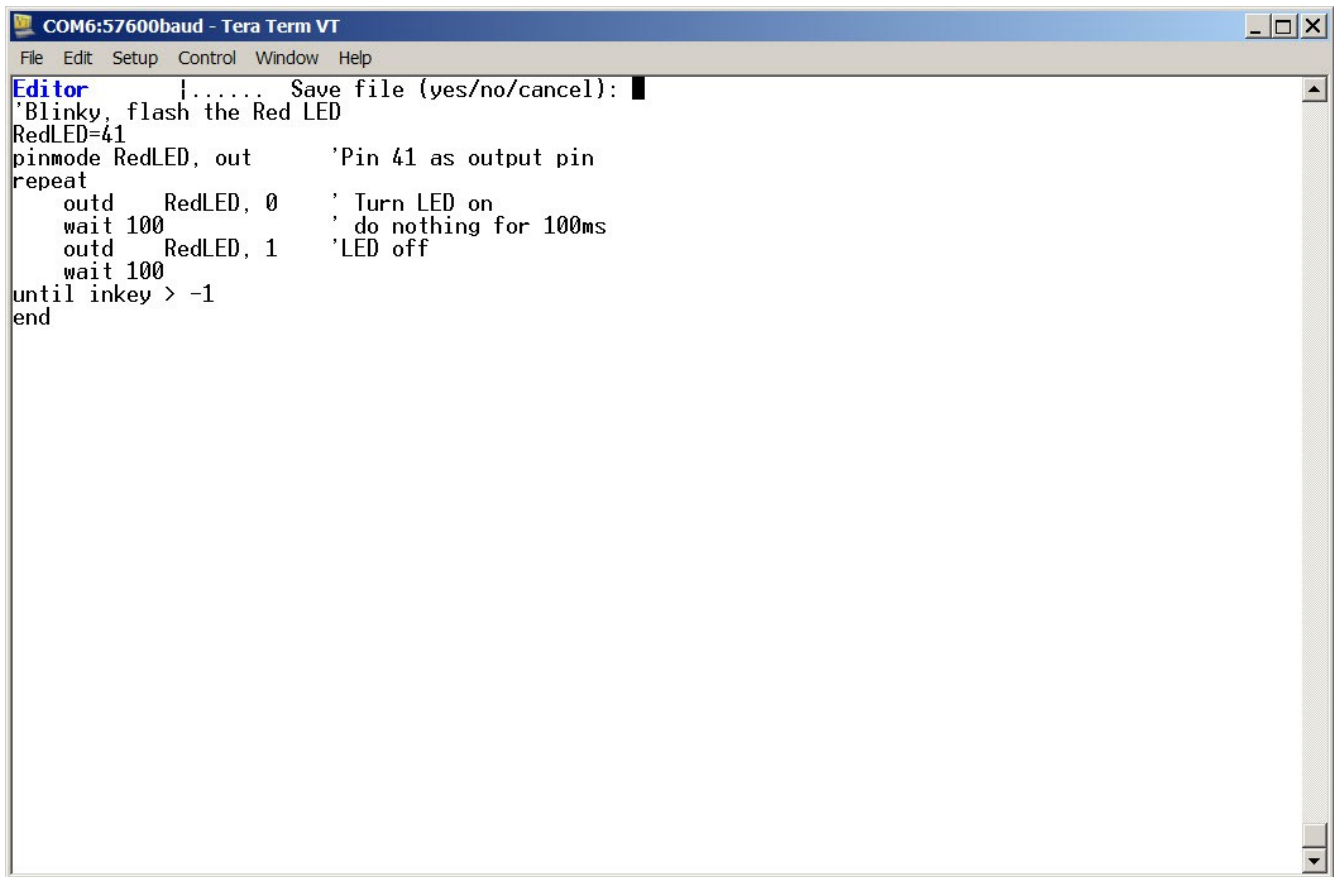
```
'Blinky, flash the Red LED
RedLED=41
pinmode RedLED, out      'Pin 41 as output pin
repeat
  outd   RedLED, 0      ' Turn LED on
  wait 100              ' do nothing for 100ms
  outd   RedLED, 1      'LED off
  wait 100
until inkey > -1
end
```



The screenshot shows a window titled "COM6:57600baud - Tera Term VT". The window has a menu bar with "File", "Edit", "Setup", "Control", "Window", and "Help". The main area contains the following code:

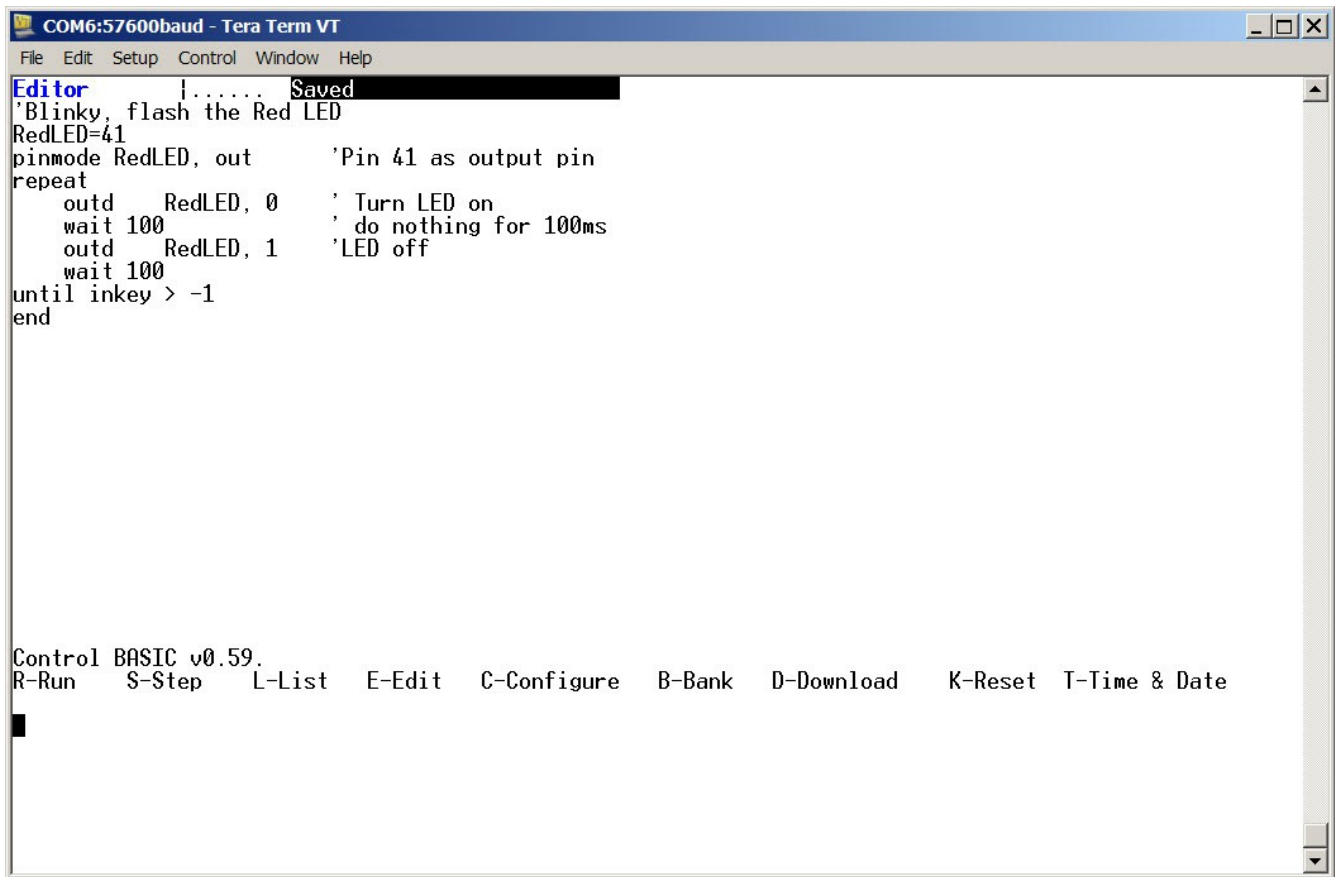
```
Editor |..... Ctrl-\ Help
'Blinky, flash the Red LED
RedLED=41
pinmode RedLED, out      'Pin 41 as output pin
repeat
  outd RedLED, 0        ' Turn LED on
  wait 100              ' do nothing for 100ms
  outd RedLED, 1        ' LED off
  wait 100
until inkey > -1
end
```

When the screen looks like the screen capture shown above press *Ctrl-W* and note that the top of the screen has changed as in the screen capture below.



```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help
Editor |..... Save file (yes/no/cancel): █
'Blinky, flash the Red LED
RedLED=41
pinmode RedLED, out      'Pin 41 as output pin
repeat
  outd   RedLED, 0      ' Turn LED on
  wait 100              ' do nothing for 100ms
  outd   RedLED, 1      'LED off
  wait 100
until inkey > -1
end
```

The 'Save file (yes/no/cancel):' message is asking for permission to write the program into the memory on the EZSBC1. Press *y* to save the program. If you press *n* the editor will quit without saving anything and *c* will not save the program but you will remain in the editor. This is what you should see:



The screenshot shows a Tera Term VT window titled "COM6:57600baud - Tera Term VT". The window contains a BASIC program for blinking an LED. The program text is as follows:

```
Editor |..... Saved
'Blinky, flash the Red LED
RedLED=41
pinmode RedLED, out      'Pin 41 as output pin
repeat
  outd RedLED, 0        ' Turn LED on
  wait 100              ' do nothing for 100ms
  outd RedLED, 1        ' LED off
  wait 100
until inkey > -1
end
```

At the bottom of the window, the following text is displayed:

```
Control BASIC v0.59.
R-Run   S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date
```

A cursor is visible on the line following the menu options.

Now press l and EZmon will list the program to the screen. It should look like this:

```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help

Control BASIC v0.59.
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date

Program Listing:
'Blinky, flash the Red LED
RedLED=41
PINMODE RedLED, OUT 'Pin 41 as output pin
REPEAT
  OUTD RedLED, 0 ' Turn LED on
  WAIT 100 ' do nothing for 100ms
  OUTD RedLED, 1 'LED off
  WAIT 100
UNTIL INKEY > -1
END

End of Listing

Control BASIC v0.59.
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date
```

See how many words are now in upper case. The interpreter (actually the tokenizer) has recognized these words as valid Control BASIC reserved words and they will now be displayed in upper case. The variables were not changed, they are exactly as you typed them. If a variable changed case to upper case then you accidentally chose a variable name that is the same as a reserved word of Control BASIC. This is an error check of your program and it is recommended that variable names not be typed in all upper case letters. In fact, typing in lower case allows many errors to be found by listing the program and paying attention to the capitalization of the text on the screen.

We are now ready to run our first program. Press *r* to run the program and see the red LED flash on and off 5 times per second.

When the program is running you will see 'Start program' displayed on the terminal screen (unless the program cleared the screen). This is an indication that the program has started and is running. If the program encounters an error then it will stop running and display an error message on the terminal.

Press any key on the keyboard to end the program.

The terminal display should like like this:

```

COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help

Control BASIC v0.59.
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date

Program Listing:
'Blinky, flash the Red LED
RedLED=41
PINMODE RedLED, OUT 'Pin 41 as output pin
REPEAT
  OUTD RedLED, 0 ' Turn LED on
  WAIT 100 ' do nothing for 100ms
  OUTD RedLED, 1 'LED off
  WAIT 100
UNTIL INKEY > -1
END

End of Listing

Control BASIC v0.59.
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date

Start program

```

List the program again. Note that there are no 'GOTO' commands or line numbers in the program. The version of BASIC used on the EZSBC does not require line numbers or the use of GOTO commands. The GOTO command is supported but should be used sparingly. The target of GOTO or GOSUB commands are labels that can appear at the beginning of any line or on lines by themselves.

The first line 'Blinky, flash the Red LED is a comment. Comments start with ' and end at the end of the line. The next line RedLED=41 assigns the value 41 to a variable called RedLED. Pin 41 on the EZSBC1 is a phantom pin to give easy access to the onboard red LED. Three more phantom pins exist; 42, 43 and 44 for controlling the yellow, green and blue LEDs. Setting these pins to outputs and low turns on the LED associated with the phantom pin.

The line pinmode RedLED, out sets the phantom pin as an output pins, as the comment suggests. The lines between the repeat and until inkey > -1 are executed repeatedly until a key is pressed on the terminal emulator keyboard. The keyword INKEY returns the ASCII value of a key and -1 if a key has not been pressed. OUTD is the digital output command of Control BASIC. IND is the input command. The WAIT 100 line causes the program to do nothing for 100 milliseconds.

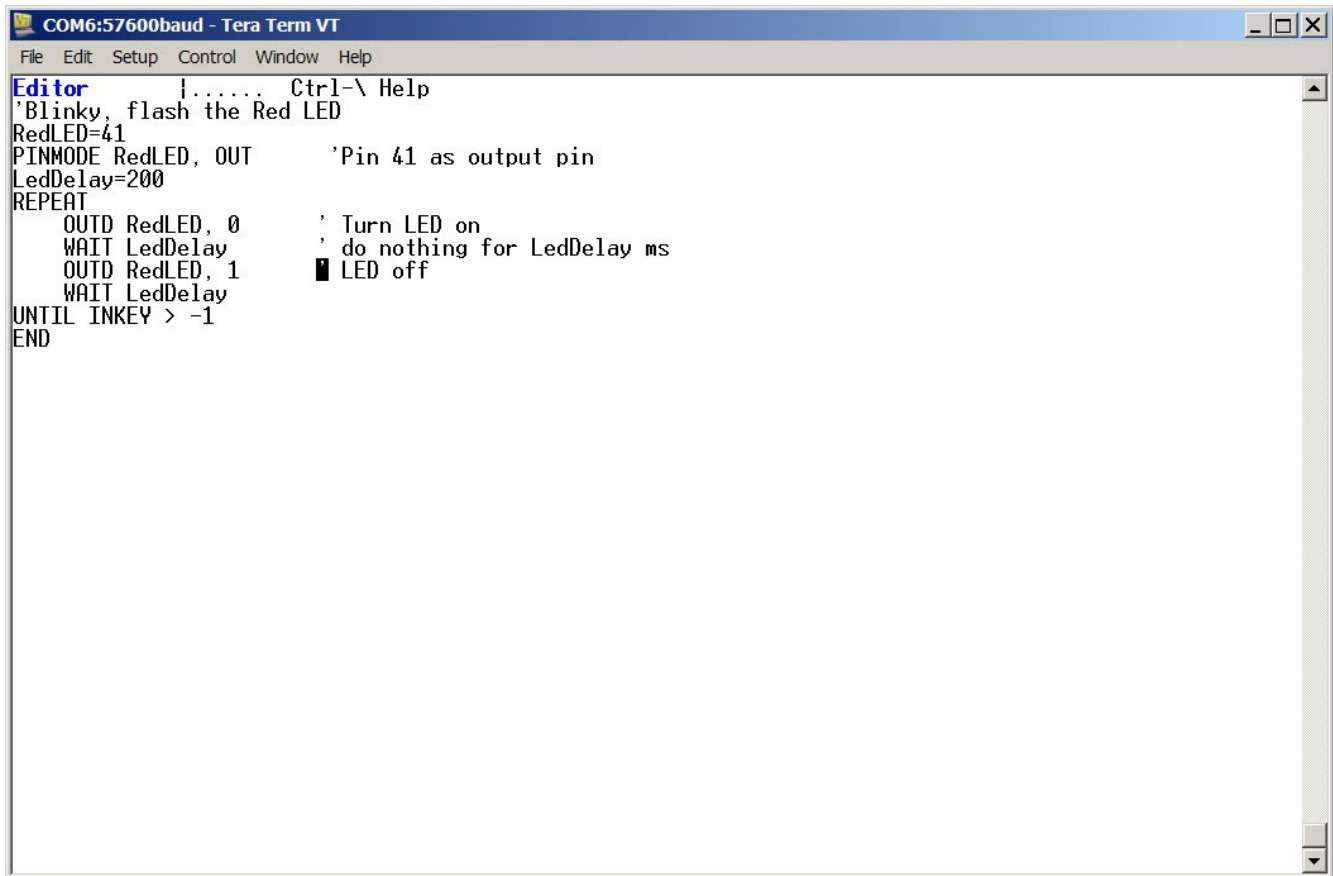
```

repeat
  outd RedLED, 0 ' Turn LED on
  wait 100 ' do nothing for 100ms
  outd RedLED, 1 'LED off
  wait 100
until inkey > -1
end

```

Now for a few more features of the EZmon editor. Press e to edit the program. You see the (old) program on the screen. Lets change the program so it is easy to change the rate at which the LED flashes. Change the program to the program in the screen capture.





```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help
Editor |..... Ctrl-\ Help
'Blinky, flash the Red LED
RedLED=41
PINMODE RedLED, OUT      'Pin 41 as output pin
LedDelay=200
REPEAT
  OUTD RedLED, 0          ' Turn LED on
  WAIT LedDelay           ' do nothing for LedDelay ms
  OUTD RedLED, 1          ' LED off
  WAIT LedDelay
UNTIL INKEY > -1
END
```

The word `LedDelay` is a new variable and it is assigned the value 200. The delay can now be changed in a single location. Make the changes by navigating with the cursor keys and typing. You will see that typing inserts new characters. The up and down cursors change lines and left and right cursors move to the left and right. Home, End, PageUp, PageDown, Delete, Tab and Backspace all work if Tera Term or your favorite terminal emulator is configured correctly. Pressing the Insert key changes the mode to Overstrike and back to Insert mode if you press it again.

`Ctrl-W` will save the changes and quit. Just remember, this is a remote terminal, it is completely unaware of the mouse, so keystrokes only.

Run the program again. The LED now flashes at half the earlier rate. While the program is running press `Ctrl-C` and see the text 'Ctrl-C detected' followed by a line that starts with a number followed by a colon maybe

```
10: UNTIL INKEY > -1
```

Now press the Space Bar and see the code appear a line at a time like this:

```
Ctrl-C detected
10: UNTIL INKEY > -1
6:   OUTD RedLED, 0          ' Turn LED on
7:   WAIT LedDelay         ' do nothing for LedDelay ms
8:   OUTD RedLED, 1          ' LED off
9:   WAIT LedDelay
10: UNTIL INKEY > -1
6:   OUTD RedLED, 0          ' Turn LED on
7:   WAIT LedDelay         ' do nothing for LedDelay ms
8:   OUTD RedLED, 1          ' LED off
9:   WAIT LedDelay
10: UNTIL INKEY > -1
```

```

6:      OUTD RedLED, 0      ' Turn LED on
7:      WAIT LedDelay      ' do nothing for LedDelay ms
8:      OUTD RedLED, 1      ' LED off
9:      WAIT LedDelay
10: UNTIL INKEY > -1
6:      OUTD RedLED, 0      ' Turn LED on

```

You are stepping through the program a line at a time. The number before the colon is the line number to be executed next and the text is the program code to be executed. To continue running the program at full speed type *g*.

Change the program again so that it looks like the screen capture below

The screenshot shows a terminal window titled "COM6:57600baud - Tera Term VT". The window contains the following text:

```

Control BASIC v0.59.
R-Run  S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date

Program Listing:
'Blinky, flash the Red LED
RedLED=41
PINMODE RedLED, OUT      'Pin 41 as output pin
LedDelay=RND(-123)      'Initialize the random number generator
REPEAT
  LedDelay=RND( 500 )
  OUTD RedLED, 0      ' Turn LED on
  WAIT LedDelay      ' do nothing for LedDelay ms
  OUTD RedLED, 1      ' LED off
  WAIT LedDelay
UNTIL INKEY > -1
END

End of Listing

Control BASIC v0.59.
R-Run  S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date

```

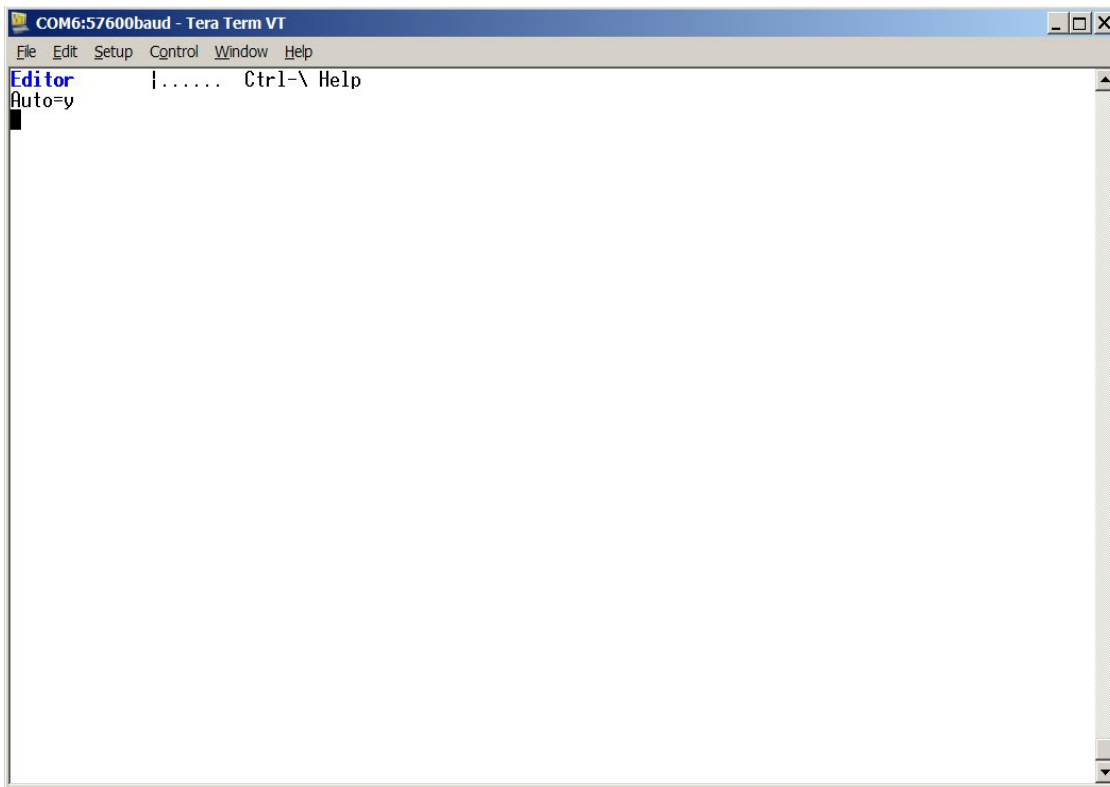
and run the program. The LED now flashes in some unpredictable way, not very useful but good for demonstrating another feature of EZmon. `LedDelay = RND(500)` assigns a random value between 0 and 500 to `LedDelay`. Press *s* and note that the program starts in single step mode. After every line press *v* before pressing the Space Bar. As the variables are defined and their values change they can be examined. Since the delay changes randomly the only way to know the value is to print it to the screen or view it by single stepping the program.

Having an embedded controller is not much use if it does not run automatically when the power is applied or the reset button is pressed. That leads to the next section.

## Running a Program Automatically

Stop the program if it is running. Press *c* and you will see the editor open up but it will not show the program.

Type Auto=y and save the edit (*Ctrl-W y*). See that the program started running immediately when you quit the



editor. When you press a key on the keyboard you see the lines

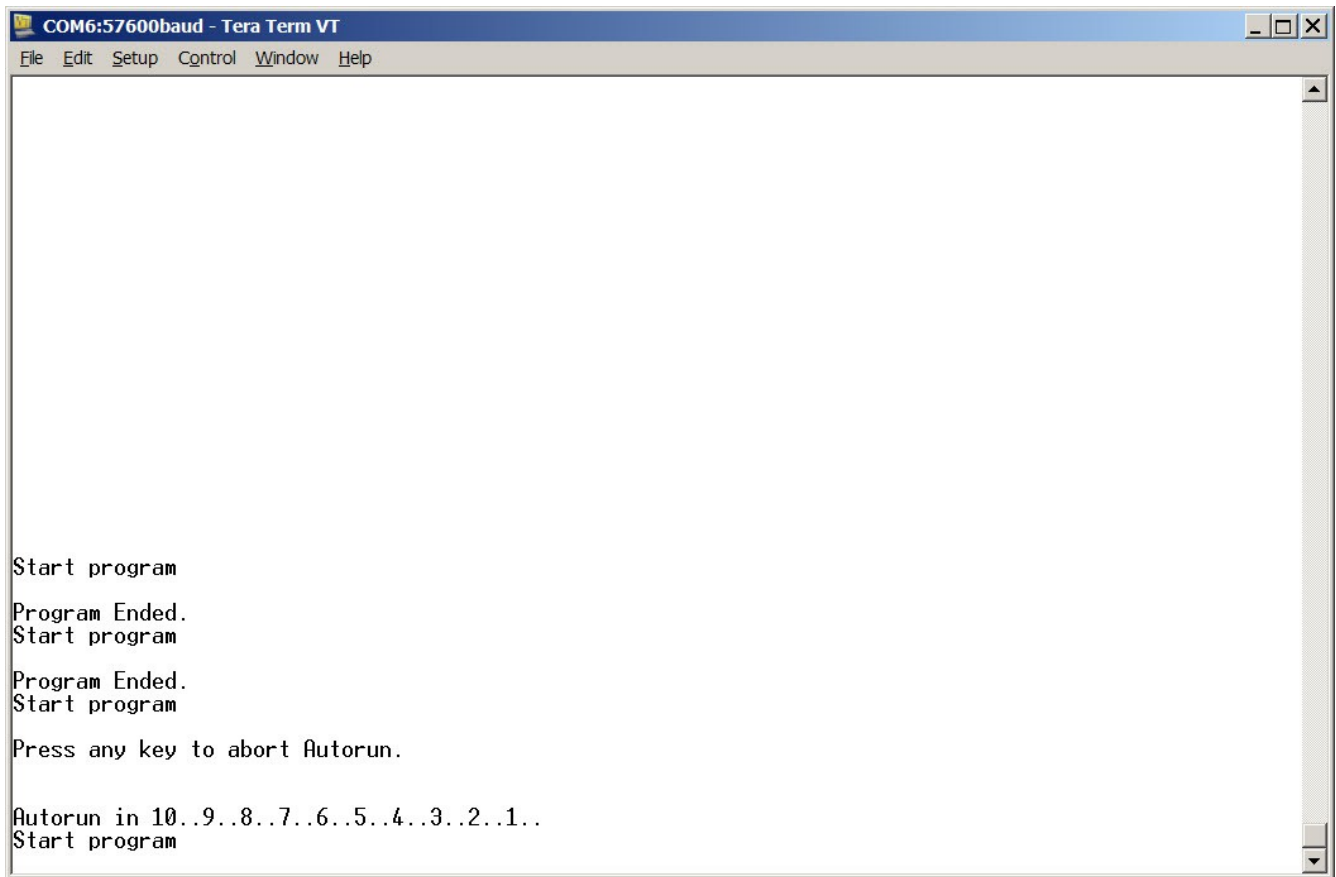
```
Start program
```

```
Program Ended.
```

```
Start program
```

appear on the display. The program now runs automatically on start up and will restart if it ends, like a real embedded controller should.

Press the Reset button on the EZSBC1 and you will see this message appear on the terminal window:



Now press the reset button again and before the countdown gets to zero press a key on the terminal emulator keyboard and this is what you will see:

```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help

Start program
Program Ended.
Start program

Program Ended.
Start program

Press any key to abort Autorun.

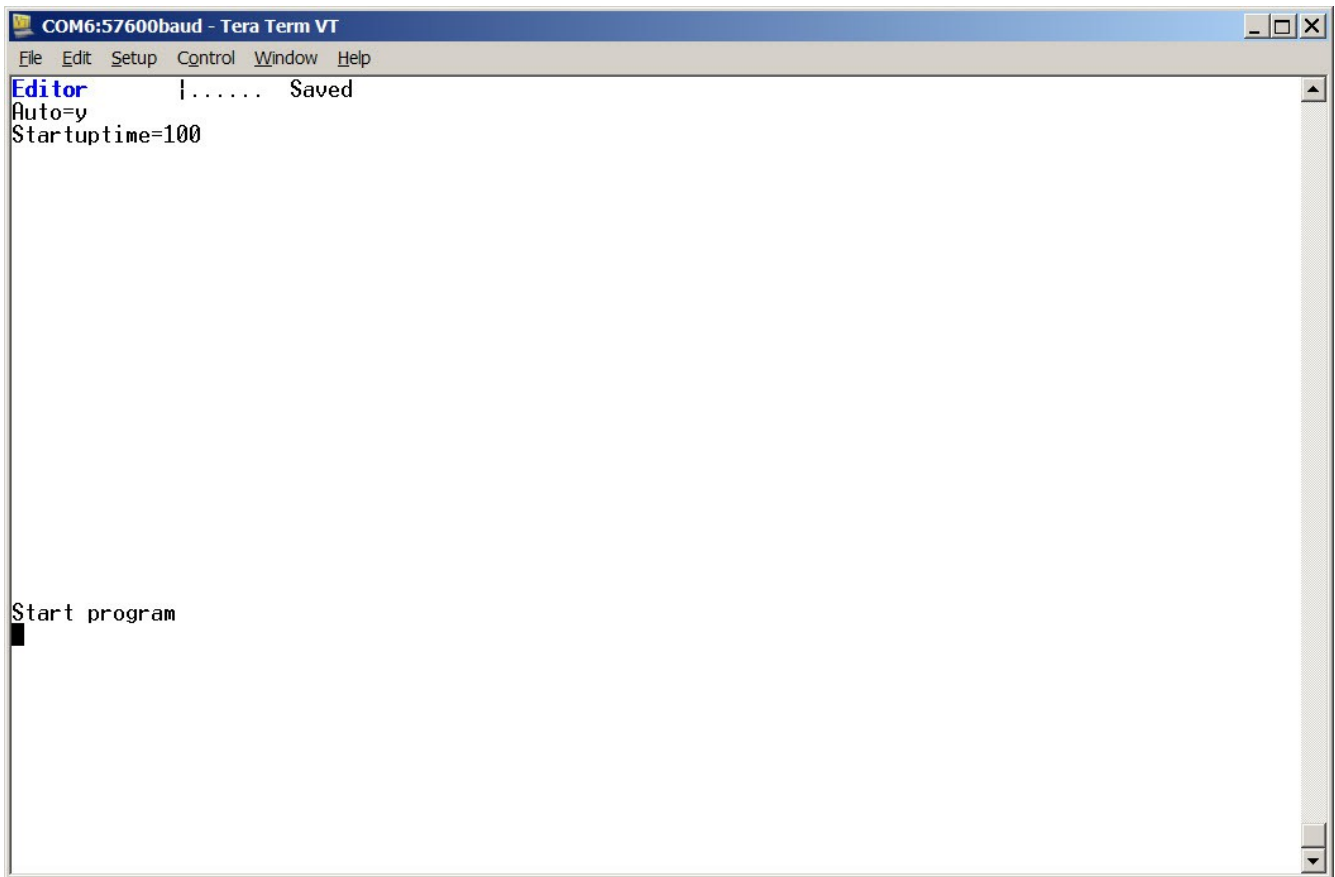
Autorun in 10..9..8..7..6..5..4..3..2..1..
Start program

Press any key to abort Autorun.

Autorun in 10..9..8..7..

Control BASIC v0.59.
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date
```

Change the Configuration parameters by pressing *c* and then typing the new entry till the screen looks like the first three lines on the screen capture. Save the changes to the Configuration parameters by typing *Ctrl-W y*.

A screenshot of a Tera Term VT terminal window. The title bar reads "COM6:57600baud - Tera Term VT". The menu bar includes "File", "Edit", "Setup", "Control", "Window", and "Help". The main text area shows an editor with the following content: "Editor", "Auto=y", "Startuptime=100", and "Start program" with a cursor on the line below. The status bar at the top right shows "Saved".

```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help
Editor |..... Saved
Auto=y
Startuptime=100

Start program
█
```

Now the delay before the program starts is very long, around 50 seconds. As a precaution against never being able to regain control of the EZSBC1 values of  $x < 5$  are ignored 5 is used. The default value is 10 when the parameter is not specified. Specifying Bank=0 (the default) or Bank=1 selects from which bank the program is started.

See the CONF\$ keyword for more uses of the configuration area.

## Memory Bank Selection

The EZSBC1's program memory is divided into two almost equal sized banks of memory, the first one Bank0 being 64k Byte in size and the second bank being 60k Byte in size. Four kilobytes of flash is 'stolen' from the second bank to implement the Configuration Area. The memory is divided into two banks for a few reasons. It is very useful to have a place to test pieces of code without going through the effort of saving the program to the host computer, loading a test program and the downloading the saved program again.

Some systems have two distinct behaviors. A good example of such a system is a data logger. When the system is in the field collecting data it is busy with one very specific task. When the system is retrieved and the data needs to be downloaded to a host computer the behavior and requirements are entirely different from the logging requirements. By having the option to split the program into the two banks the two resulting programs are much simpler to design and implement than one big program that has to perform both sets of tasks. By loading one program in each bank and selecting which bank's program executes, the system has two entirely different personalities, each one fairly easy to implement.

When EZmon is active, press the B key and a three lines of text will appear:

```
Current bank: 1
Bank #0 contents: 'Blinky, flash the Red LED
Bank #1 contents:
```

The first line of code from each bank is displayed as well as the currently active bank. Pressing 0 or 1 selects the appropriate bank, any other key causes no change. Since the first line of code is displayed it is good practice to make the first line of any program a comment describing the function of the program.

Specifying `Bank=0` (the default) or `Bank=1` in the Configuration area selects from which bank is active after power-up or reset.

## Download a Program

It gets very tiresome to type long programs using only the editor function of EZmon. It is intended for making quick changes to programs in the field or while debugging. Writing programs is best done on the PC using your favorite programmers editor. Programmers Notepad and CodeLite are both free editors for writing programs. Don't write the program using Wordpad or similar word processors; they do not store the programs in ASCII format.

By pressing *D* at the prompt the Controller will wait for a program to save into the current bank. Immediately pressing `Ctrl-D` will clear the bank. Using the 'Send File' option on the TeraTerm **File** menu will download a program to the controller. The program must be in ASCII. For large programs the download operation can take some time. The controller writes the Flash in 256 byte blocks and each block is erased before the new code is written into the block of Flash. Also, the program is not stored in ASCII in the Flash but rather in a tokenized format to save space and increase execution speed. Tokenization occurs after the block is downloaded but before it is programmed into the Flash. When the controller is busy writing to the Flash, it can not save data arriving on the serial port. When the EZSBC1's buffer is nearly full or prior to programming or erasing the Flash, the controller sends the X-Off character to the host to temporarily suspend serial communication via the USB. When the controller is ready for more data it sends the X-On character to allow the host to resume sending data. To enable this feature to work Xon/Xoff flow control must be enabled on the serial port connecting to the controller

## Deleting an Entire Program

A quick way to get a blank page (bank/program) is to 'Download' nothing, an empty program. Get to the EZmon prompt:

```
Control BASIC v0.59.  
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date
```

Type '*d*'

You should see:

Send program.

Use XON/XOFF flow control.

Press `Ctrl-D` after download completes.

Now type *Ctrl-D* (Control key and D simultaneously) You should see:

```
Programmed bytes 0x100
```

Meaning that 256 bytes of the program memory was written to (out of 65535 bytes).

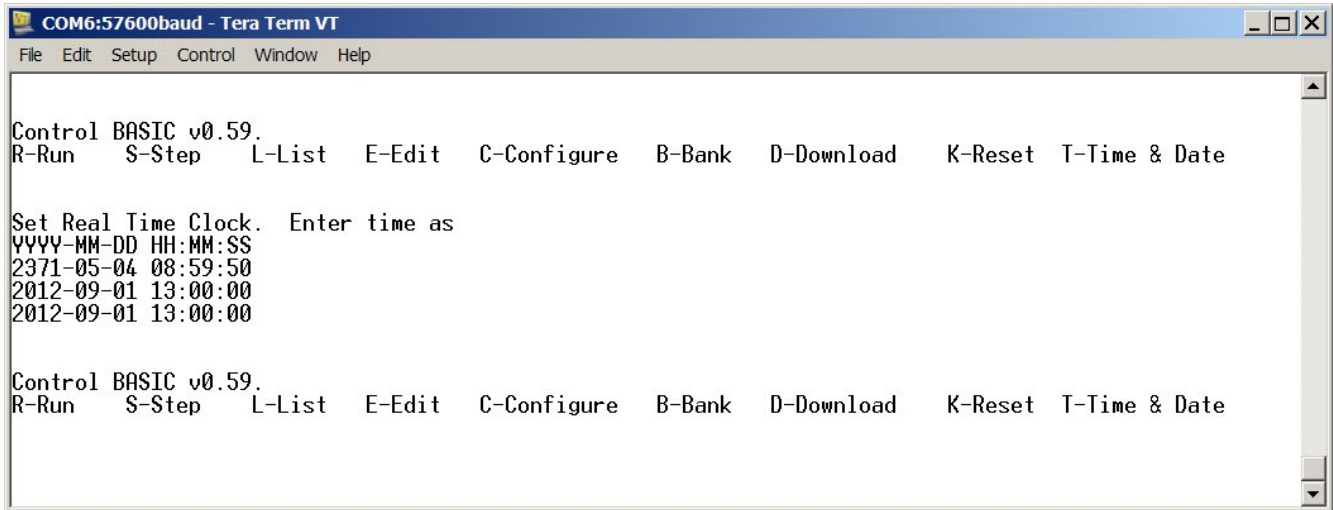
## Setting the Time

```
Control BASIC v0.59  
R-Run S-Step L-List E-Edit C-Configure B-Bank D-Download K-Reset T-Time & Date
```

Press *T* and you should see something like this:

Set Real Time Clock. Enter time as  
YYYY-MM-DD HH:MM:SS  
2371-05-04 08:54:61

Now type the date and time exactly as shown using the - key next to the 0 on the keyboard, not the keypad -. Hours must be entered in 24 hour format so to specify 1PM, enter 13 in the hour position. Leading zeros must be entered. Any error will not update the time. A successful update will look like the screen capture below



```
COM6:57600baud - Tera Term VT
File Edit Setup Control Window Help

Control BASIC v0.59.
R-Run  S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date

Set Real Time Clock. Enter time as
YYYY-MM-DD HH:MM:SS
2371-05-04 08:59:50
2012-09-01 13:00:00
2012-09-01 13:00:00

Control BASIC v0.59.
R-Run  S-Step  L-List  E-Edit  C-Configure  B-Bank  D-Download  K-Reset  T-Time & Date
```

If you press the T-key again you will see that the time has advanced.

The Real Time Clock can be powered by connecting a Lithium coin cell to pin 38 of the EZSBC1 and it will keep time while the main power is off. There is an automatic power changeover circuit included on the EZSBC1.

## Resetting the EZSBC1

There are two ways of resetting the EZSBC1. Pressing the reset button on the EZSBC1 performs a hardware reset on the module and the RSTn pin on the module will be driven low to reset external device which are tied to the pin. The USB-bus does not enumerate again when the reset button is pressed.

By pressing *K* at the prompt causes a hardware reset of the CPU on the EZSBC1. The USB bus is not reset allowing the terminal connection to be maintained. Also, the Reset pin (RSTn) on the controller board does not toggle when the Reset command is used. Some BASIC keywords have effects even after the program ends. A good example of such an instruction is the PWM command. Once the PWM command is given, the pin will be driven even after the program ends. The Reset command is useful to restore the IO-pins to their power on state without having to reset the entire system.

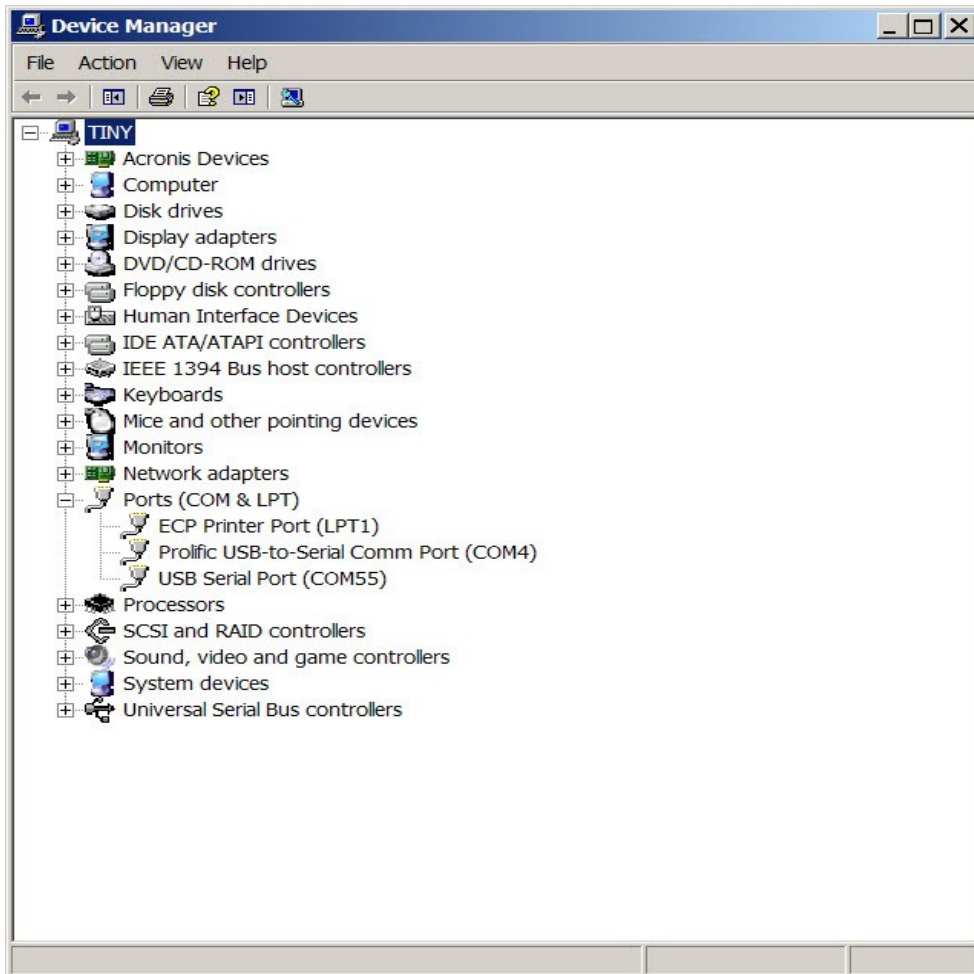
The RSTn pin has a pull-up resistor on the EZSBC1 and can be driven low by external components to reset the entire system. The reset controller used on the EZSBC1 will hold the reset pin low for a time (30ms) after the reset pulse disappears to ensure a valid reset condition for the EZSBC1. External components should not drive the RSTn pin high.



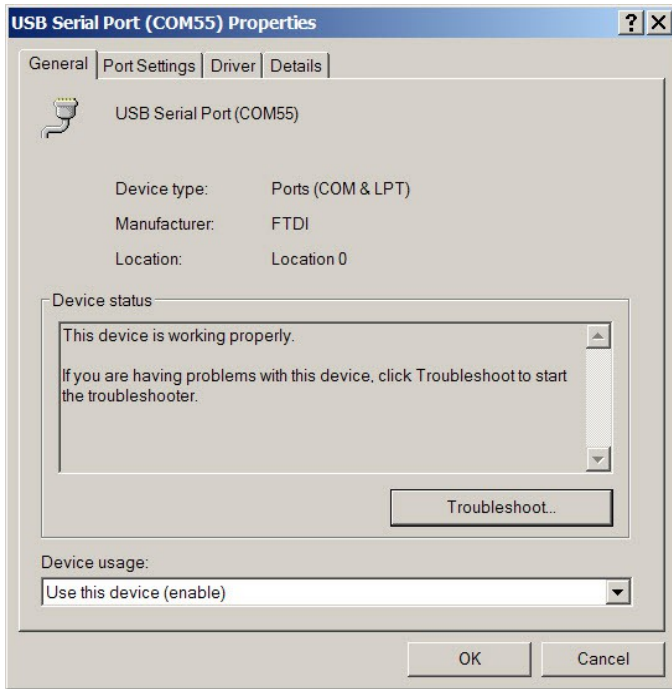
## Appendix A

### Reset the Comm Port Number

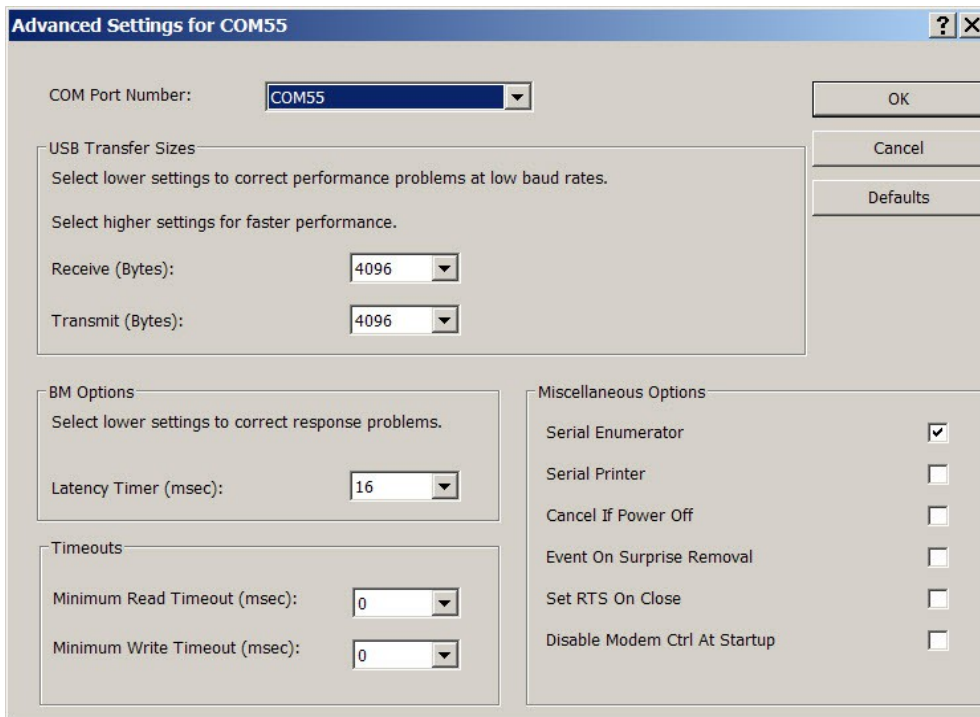
Not all programs are happy dealing with double digit COMxx number. You may want to use a data logging program that needs a low COMx port number and Windows assigned a large number such as COM55 (in this example). It is possible to reset the number.



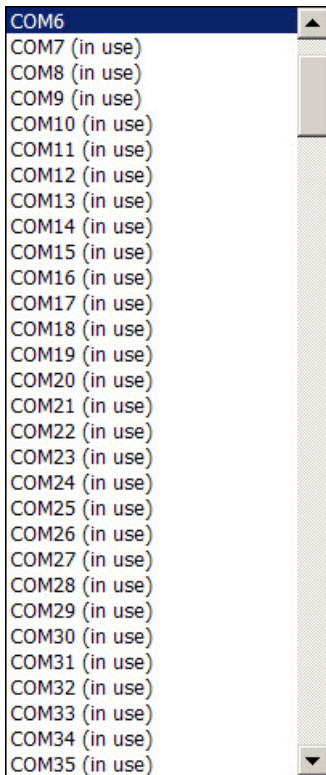
In the Device Manager right click on the serial port with the number that you want to re-assign. Select the 'Properties' option.



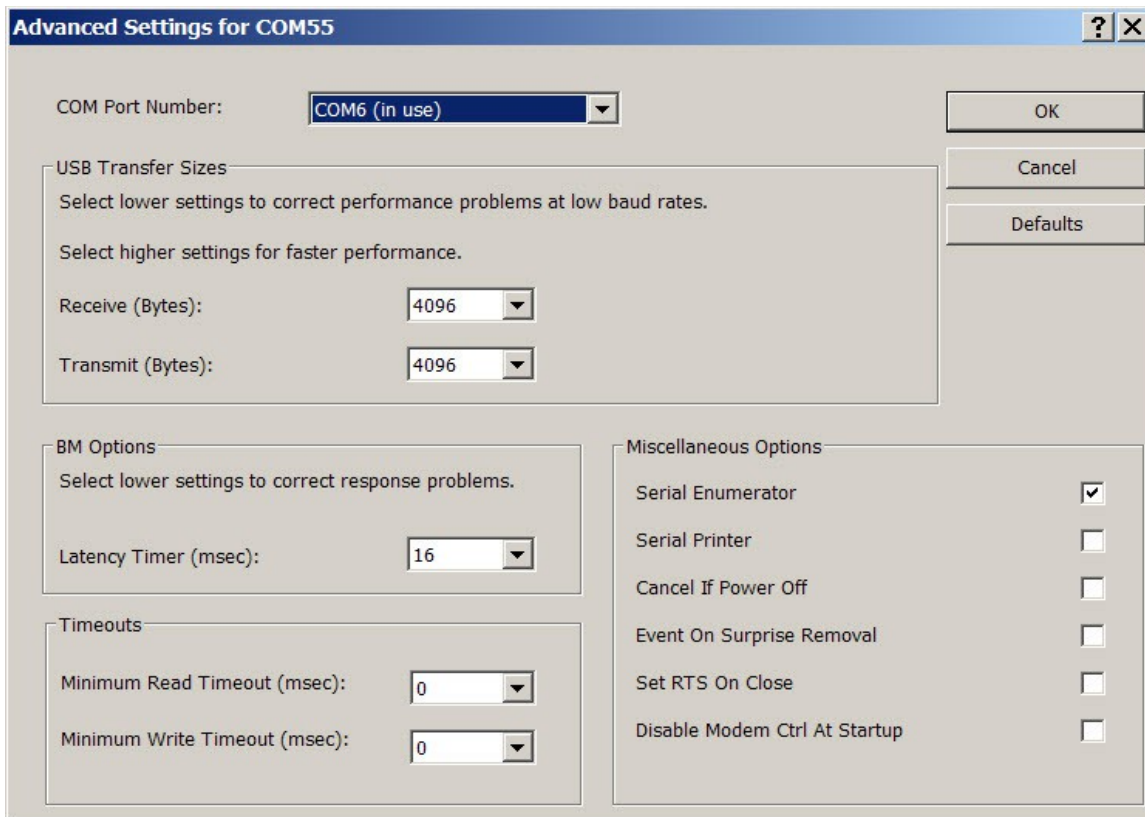
and now select the 'Port Settings' tab and on the Port Settings tab, press the Advanced button.



Click the COM Port Number drop down item and it will show a list of available ports. On the machine used here it shows a ridiculous list of ports as in use.

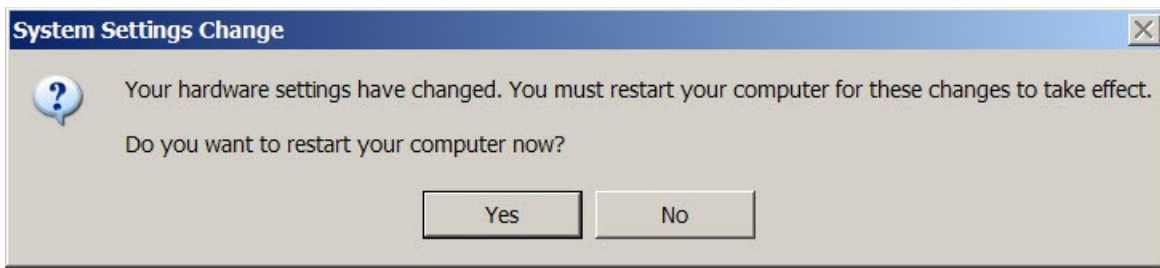


It shows COM6 as unused so selecting it assigns it to the serial port. If all the ports are marked as 'in use' select one that you know is not (actually) in use.



Here I selected COM6 that is shown as in use. Press OK. Close all the menus. You may get an prompt

to restart the computer.



After re-starting the computer, re-visit the Device Manager and the serial port will have the assigned COM number.