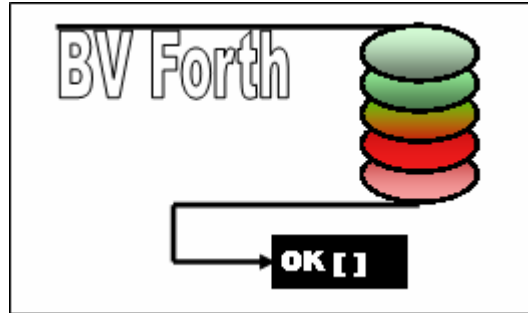


# BV511 Hardware Guide



## ByVac

©ByVac 2007

[www.byvac.co.uk](http://www.byvac.co.uk)

Revision 1.0

Copyright in this work is vested in ByVac and the document is issued in confidence for the purpose only for which it is supplied. It must not be introduced in whole or in part or used for tendering or manufacturing purposes except under an agreement or with the consent in writing of ByVac and then only on condition that this notice is included in any such reproduction.

© Copyright ByVac 2007

**No warranty**

THE WORK IS PROVIDED "AS IS," AND COMES WITH ABSOLUTELY NO WARRANTY, EXPRESS OR IMPLIED, TO THE EXTENT PERMITTED BY APPLICABLE LAW, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

**Disclaimer of liability**

IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS WORK, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 1. Introduction

The BV511 is an ARM7 based microcontroller board using a Philips LPC2132 processor with built in real time clock. The main serial interface and method of communication with the board is provided by USB.

The primary purpose of the board is to run Forth as its operating system and as a special feature of this operating system and board combination makes the early stages of implementing much quicker than would otherwise be possible on conventional compilers (C like) systems. There is nothing to stop this board from being used with other languages or development systems but the focus of this and the other guides is using BV Forth.

The following guides form the document set for this system.

- BV Forth Language Guide
- BV Forth Core Glossary
- BV Interface Foundation

The board can be used for educational purposes, as a development board or incorporated into a stand alone system.

### 1.1. Features Hardware

- LPC2132 ARM Processor
- 32 Bit architecture
- 64k Flash
- 16K RAM
- USB interface
- 5V and 3.3V regulated outputs
- RTC with battery back up
- IASI, second UART interface
- 3.3V core with 5V tolerant ports
- 45 plus I/O lines
- Low power 40mA nominal
- 10bit D to A
- 8 channel A to D
- Two 32 bit timers
- I2C, SPI and SSP interfaces
- 60MHz CPU clock frequency
- Power saving modes

### 1.2. Features Firmware

- Self contained operating system
- Expandable
- Easy to use

- Auto run at start up

The firmware resides on the first 20K (approximately) of Flash and can be replaced or re-programmed if required. Details of how to do this are in the firmware section of this text.

## 2. Getting Started

The BV511 is ready to run out of the box, getting started is a question of installing the USB driver so that it can talk to the host system.

The instructions are based around a Windows system but there are also USB drivers available for Linux systems. A very comprehensive set of instructions for installing the USB driver including for other operating systems are on the CD-ROM in the installation guides directory.

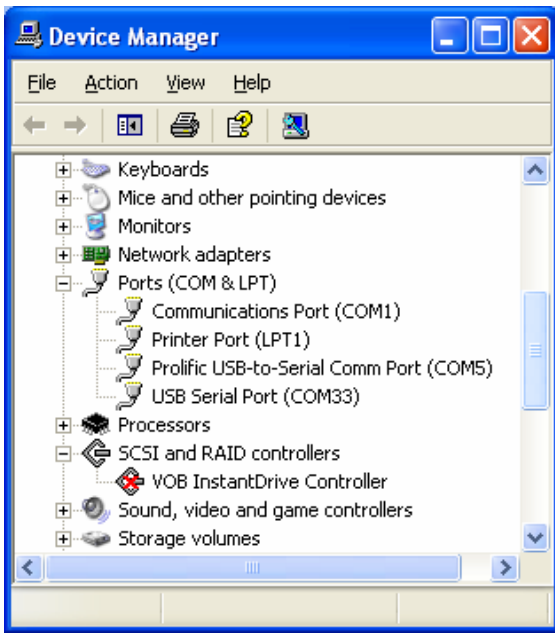
The following is an abbreviated version of how to install on Windows XP.

### 2.1. Step 1 Connect the BV511

Connect the BV511 to a spare USB port using the cable provided. The port can be USB versions 1.0 , 1.1 or 2.0. As this is a serial interface the speed of the USB is not important and will not effect the performance of the BV511.

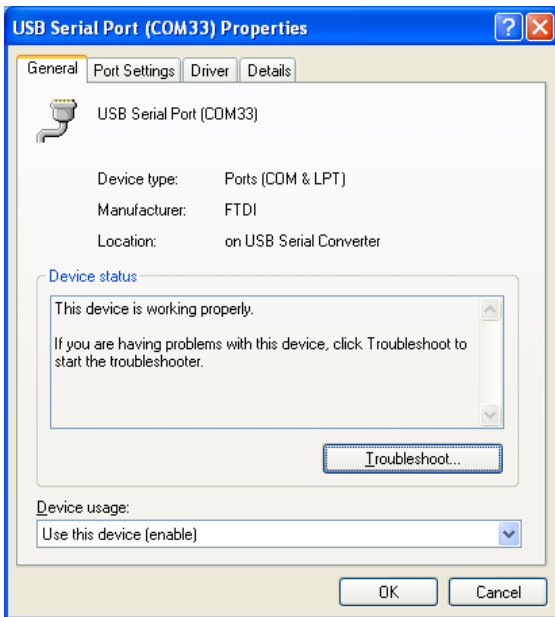
### 2.2. Step 2 Install the USB Driver

The USB driver is located on the CD-ROM, assuming a Windows install, when asked, choose 'install from a list or specific location'. The latest drivers and installation documents can be obtained from [www.ftdichip.com](http://www.ftdichip.com). The driver required is the **VCP** (Virtual Com Port). The VCP driver will install itself as a new Com port.



**Figure 1 New COM Port Installed**

On this particular machine the new com port is port 33. This will need changing to a lower number if ISP is going to be used but it is okay for now.

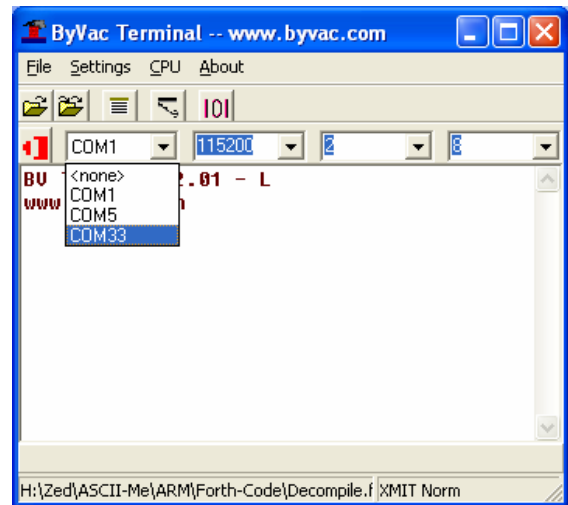


**Figure 2 FTDI Confirmation**

By double clicking on the new port it is possible to confirm that it is the correct one by looking at the manufacturers name (FTDI).

### 2.3. Step 3 Sign On

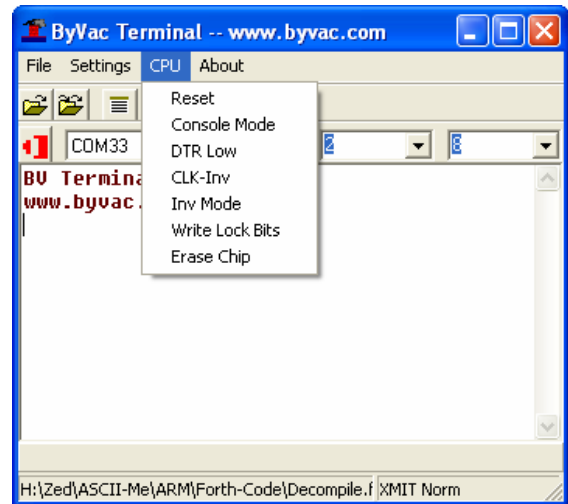
Any terminal emulator can be used but for simplicity BV Term will be used, it is on the CD-ROM and is free.



**Figure 3 Start BV Term**

BV Term is a single exe file that will run from the CD-ROM, it is better though to copy it to the hard drive as it creates a file to hold the current settings in, saving the trouble of doing this each time.

Using the drop down look for the new com port that has been created, com33 as in the case of Figure 3. The other settings should be as the figure, **Baud rate 115200, 2 stop bits (1 will do) and 8 data bits.**



**Figure 4 CPU menu should have nothing ticked**

Make sure there is nothing ticked on the CPU menu and use the red icon to connect and this will then turn green.

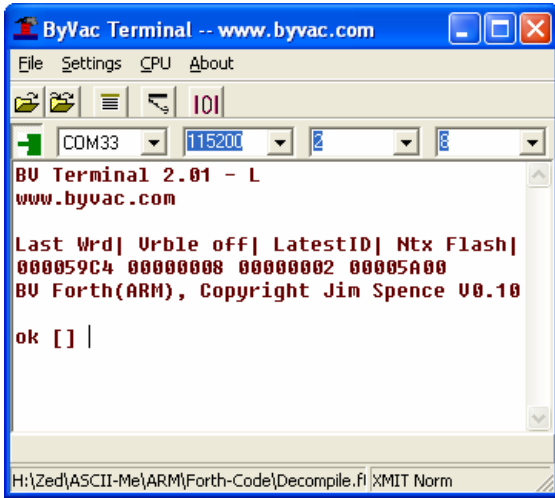



Figure 5 Sign on

The sign on message should appear as above, if not press the reset  icon.

Just to make sure everything is working type:

**12 12 \* .**

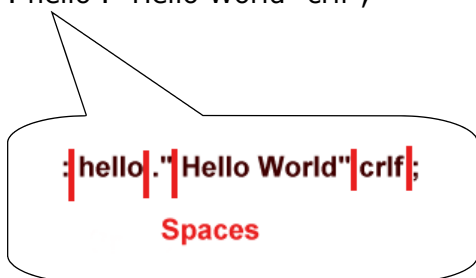
(This is 12 **space** 12 **space** star **space** dot and return, the **spaces are important**)

The console should return the value 144 which is 12 \* 12.

### 2.4. Step 4 First Program

Now type the following carefully:

`: hello ." Hello World" crlf ;`



Note that **spaces are important**, there is a space between the : and hello and also

between ." and Hello. This may not be obvious from the printed text. Now type hello and the 'program' will run to print out 'Hello World' on the console.

**That's it a program in less than 20 minutes.**

### 2.5. Step 5 Save

This step is optional but you can now save all of the words downloaded, including the 'hello' word simply by typing SAVE, the words are now saved to flash that and are available at switch on, including your new word 'hello'

## 3. Useful Words

The following is a selection of words that will get you going without having to search through the documentation or glossary:

**SAVE** [optional word]

SAVE on its own will save the words in RAM to the Flash so that they will be there at switch on next time. SAVE used in the form SAVE MY-WORD will do the same but when the device is reset; MY-WORD will run.

**NEW** will clear the Flash and leave the 20k core Forth system.

**COLD** is the same as a hardware reset.

**For further practice go to the BV ARM Foundation book (B2).**

### 3.1. Battery Installation

Some of the later texts cover the use of the Real Time Clock (RTC), while this works perfectly well without the battery, the clock will stop once the power is removed. The battery will apply power to the RTC section of the ARM and keep the oscillator going between power downs.

The battery is installed with the + facing upwards.

## 4. BV511 Hardware description

The board can be explained in terms of the block diagram as shown below.

### 4.1. Power Supply

The power supply is derived from either the USB connector or an external source, either way the power is fed into a 5V low drop out voltage regulator. This serves two purposes, 1. it will regulate the voltage from the external source so that up to 10V DC can be used as a power supply. And 2. it isolates the USB from the system so that accidental short circuits will not affect the host system.

The 5V supply is used to feed an external output, the IASI connector and the 3V3 regulator. The 3V3 regulator is used to supply power to the ARM and also a power output on connector 2 (see Figure 6).

The USB chip is supplied only from the USB and thus when the USB is disconnected it is powered down, thus saving power.

### 4.2. Serial Interfaces

The primary serial interface is via the USB port and the USB to serial converter chip. Two other relevant signals are available from this chip and that is the reset and In System Programming Signal.

#### 4.2.1. Reset

The former is connected to the DTR (Data Terminal Request) which in turn is connected to the reset circuit on the ARM board. By pulsing this line a reset is

obtained without having to physically touch the board. Some terminal software has this facility built in as it is common practice. BV Term for example has a reset icon that will pulse this line.

As an alternative the board also has a physical reset in the form of connector holes, marked RST next to the power input pins. Shorting the RST holes together will reset the board.

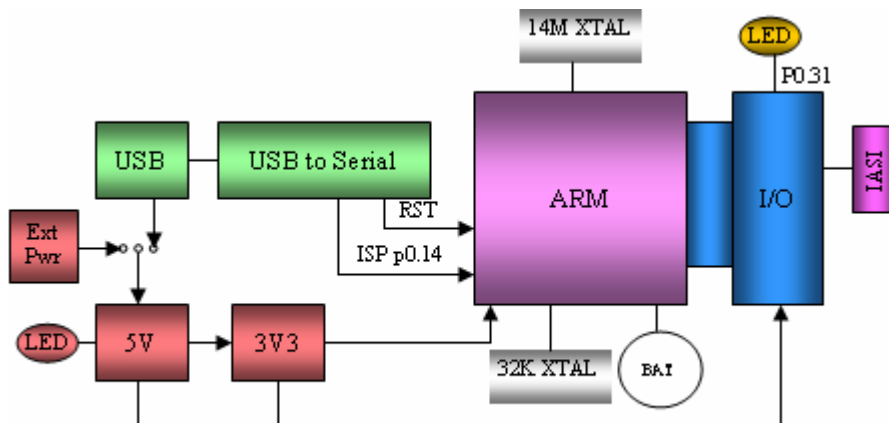
#### 4.2.2. ISP

In System Programming (ISP) is dealt with more fully in the section dealing with updating the firmware however the serial interface USB chip provides an output that is connected to p0.14. This makes it much easier to perform Flash programming as, again no physical contact is required with the board – the sequence is fully automatic.

#### 4.2.3. IASI

The ARM has two UARTS, UART0 and UART1 (Universal Asynchronous Receiver and Transmitter). UART0 is connected to the USB interface and UART1 is connected to the pins at the end of the board. This can be used for anything but the physical layout is intended for interfacing **IASI** (Intelligent Asynchronous Serial Interface) parts.

The firmware sets UART1 at 9600 Baud but can easily be changed by writing to the appropriate registers. UART0 is set for 115200 Baud.



Block Diagram

## 5. System Clocks

There are two system clocks, the real time clock (RTC) and the processor clock. The RTC is regulated by the 32KHz crystal, this enables reasonably accurate timing for the seconds counter. The section of the processor that handles RTC is powered separately by the on board lithium battery. It is also powered by the 3V3 regulator when power is available by a twin diode arrangement. The clock therefore keeps going when the power is disconnected.

The processor clock is driven by the 14.7456 MHz crystal. The firmware quadruples this frequency to about 60 MHz by using an on board Phase Locked Loop.

## 6. Input / Output

The LPC2132 has 64 I/O lines sectioned into two ports, port 0 and port 1. All of the lines on port 0 are available for general purpose I/O but only the top half of lines are available for I/O on port 1. This reduces the total I/O to 48, however some lines are used for specialist purposes so this reduces the total I/O on the board to 46.

All of the available I/O lines are taken to the two side sockets that are 24 pins wide. It must be noted though that some of the lines, in particular P0.0, p0.1 (main serial interface) p0.8,p0.9 (IASI interface) P0.14 (ISP) and P0.31 (LED) are used for secondary purposes.

### Port Notation

Pn.x Is the notation used for specifying ports where n is the port and x is the pin of that port, thus port 1 pin 21 would be p1.21

## 7. Physical Configuration

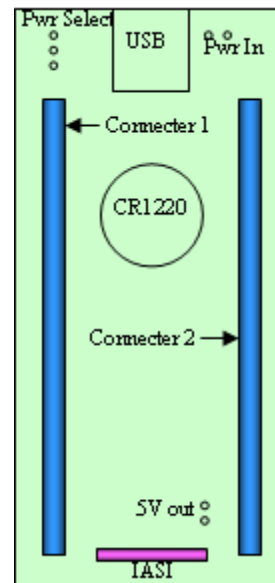


Figure 6 Physical Layout

### 7.1. Input Power

The power select on board jumper selects the source of power, this can be taken from either the USB or an external source. The power select jumper looking at the board as shown in Figure 6 is to the left hand side of the USB connector.

Pin	
1	
2	Link
3	

Power taken from USB

Pin	
1	Link
2	
3	

Power taken from External Source

The USB interface is provided by a USB to com port interface chip. The power for this chip is provided ONLY when the link is in position 2-3 (powered from USB). When the external source provides power the chip is not active and therefore the USB is not active.

The reason for implementing this way is that on a stand alone system the USB interface will be redundant and therefore

should not contribute to the power consumption.

If the USB interface is required then it makes sense to take the power from the USB to power the USB chip.

When taking power from an external source with pins 1 and 2 linked, the power goes through a regulator. A DC source up to a maximum of 10V can provide power to the board. The power is applied to the two pins on the right hand side of the USB.

**NOTE There is no reverse polarity protection for these pins, take care.**

## 7.2. Output Power

There are three places where power can be 'tapped' off of the board: connector 2, the IASI connector and the 5V out connector.

Connector 1 provides an output from the 3v3 voltage regulator to power external 3v3 devices. The power out and IASI connectors have a feed from the 5V regulator to power 5V devices. As the regulators are surface mounted and don't have heat sinks the current taken should be sensibly limited. The 5V regulator has a maximum rating of 150mA and the 3V3 regulator has a maximum rating of 100mA. Some of this of course is used to power the ARM and as all of the power goes through the 5V regulator first, the 5V regulator will be affected.

## 7.3. USB Isolation

One of the main features of having on board regulation is that it effectively isolates the USB power and offers some protection to the host PC. The power output pins can be sorted for example without any ill effect on the host system.

It will not of course isolate the USB from extreme conditions, high voltages on the bus for example so some care must be still exercised.

## 8. Connectors

The pin designations for the connectors are clearly marked on the PCB and are repeated here:

Connector 1		Connector 2	
24	Port 1 pin 24	24	Port 0 pin 8[2]
23	Port 0 pin 7	23	Port 0 pin 9[2]
22	Port 0 pin 6	22	Port 0 pin 10
21	Port 0 pin 5	21	Port 1 pin 23
20	Port 1 pin 25	20	Port 0 pin 11
19	Port 0 pin 4	19	Port 0 pin 12
18	Port 0 pin 3	18	Port 0 pin 13
17	Port 1 pin 26	17	Port 1 pin 22
16	Port 0 pin 2	16	Port 0 pin 14[3]
15	Port 0 pin 1[1]	15	GND
14	Port 1 pin 31	14	3V3
13	Port 0 pin 0[1]	13	Port 1 pin 21
12	Port 1 pin 16	12	Port 0 pin 15
11	Port 0 pin 30	11	Port 0 pin 16
10	Port 0 pin 29	10	Port 0 pin 17
9	Port 0 pin 28	9	Port 1 pin 20
8	Port 1 pin 17	8	Port 1 pin 30
7	Port 0 pin 27	7	Port 0 pin 18
6	Port 0 pin 26	6	Port 0 pin 19
5	Port 0 pin 25	5	Port 0 pin 20
4	Port 1 pin 18	4	Port 1 pin 29
3	Port 1 pin 19	3	Port 0 pin 23
2	Port 0 pin 22	2	Port 1 pin 28
1	Port 0 pin 21	1	Port 1 pin 27

NOTES:

1. p0.31 is used for the on board LED and does not appear on any port.
2. [1] This is the serial interface to the UART
3. [2] Serial interface used for the IASI connector
4. [3] Used during ISP, has a 10k pull up resistor to 3v3.

Nearly all of the pins have secondary functions in particular those using port 0. The following tables conveniently group the major functions together.

UART0		
Pin	Name	Function
P0.0	TXD	1
P0.1	RDX	1

UART 1		
Pin	Name	Function
P0.8	TXD	1
P0.9	RXD	1

PWM		
-----	--	--

Pin	Name	Function
P0.0	PWM1	2
P0.1	PWM3	2
P0.7	PWM2	2
P0.8	PWM4	2
P0.9	PWM6	1

Interrupt 0		
Pin	Name	Function
P0.1	EINT0	3
P0.16	EINT0	2

Interrupt 1		
Pin	Name	Function
P0.3	EINT1	3
P0.14	EINT1	2

Interrupt 2		
Pin	Name	Function
P0.7	EINT2	3
P0.15	EINT2	2

Interrupt 3		
Pin	Name	Function
P0.9	EINT3	3
P0.20	EINT3	3

I2C 0		
Pin	Name	Function
P0.2	SCL0	1
P0.3	SDA0	1

I2C 1		
Pin	Name	Function
P0.11	SCL1	3
P0.14	SDA1	3

SPI 0		
Pin	Name	Function
P0.4	SCK0	2
P0.5	MISO0	2
P0.6	MOSI0	2
P0.7	SSEL0	2

SPI 1 (SSP)		
Pin	Name	Function
P0.17	SCK	2
P0.18	MISO	2
P0.19	MOSI	2
P0.20	SSEL	2

A to D 0		
Pin	Name	Function
P0.4	AD0.6	2
P0.5	AD0.7	2
P0.25	AD0.4	1
P0.26	AD0.5	1
P0.27	AD0.0	1
P0.28	AD0.1	1
P0.29	AD0.2	1
P0.30	AD0.3	1

Timer 0		
Pin	Name	Function
P0.2	Capture 0.0	2
P0.3	Match 0.0	2
P0.4	Capture 0.1	2
P0.5	Match 0.1	2
P0.6	Capture 0.2	2
P0.16	Match 0.2	2
P0.16	Capture 0.2	3
P0.22	Capture 0.0	2
P0.22	Match 0.0	3
P0.27	Capture 0.1	2
P0.27	Match 0.1	3
P0.28	Capture 0.2	2
P0.28	Match 0.2	3
P0.29	Capture 0.3	2
P0.29	Match 0.3	3
P0.30	Capture 0.0	3

### 8.1. IASI

The IASI (Intelligent Asynchronous Serial Interface) is provided by 5 pins. This is simply connected to the second UART (Universal Asynchronous Receiver and Transmitter) of the ARM. There are several IASI devices that can be purchased from

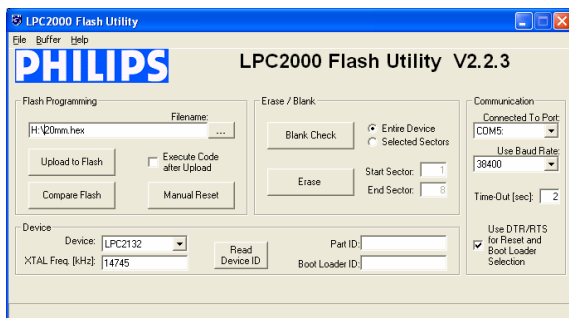
[www.byvac.com](http://www.byvac.com) to offload some of the work that the processor may need to do.

## 9. Updating the Firmware

The firmware can easily be updated or even replaced altogether. There are several reasons why this may be required:

- Reprogram the ARM using a different system in C or Assembler perhaps.
- Updating the BV Forth to a later version.
- Recovering from an endless loop programmed by the user and saved as an autorun word.

Philips have provided, free of charge, a utility for re-programming the flash memory. It is called the LPC2000 Flash utility. This can be obtained at the web site or a search on the net may reveal a later version. There is also one on the CD-ROM that came with the BV511.



**Figure 7 The LPC2000 Flash Utility**

All that is required to program the BV511 is to ensure that the Flash Utility has the same settings as shown in Figure 7, except of course the location and name of the file and the COM port.

The flash utility will read a HEX file and place this into the flash memory, the hex file must have been compiled for the address of the memory, this of course will be taken care of if you are simply replacing the operating system.

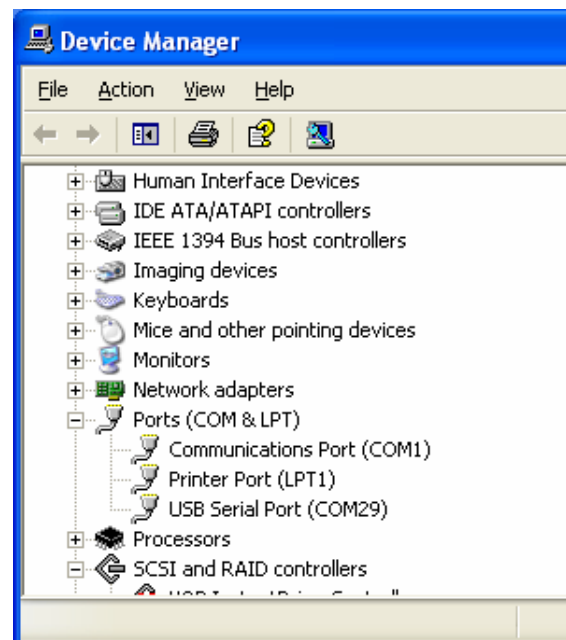
The system may work at 115200 Baud, the only way is to try it. Some systems will only work on the baud rate as shown.

The Philips mechanism for programming the LPC2000 part is to put the device into 'programming mode'. This is achieved by pulling port 0 pin 14 low and then resetting the device. On the BV511 this is taken care of automatically by the USB interface. The 'Use DTR/RTS for Reset and Boot Loader Selection' must be checked for this as shown in Figure 7

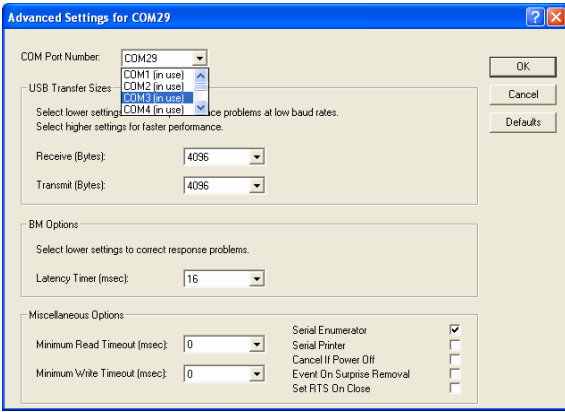
### 9.1. Com Port Numbers

Unfortunately the Flash Utility will only allow com ports from 1 to 5, this may be a problem if the USB driver has allocated a port higher than this. If this is the case it can be changed in the registry for Windows versions below Windows XP (not covered in this text). Or it can be changed using the control panel as follows:

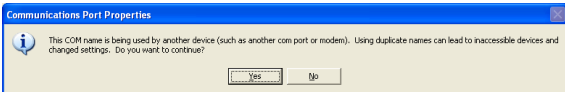
1. From the control panel select System
2. In the system dialog select the Hardware TAB
3. In the hardware tab select Device manager BUTTON
4. From the Device manager double click on the USB port, see Figure 8
5. From the Port Settings TAB, click the Advanced BUTTON
6. It will now be possible to select an alternative port number, see the next section on 'Com port in Use'



**Figure 8 USB Serial Port Selection**



**Figure 9 Changing Port number**



**Figure 10 Warning after changing the com port**

This warning will be issued if you select a com port that is already in use. You don't really have much choice if all of the com ports below 5 are in use.

### 9.2. Com Port In Use

The reason for allocating a higher number than 5 is that Windows has allocated the

next available number as all of the com ports below 5 are in use. They may well be in use in your system and you will know this by what is currently connected.

The problem with the 'in use' flag is that any port that HAS BEEN used is flagged up as in use whether is in use **now** or not. It is quite likely that plugging in lots of USB devices will use up the allocation for these ports, as a new port is allocated each time a new device is plugged into a different port. If for example you had a com port device and 4 USB ports, plugging that device into each USB port would allocate 4 com ports.

From this it can be seen that the 'In Use' designation is not that accurate. Choosing a port that is 'In Use' will give a warning, if you are sure that this port is not in use then go-ahead and use it.

## 10. Revisions B1

Doc.	Code	Date	Change
1.0	0.10	Apr 07	Release