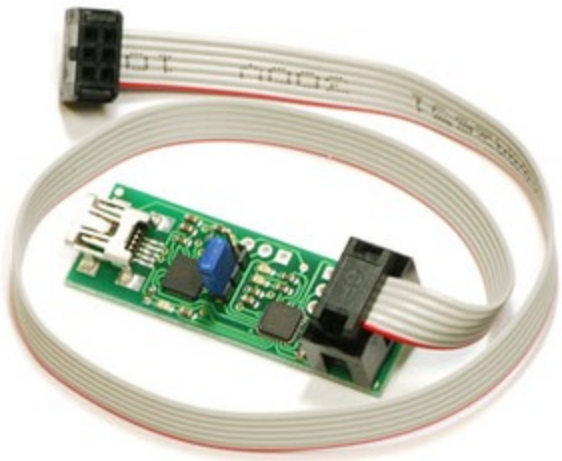
Pololu Orangutan USB Programmer User's Guide



1. Overview

The **Orangutan USB Programmer** [http://www.pololu.com/catalog/product/740] is a compact solution for programming our **Orangutan robot controllers** [http://www.pololu.com/catalog/category/8] or **3pi robot** [http://www.pololu.com/catalog/product/975] through a USB port. Our programmer incorporates a USB-to-serial adapter and emulates an AVR ISP programmer so that you can program your Orangutans or 3pi with any software that can talk through a serial port to an AVR ISP. The unit also doubles as a USB-to-serial adapter (TX and RX only), allowing you to communicate with your Orangutan, 3pi, or any other microcontroller, through a terminal program.

Unlike some other AVR programmers, the Orangutan USB programmer does not recieve its power from the circuit to be programmed. Instead, it receives its power via the **USB A to mini-B cable** [http://www.pololu.com/catalog/product/130] that connects it to a personal computer. This cable is not included with your programmer; if you do not already have one, you can purchase it separately from our web site. The 6-pin ISP programming cable that connects the programmer to your target device is included.

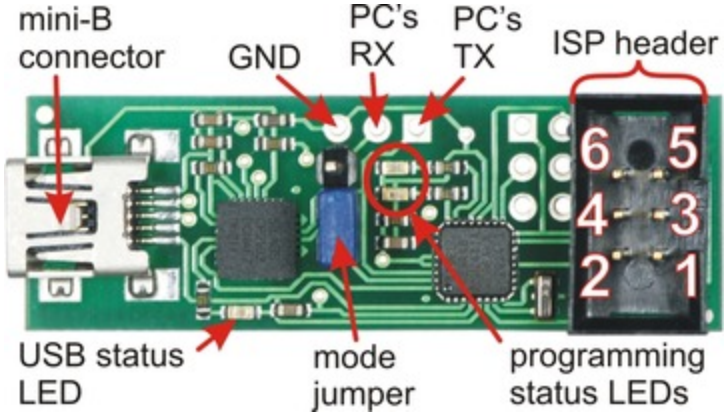
Note: The Orangutan USB programmer should be able to program all AVR microcontrollers that support paged program and EEPROM memory, but we can only support Orangutan and 3pi programming at this time. This product is not needed for the Orangutan X2, which has its own built-in AVR ISP programmer.

For a Spanish version of this document, please see **Pololu Orangutan USB Programmer Guia Usuario** [http://www.pololu.com/file/download/PololuOrangutanUSBProgrammerGuiaDeUsuario.pdf?file_id=0J136] (1MB pdf) (provided by customer Jaume B.).

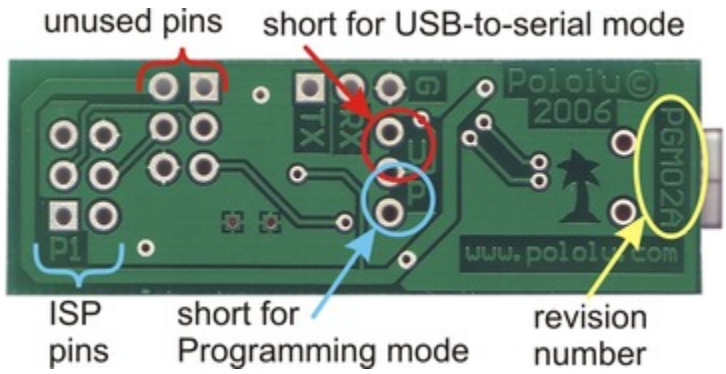
2. Contacting Pololu

You can check the **Orangutan USB Programmer page** **[http://www.pololu.com/catalog/product/740]** for additional information. We would be delighted to hear from you about any of your projects and about your experience with the Orangutan USB programmer. You can **contact us** **[http://www.pololu.com/contact]** directly or post on our **forum** **[http://forum.pololu.com/]**. Tell us what we did well, what we could improve, what you would like to see in the future, or anything else you would like to say!

3. Module Pinout and Components



Pololu Orangutan USB Programmer labeled top view



Pololu Orangutan USB Programmer labeled bottom view

USB-to-Serial Adapter Mode:
The location of the blue mode jumper determines whether the device will function as a programmer or a USB-to-serial adapter. When the mode jumper spans the two pins that are marked on the bottom of the board with a "U", the computer's RX line is connected to the pad labeled RX and the device will function as a basic USB-to-serial adapter. The RX and TX pads are labeled from the computer's perspective, so to make use of the USB-to-serial adapter you need to connect the programmer's RX pad to your target's TX pin (PD1 on the Orangutan/3pi) and the TX pad to your target's RX pin (PD0 on the Orangutan/3pi) while in USB-to-serial mode. These pads expect logic-level signals (i.e. 0 V lows, 5 V highs).

Programmer Mode:
When the blue mode jumper spans the two pins that are marked on the bottom of the board with a "P", the computer's RX line is connected to the programming microcontroller and the device will function as an in-circuit AVR ISP programmer. The computer's TX line is always connected to both the pad labeled TX and the programming microcontroller.

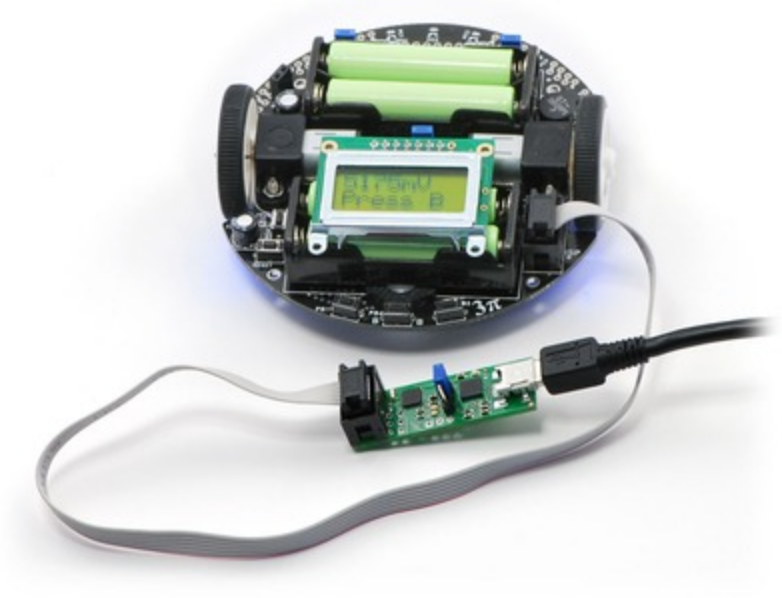
Revision Number:
There are currently two versions of the Orangutan USB Programmer: PGM02**A** and PGM02**B**. The programmer's revision number is written along the right side of the bottom of the PC board. The newer PGM02B revision has two key improvements over the original PGM02A: 1) it has the ability to accept firmware updates from Pololu (see **Section 8**) and 2) it won't let you program your target device if that device is not powered, which can help prevent you from accidentally damaging your Orangutan/3pi. Please take note of your programmer's revision number so you know which statements in this user's guide apply to your specific programmer.

LEDs:
The green USB status LED near the mini-B connector will light when the Orangutan USB programmer is connected to a personal computer and functioning properly as a serial port. If you have not installed the programmer's drivers before connecting it, this status LED will be off.

The red and green programming status LEDs near the center of the board will give you feedback when the Orangutan USB programmer is being used in programming mode. The green LED will flash every time the programmer receives a valid AVR ISP command packet from the personal computer to which it is connected. The red LED will flash every time the programmer sends information over the 6-pin ISP cable to the device being programmed. These LEDs do not do anything when the Orangutan USB programmer is in USB-to-serial mode.

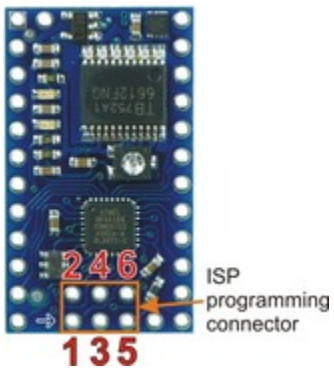
If you have programmer version **PGM02B**, you will have additional LED feedback. Every time your programmer powers up (i.e. when you connect it to your computer), the red and green status LEDs will both light for five seconds. During this period, the programmer is receptive to firmware updates; you should avoid trying to program until the initial five seconds have elapsed and the two status LEDs have turned off. Additionally, if your programmer is not connected to a target device or if your target device is not powered, the programmer's red LED will flash once per second to indicate that it will not let you program.

Connecting to Your 3pi Robot:



The Orangutan USB programmer connects to your 3pi robot via the included 6-pin ISP cable, which plugs into the 3pi's keyed ISP port located just behind the right wheel as shown above.

Connecting to Your Orangutan:



Baby Orangutan ISP pins.

Orangutan SV-168, SV-328, and LV-168 ISP pins.

Orangutan ISP pins

The Orangutan USB programmer connects to your Orangutan or Baby Orangutan via the included 6-pin ISP cable. The cable must be oriented so that the programmer's ISP header pin 1 connects to your (Baby) Orangutan's ISP

header pin 1. Unlike the Orangutan, the Baby Orangutan does not come with a shrouded header to enforce correct cable orientation; the red wire and arrow mark on the cable's ISP connector should be lined up with the arrow to pin 1 on the Baby Orangutan PCB. You will only be able to achieve this alignment by connecting to the top side of the Baby Orangutan PCB, so **be very careful not to solder your 6-pin ISP header onto the wrong side of your Baby Orangutan**!

Note: The programmer does not deliver power to the device it is programming, so your Orangutan must be turned on to be programmed. If you are using programmer version PGM02A, attempting to program an unpowered device will have unpredictable results; this might randomly change the fuse settings, which can in turn permanently disable your Orangutan (see *Fuses* in Section 5.b). If your programmer version is PGM02B, the programmer will not let you program an unpowered device (you will see the red status LED blink once per second if your target device is not powered). You must still take great care to ensure that your target device does not lose power during programming.

4. USB-to-Serial Drivers

Before you connect your Orangutan USB programmer to your computer, you must **install the driver [http://www.pololu.com/docs/0J7/]** for the programmer's CP2102 USB-to-UART bridge. Once you have successfully installed your this driver, you should see the green USB status LED near the mini-B connector lit whenever it is connected to your computer.

5. Getting Started Using Windows

After you've installed the necessary drivers, the next step is to download and install a compiler. WinAVR, located at **http://winavr.sourceforge.net/** **[http://winavr.sourceforge.net/]**, is an open source suite of software development tools for the Atmel AVR series of microcontrollers. It includes the GNU GCC compiler for C and C++. Follow the installation instructions they provide.

WinAVR alone will give you all the tools you need to start programming your Orangutan or 3pi robot, but Atmel offers AVR Studio 4, a free integrated development environment that works with the GCC C/C++ compiler. AVR Studio 4 includes a simulator and other useful tools, and supports the AVR ISP protocol used by the Orangutan USB programmer. You can **download AVR Studio 4** **[http://www.atmel.com/tools/AVRSTUDIO4.aspx]** from Atmel's website. Follow Atmel's installation instructions. Note that newer versions of AVR Studio 4 might not work with older versions of WinAVR, so we recommend you upgrade to the newest version WinAVR every time you get a new version of AVR Studio 4.

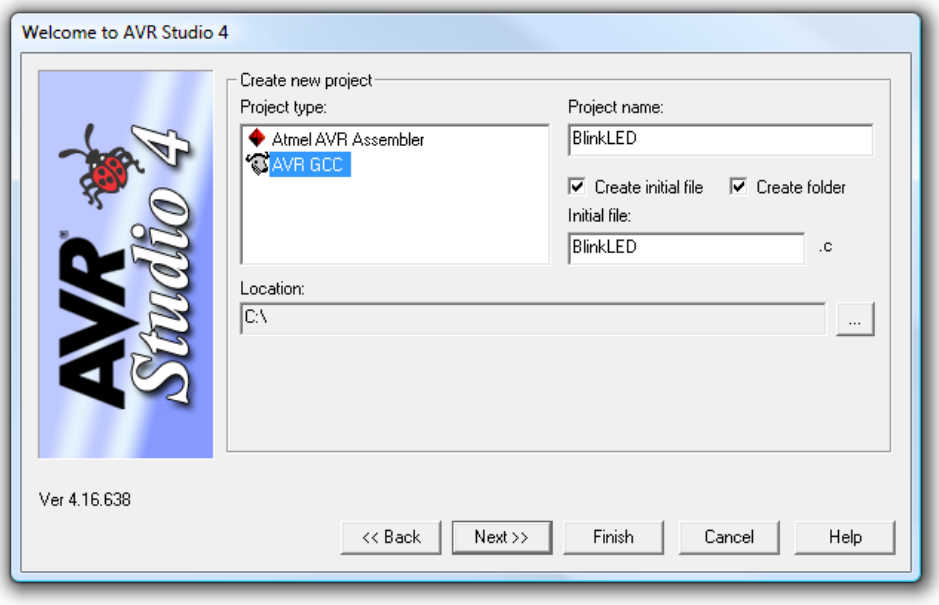
The Orangutan USB Programmer is not compatible with the newer AVR Studio 5 or Atmel Studio, but the newer **Pololu USB AVR Programmer** **[http://www.pololu.com/catalog/product/1300]** is.

5.a. Using AVR Studio 4

Here we will show you step by step how to use AVR Studio 4 to make the red user LED blink on your Orangutan, Orangutan SV-168, Orangutan SV-328, Orangutan LV-168, Baby Orangutan, or 3pi robot. Please note that this program will not work on the Orangutan X2. If you want to skip the steps that set up the LED-blinker code and jump straight into using your Orangutan USB programmer, you can download the AVR Studio project these steps would help you create and proceed straight to step 4.

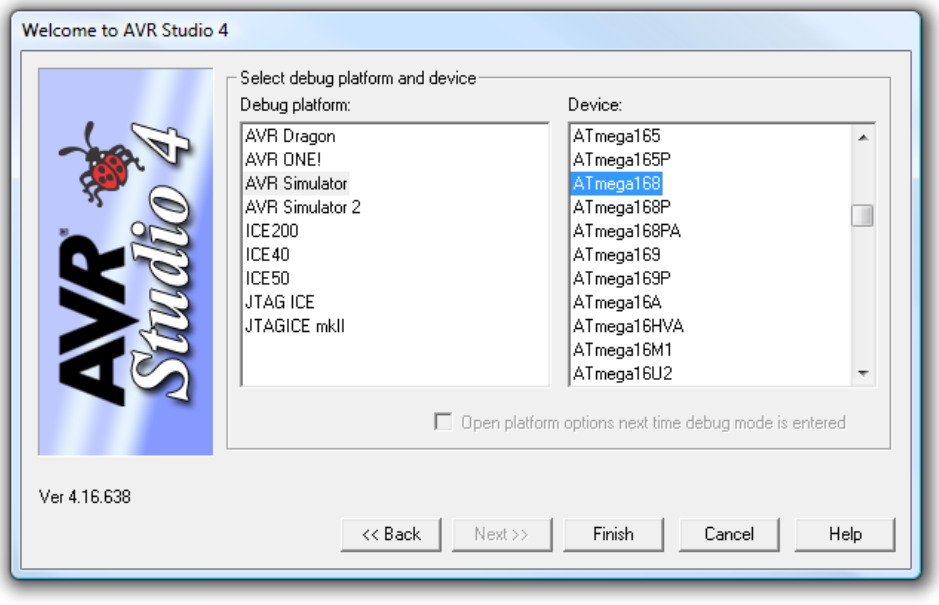
- mega48: **BlinkLED_m48.zip** **[http://www.pololu.com/file/download/BlinkLED_m48.zip?file_id=0J188]** (9k zip)
- mega168: **BlinkLED_m168.zip** **[http://www.pololu.com/file/download/BlinkLED_m168.zip?file_id=0J189]** (9k zip)
- mega328: **BlinkLED_m328.zip** **[http://www.pololu.com/file/download/BlinkLED_m328.zip?file_id=0J190]** (9k zip)

1. Open AVR Studio and click **New Project**. Select **AVR GCC** for the project type. We called our project "BlinkLED" and elected to have a folder called "C:\BlinkLED" created containing the blank file "BlinkLED.c". Click **Next >>**. DO NOT click "Finish" yet. If you do accidentally click "Finish", you will not be able to perform step 2 and will instead have to set the device by going to the "Project" menu and selecting "Configuration Options".



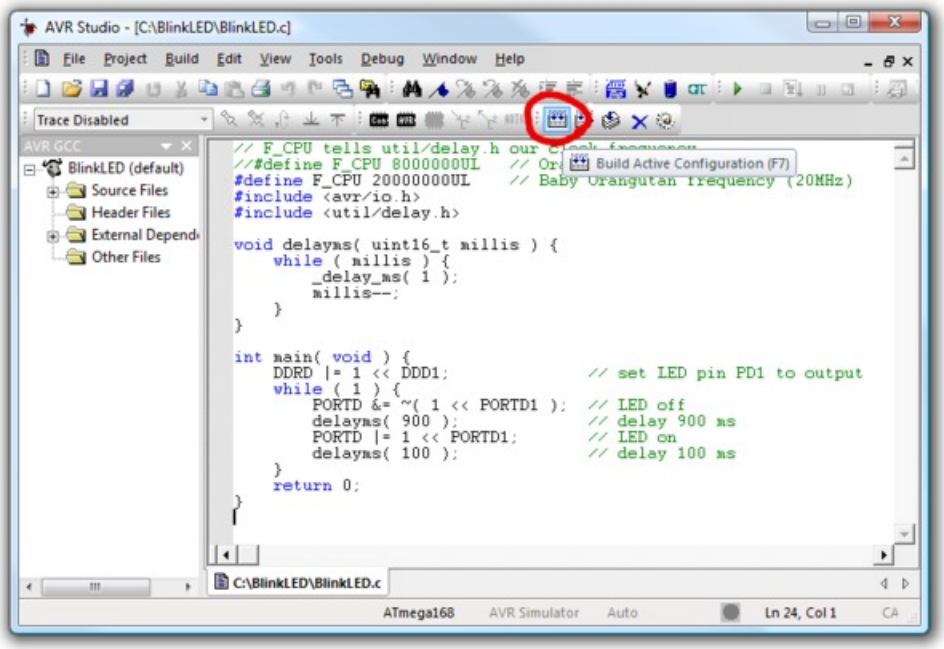
Creating a new AVR Studio project, step 1

2. Select **AVR Simulator** as the debug platform and then select the appropriate device for your Orangutan or 3pi. This will either be **ATmega48**, **ATmega168**, or **ATmega328P** depending on which chip your Orangutan or 3pi has. Click **Finish**.



Creating a new AVR Studio project, step 2

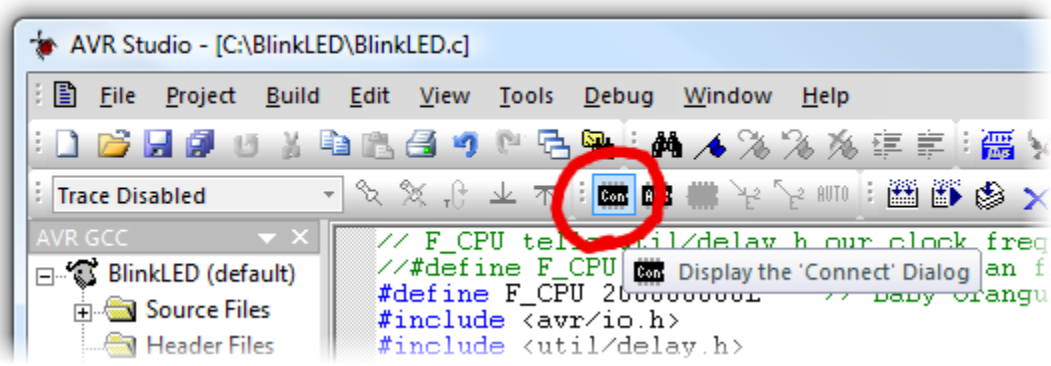
3. Write your program in BlinkLED.c as seen in the screen shot below and click the **Build** button on the toolbar (or press **F7**), or just download the zipped BlinkLED project archive for your device from the list above.



Building a project with AVR Studio

Note: You will probably want to customize this program slightly if you are working with an original Orangutan rather than a Baby Orangutan, Orangutan SV-168, Orangutan SV-328, Orangutan LV-168, or 3pi robot. **F_CPU** should be defined as the clock frequency of your target device in Hz. For the Orangutan this should be **8000000UL** (8 MHz), while for the Orangutan SV-xx8, Orangutan LV-168, Baby Orangutan, and 3pi robot this should be **20000000UL** (20 MHz). You can achieve this by commenting out line 3 and uncommenting line 2. If you don't make this change, the timing of delayms() will be off.

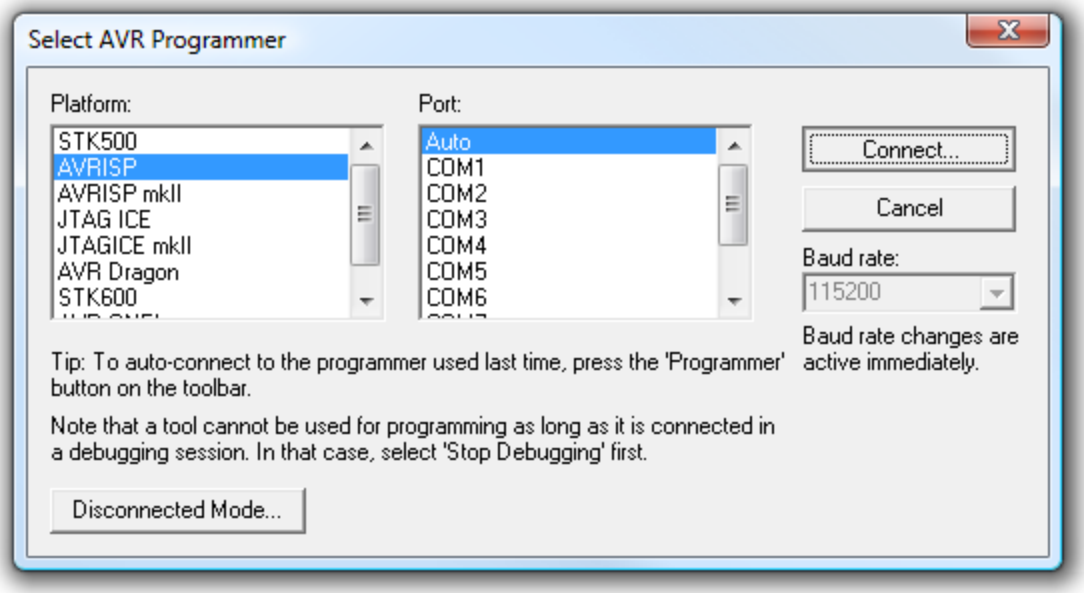
4. Make sure your Orangutan USB programmer is connected to your computer via its USB A to mini-B cable and then click the **Display the 'Connect' Dialog** button on the toolbar . You can also accomplish this by going to the "Tools" menu and selecting **Program AVR → Connect…**.



Connecting to the programmer with AVR Studio

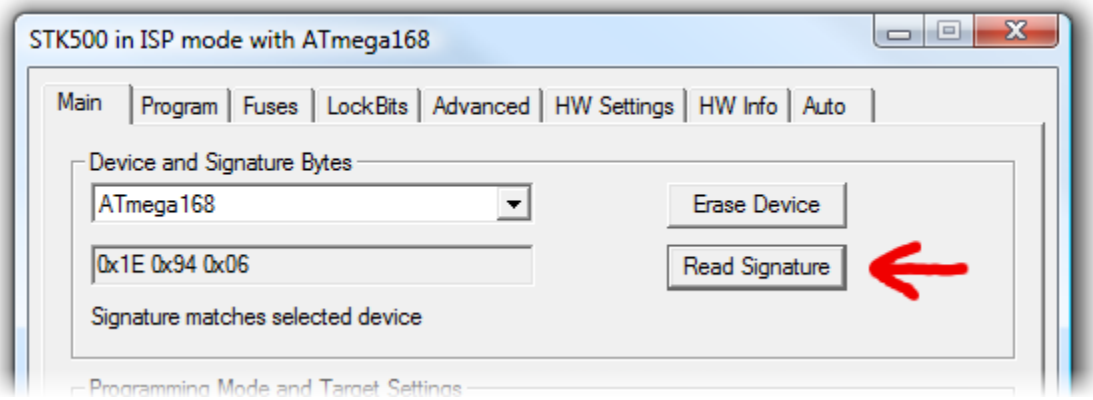
5. This will bring up a programmer selection dialog. The platform should be **AVRISP**. The Orangutan USB programmer uses AVR ISP version 2 (written as AVRISPv2), but this is not the same as AVR ISP mkII. Select the serial port of your programmer if you know what it is, or select **Auto** and it will try all the

ports until it detects the programmer. You can determine your programmer's serial port by looking in the "Ports (COM & LPT)" list of your Device Manager for "Pololu USB-to-serial adapter". Click "Connect…" to bring up the AVRISP dialog. You should see the green programming status LED flash very briefly as the dialog appears. If an error dialog appears instead of the AVRISP dialog, your computer cannot detect the programmer; please go to Troubleshooting (**Section 7**) for help identifying and fixing the problem.



AVR Studio's programmer-selection dialog

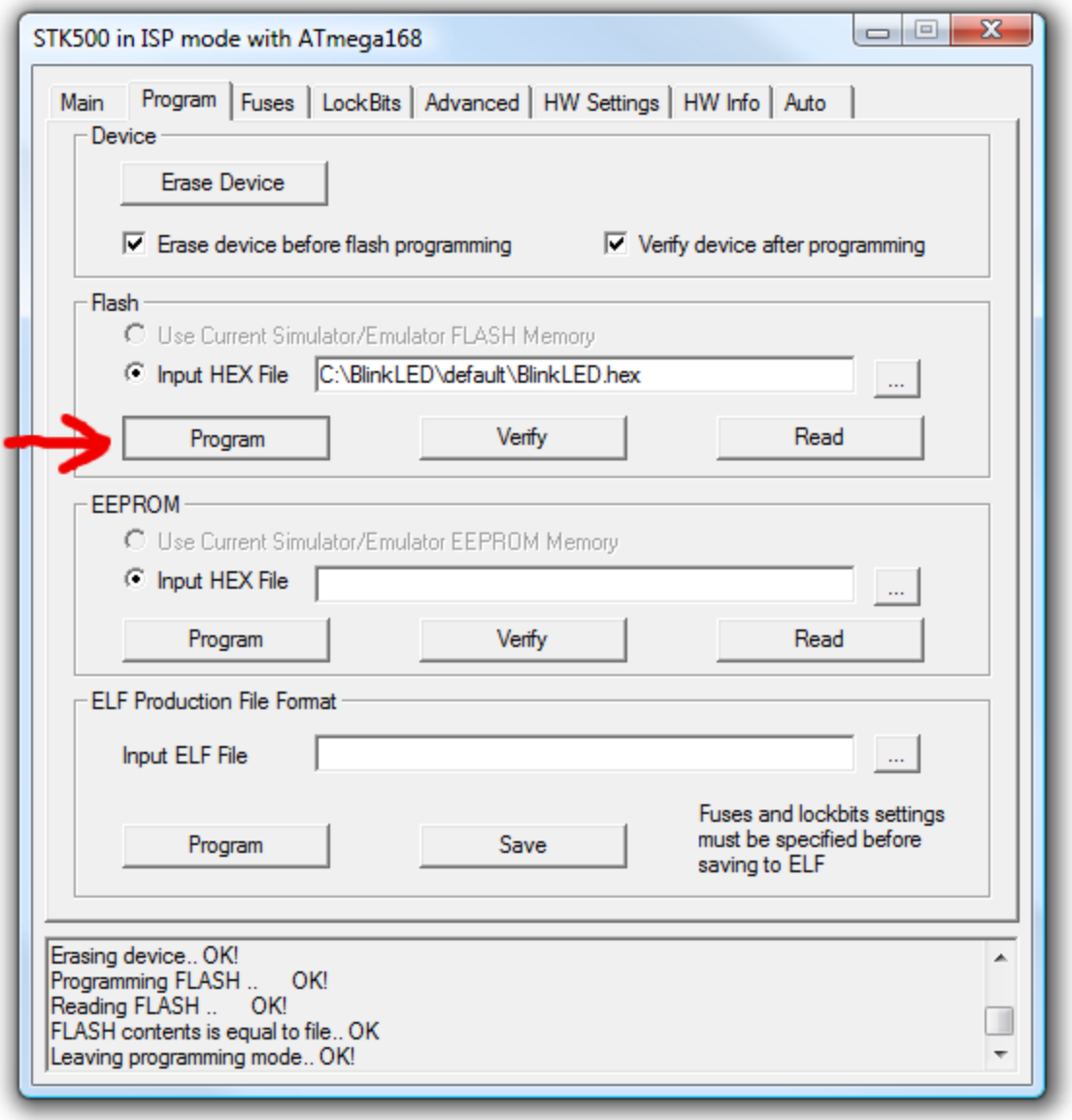
6. If you have not done so already, connect your programmer to your Orangutan or 3pi using the 6-pin ISP cable. Make sure the cable is oriented so that pin 1 on the connector lines up with pin 1 on your target device! You can test your connection by going to the **Main** tab and clicking the **Read Signature** button. This sends a command to the target microcontroller asking for its device signature. If everything works correctly, you should see *"Signature matches selected device"*. If the signature does not match the selected device, you probably have the wrong device selected (or possibly your target device is turned off). If reading the signature fails entirely, please refer to our troubleshooting advice (**Section 7**) for help getting your connection working.



Reading the device signature in AVR Studio's Main ISP tab

7. Now it is time to program your target device. Select the **Main** tab. Your **Device** should match the one you selected when you created the project: ATmega328P, ATmega168, or ATmega48. Select the **Program** tab. Your **Input HEX File** in the **Flash** section needs to be the hex file that was generated when you built your program. You can browse for this using the **"..."** button to the right of the input file text box. If you

navigate to your project's folder, you should find it as *"default\<project name>.hex"*. Click the **Program** button (make sure you click the one in the **Flash** section, not one in the "EEPROM" or "ELF Production File Format" sections!).



AVR Studio's Program ISP tab

You should see both the red and the green programming status LEDs flicker on your Orangutan USB programmer as it programs and you should see the following text appear in the text box at the bottom of the Program window:

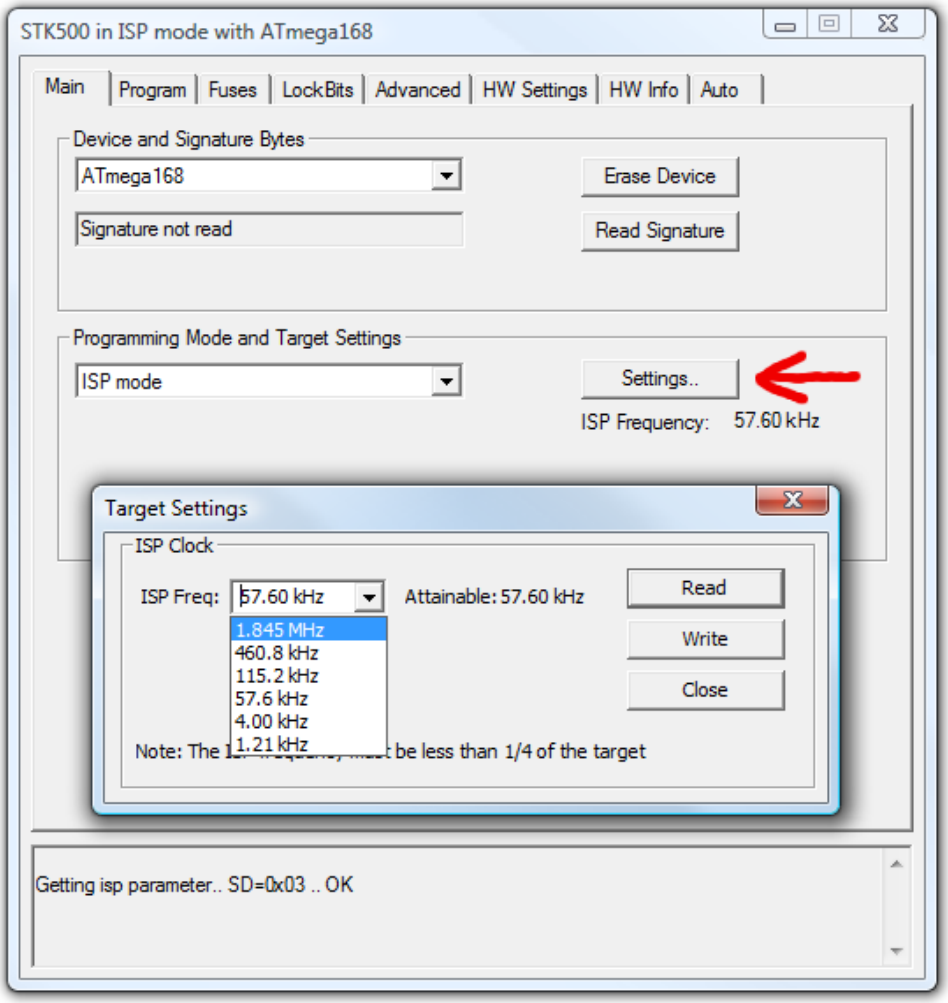
```
Reading FLASH input file.. OK
Setting mode and device parameters.. OK!
Entering programming mode.. OK!
Erasing device.. OK!
Programming FLASH ..       OK!
Reading FLASH ..      OK!
FLASH contents is equal to file.. OK
Leaving programming mode.. OK!
```

If there were no problems, the red LED on your Orangutan, Baby Orangutan, or 3pi should now be flashing! Note that if you are trying this on a 3pi robot and you haven't yet soldered in the optional through-hole LEDs, you will need to turn your 3pi robot over to see the flashing LED as the surface-mounted LEDs are located on the underside of the 3pi's PCB. If there was a problem, please go to Troubleshooting (**Section 7**) for help identifying and fixing it.

5.b. AVR Studio 4 in More Detail

This section provides a brief overview of the other tabs in the AVRISP dialog. It is rare that you will want to use anything other than the **Program** tab, but it is useful to know what the other tabs let you do for the rare occasions when you will need them.

- **Main: setting your ISP frequency**



AVR Studio's interface for setting the ISP frequency.

The **Programming Mode and Target Settings** section lets you set the frequency of the clock used when programming the target device. *This frequency must be less than a quarter of your target's clock*. It is important to note that the frequencies in the **ISP Freq** list are **not correct** when you are using the Orangutan USB programmer. Here is how the listed frequencies relate to the actual frequencies:

Listed Frequency	Actual Frequency	Allowed Target Frequency
1.845 MHz	2.5 MHz	> 10 MHz
460.8 kHz	1.25 MHz	> 5 MHz
115.2 kHz	625 kHz	> 2.5 MHz
57.6 kHz	156 kHz	> 625 kHz
4.00 kHz	3.91 kHz	> 16 kHz
1.21 kHz*	610 Hz*	> 2.5 kHz*

** This ISP frequency is so low that AVR Studio times out when it tries to program flash or EEPROM, but it can be used to program fuses and lock bits.*

You should be able to program any device running at 1 MHz or higher using the listed **57.6 kHz** setting, which is the Orangutan USB programmer's default ISP frequency. An 8 MHz device (e.g. the Orangutan) can be safely programmed using the **460.8 kHz** setting and a 20 MHz device (e.g. the Orangutan SV-xx8, Orangutan LV-168, Baby Orangutan, and 3pi robot) can be safely programmed using the fastest **1.845 MHz** setting. The two lowest frequencies exist as a safety net in case you accidentally set the clock of your target device to something under 1 MHz. The **1.21 kHz** setting is too slow to actually program your target device (AVR Studio will time out waiting for a response from your Orangutan), but it will still let you set the fuses. Be aware that if you attempt to program flash or EEPROM using the **4.00 kHz** setting, it might take somewhere on the order of five to ten minutes depending on the size of your program, so you should only use this ISP frequency as a last resort.

- **Fuses (proceed with caution!)**

Clicking on the **Fuses** tab will cause the programmer to read the fuse settings of the target device. If your programmer is not connected to your Orangutan when you select this tab, you will get an error message. Fuses allow you to configure certain aspects of your microcontroller such as boot flash size, brown-out detection level, and the clock off of which it should run (e.g. external crystal, internal oscillator). To learn more about the fuses and what they do, please see the ATmega48/88/168/328P datasheet.

Note: You can permanently disable your Orangutan by setting the fuses incorrectly. Only advanced users who know precisely what they are doing should change these fuse settings!

- **Lock Bits**

Clicking on the **Lock Bits** tab will cause the programmer to read the lock bits settings of the target device. If your programmer is not connected to your Orangutan when you select this tab, you will get an error message. The lock bits allow you to secure your microcontroller by preventing further flash writing or reading. The lock bits can be reset to a fully unlocked state by performing a chip erase (i.e. by clicking the **Erase Device** button in the **Program** tab). Lock bits are usually only important if you wish to release a product to other people without giving them access to the program it is running, or if you wish to make it slightly more difficult to overwrite a programmed chip.

- **Advanced**

You should not need to worry about Oscillator Calibration.

- **HW Info**

This tab will tell you the hardware and software versions of your Orangutan USB programmer. **Section 5.c** explains how you can configure your programmer to change these version numbers so they will match what your AVR Studio is expecting.

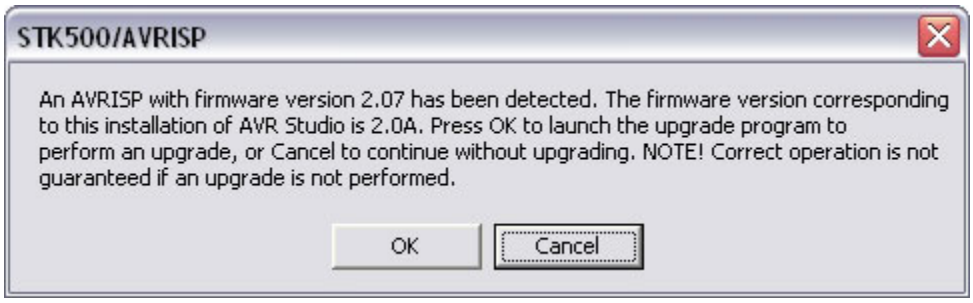
- **Auto**

  This tab lets you automatically perform a scripted set of programming actions. If you are going to be configuring many devices the same way, it might be convenient for you to do this using the Auto tab features.

5.c. Configuring Your Programmer for AVR Studio 4

Configuring your programmer's version numbers is **purely optional**. Your programmer will still function with incorrect or mismatched version numbers.

When AVR Studio connects to your programmer, it requests the programmer's hardware and software version numbers. If these versions don't match what AVR Studio is expecting, it will bring up a dialog box asking if you want to upgrade (or, if you have an old version of AVR Studio, downgrade) your programmer's firmware as shown below. In the example screen shot below, our Orangutan USB programmer has a software version of 02.07 and AVR Studio is expecting a software version of 02.0A:

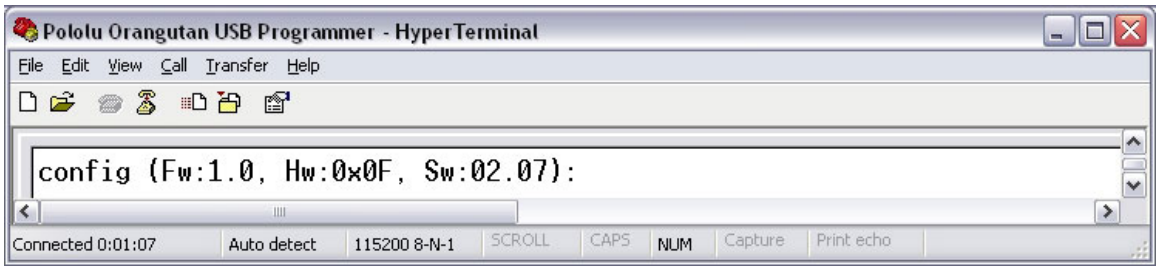


AVR Studio's AVRISP firmware version warning

The Orangutan USB programmer does not support AVR Studio's firmware upgrade/downgrade system, so this dialog is nothing more than an annoyance that we will have to dismiss every time we try to bring up the AVRISP dialog. Pressing OK will result in AVR Studio's attempting (and failing) to upgrade our programmer's firmware, so the proper response is to press **Cancel** to proceed to the AVRISP dialog. The warning can be safely ignored, so to stop AVR Studio from issuing it we can configure the programmer's software and version hardware numbers to match what AVR Studio is expecting.

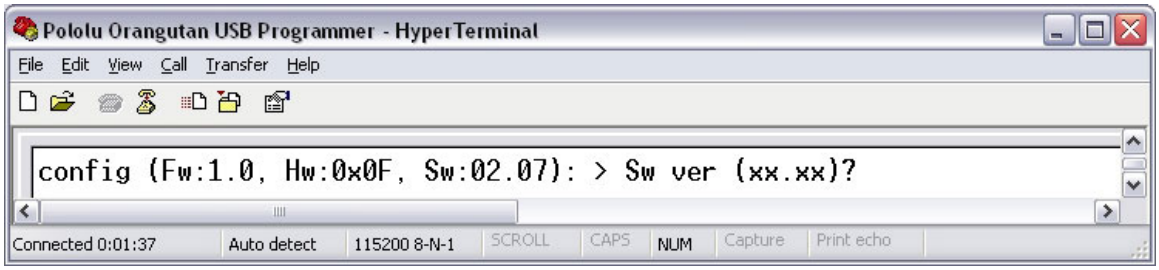
The first step in the configuration process is to determine the COM port of your Orangutan USB programmer. One way to do this is to bring up your computer's Device Manager, expand the "Ports (COM & LPT)" list, and note the COM port associated with "Pololu USB-to-serial adapter". Use your favorite terminal program to connect to this port at **115,200** baud with 8-bit characters, no parity and one stop bit. This is occasionally represented as *115200-8-N-1*. In the screen shots below we use Hyper Terminal for this.

Once the connection is made, type "config" in **all lowercase** to get into configuration mode. As you type the letters you should see the programmer echo them back to your terminal. If you make a mistake entering the string, you will see a "*" echoed and you will need to start entering the string again from the beginning. Once the string is entered correctly, the programmer will transmit its firmware, hardware, and software version numbers. The firmware version number is the version of the Pololu firmware that your programmer is running; this is not configurable. The software version is the version sent to AVR Studio when it first connects to your programmer.

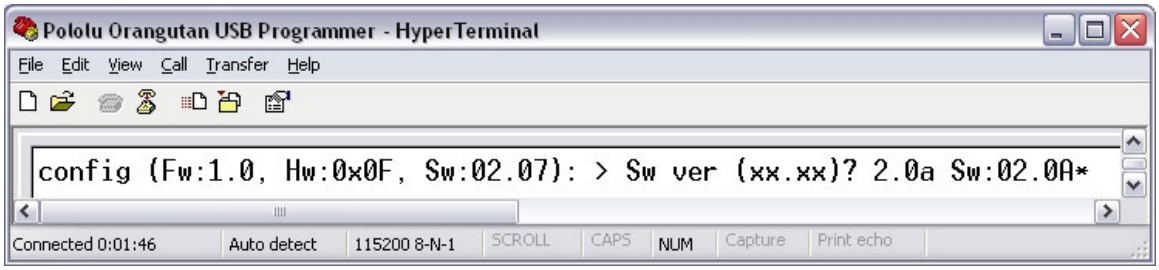


here we have transmitted: ***config***

Once the versions are transmitted, the programmer waits for a configuration command. The two valid commands are “s”, for software version configuration, and “h”, for hardware version configuration. The screen shots below show us configuring our programmer’s software version to be 2.0A, which will match what our AVR Studio expects.



here we have transmitted: ***s***



here we have transmitted: ***2.0a***

After the final character of the software version is sent, the programmer will echo back its new software version. If you have sent the version correctly, the version echoed should be what you sent. If you have not sent it correctly, the version echoed should be the software version it had before you started the configuration.

5.d. Using AVRDUDE

It is also possible to program your Orangutan using AVRDUDE, which is included in the WinAVR package. To send the blinking LED program to your Orangutan, you would type something similar to following into a command console:

```
cd C:\BlinkLED\default
avrdude -p m168 -P COM2 -c avrispv2 -e -U flash:w:BlinkLED.hex
```

The argument following the **-p** is the part number and should be **m328p**, **m168**, or **m48**. The argument following the **-P** is the port; you can use your computer's Device Manager to figure out the COM port of your Orangutan USB programmer. The programmer ID is specified using the **-c** option and should be **avrispv2**. The **-e** option results in a chip erase and the **-U** option is used for writing, reading, or verifying flash, EEPROM, fuses, or lock bits. In this example we are using -U to write BlinkLED.hex to flash. Please see the AVRDUDE documentation in *WinAVR\doc\avrdude* for more detailed information.

6. Getting Started Using Linux

Recent versions of the linux kernel include support for the Pololu Orangutan USB Programmer as part of the usb-serial driver. We have tested the following instructions under Ubuntu Linux 7.04; if you experience any problems, we recommend you upgrade to the most recent version of your distribution.

To begin working with AVRs under linux, you will need to install four software packages, which can be downloaded from their respective websites. Under Ubuntu Linux, these packages are provided in the "Universe" repository.

1. **gcc-avr:** the GNU C compiler, ported to the AVR architecture
2. **avr-libc:** a library giving access to special functions of the AVR
3. **binutils-avr:** tools for converting object code into hex files
4. **avrdude:** the software to drive the programmer

Once these packages are installed, you will be able to compile C programs for the AVR with gcc to produce hex files. We will not go into the details of writing C programs for the AVR here, but, as an example, we will show you how to use your linux computer and Orangutan USB programmer to make the user LED on your Orangutan or Baby Orangutan blink.

Download the archive that is appropriate for your device's microcontroller:

- mega48: **BlinkLED.zip [http://www.pololu.com/file/download/BlinkLED_m48.zip?file_id=0J188]** (9k zip)
- mega168: **BlinkLED.zip [http://www.pololu.com/file/download/BlinkLED_m168.zip?file_id=0J189]** (9k zip)
- mega328: **BlinkLED.zip [http://www.pololu.com/file/download/BlinkLED_m328.zip?file_id=0J190]** (9k zip)

and unpack it on your linux computer. Copy the file `BlinkLED/linux/Makefile` into the `BlinkLED/` directory. It may be necessary for you to edit this file, changing the settings at the beginning to reflect the locations where the AVR utilities were installed. Additionally, if you are using an original Orangutan mega168 (instead of a Baby Orangutan, Orangutan SV-xx8, Orangutan LV-168, or 3pi robot), you might want to edit `BlinkLED.c` by uncommenting line 2 and commenting out line 3 to account for the Orangutan's lower 8 MHz clock frequency.

At this point, you should be ready to compile the example program and load it onto the Orangutan. Plug in the programmer and type **make**.

You should see output like this:

```
/usr/bin/avr-gcc -g -Os -Wall -mcall-prologues -mmcu=atmega168   -c -o BlinkLED.o BlinkLED.c
/usr/bin/avr-gcc -g -Os -Wall -mcall-prologues -mmcu=atmega168 BlinkLED.o -o BlinkLED.obj
/usr/bin/avr-objcopy  -R .eeprom -O ihex BlinkLED.obj BlinkLED.hex
/usr/bin/avrdude -c avrispv2 -p m168 -P /dev/ttyUSB0 -e

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.02s

avrdude: Device signature = 0x1e9406
avrdude: erasing chip

avrdude: safemode: Fuses OK

avrdude done.  Thank you.

/usr/bin/avrdude -c avrispv2 -p m168 -P /dev/ttyUSB0 -U flash:w:BlinkLED.hex

avrdude: AVR device initialized and ready to accept instructions
```

```
Reading | ################################################## | 100% 0.02s

avrdude: Device signature = 0x1e9406
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed
         To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "BlinkLED.hex"
avrdude: input file BlinkLED.hex auto detected as Intel Hex
avrdude: writing flash (160 bytes):

Writing | ################################################## | 100% 0.07s

avrdude: 148 bytes of flash written
avrdude: verifying flash memory against BlinkLED.hex:
avrdude: load data flash data from input file BlinkLED.hex:
avrdude: input file BlinkLED.hex auto detected as Intel Hex
avrdude: input file BlinkLED.hex contains 160 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 0.05s

avrdude: verifying ...
avrdude: 160 bytes of flash verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.

rm BlinkLED.o BlinkLED.obj
```

This output indicates the Orangutan was successfully programmed, and the LED on the board should begin to blink. If programming was not successful, please take a look at our troubleshooting advice (**Section 7**).

7. Troubleshooting

If Your Computer Fails to Connect to the Programmer

- Make sure your programmer is connected to your computer via a USB A to mini-B cable. If it was previously working and has since stopped, try cycling the power by unplugging it from your computer and then reconnecting it.
- Make sure you have installed the drivers the Orangutan USB programmer needs to operate.
- Is the programmer's green USB status LED on? This is the single LED on the same side of the board as the mini-B connector. If this LED is not on, you do not have a valid USB-to-serial connection between your programmer and your computer.
- If you are running Windows, can you see your programmer listed among your Device Manager's COM ports? If you expand your Device Manager's "Ports (COM & LPT)" list, you should see your programmer appear as "Pololu USB-to-serial adapter".
- If you are running linux, examine the system messages generated when the programmer is connected, using the command "dmesg".
- Your computer will only let one program have a given COM port open at a time. If you are connected to your Orangutan USB programmer's COM port using another program, such as Hyper Terminal or a second instance of AVR Studio, you will not be able to connect to it with your programming software. Please make sure you don't have any terminal programs connected to your programmer, and if you have multiple versions of AVR Studio running, make sure that you have closed the "AVRISP" programming dialogs in all of them. When the AVRISP dialog is open, that instance of AVR Studio has an open serial connection to your programmer.
- If you are using AVR Studio, try connecting to your programmer's specific COM port instead of selecting the "Auto" option, which attempts to locate the port automatically.
- Is your programmer's blue mode jumper selecting for programming mode? If the mode jumper is missing or is selecting USB-to-serial adapter mode, your computer will not detect it as a programmer.
- If none of the above work, try connecting to your programmer using a terminal program to verify that it's alive. Please see the Configuring Your Programmer for AVR Studio section for instructions on how you can interact with your Orangutan USB programmer using a terminal program.

If Your Programmer Has Problems Connecting to Your Target Device

- The most common cause for this problem is an incorrect connection between your programmer and your target device. If the ISP pins are misaligned between your programmer and your target device, the two will not be able to communicate. Please make sure that the ISP pins as numbered in the Module Pinout & Components section are correctly connected between your Orangutan and your programmer (i.e. 1 goes to 1, 2 goes to 2, etc.).
- Your target must be powered for programming to work. Please make sure that your Orangutan has power and is turned on.
- Your programmer's ISP frequency must not be more than a quarter of your target device's clock frequency. If you are having trouble communicating with your target device, try lowering the ISP frequency using the Board tab of AVR Studio's AVRISP dialog. For more information about ISP frequency, see the Board section.
- There may be a problem with your target device. It's possible to kill a device with a static shock, by incorrectly connecting power, or by programming the fuses incorrectly. There could also be a short or cut

trace somewhere on your target device. The ideal way to test for this would be to try programming a different device with your Orangutan USB programmer (or, conversely, try using a different programmer to program your target device). If this is not an option, try verifying that your target device is still functional and perform some continuity tests to check for shorts or disconnections on the ISP programming lines. Don't forget to check the 6-pin ISP cable for shorts as well.

If none of the above troubleshooting suggestions help, please **contact us [http://www.pololu.com/contact]** for support.

8. Updating Your Programmer's Firmware

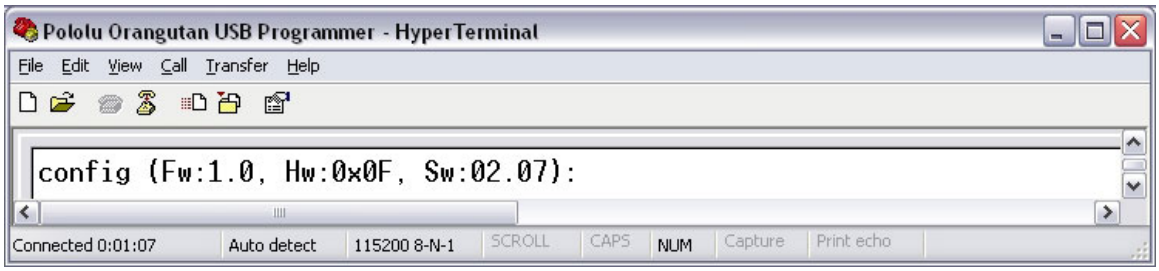
Note: Only programmer revision **PGM02B** supports firmware upgrades. If you have revision PGM02A, this section does not apply to your product.

On August 20, 2008 we released a firmware update (firmware version **1.4**) for the Orangutan USB programmer revision PGM02B that fixes a bug where the programmer could be left trying to drive some of its programming lines high if programming failed.

On July 31, 2008 we released a firmware update (firmware version 1.3) for the Orangutan USB Programmer revision PGM02B that causes the programmer to immediately abort if the target device loses power while programming is in progress, which makes it much less likely for the AVR to be corrupted. It also allows the programmer to work better with our **Orangutan SV-xx8** **[http://www.pololu.com/catalog/product/1225]**, **Orangutan LV-168** **[http://www.pololu.com/catalog/product/775]**, **Baby Orangutan B** **[http://www.pololu.com/catalog/product/1216]**, and **3pi robot** **[http://www.pololu.com/catalog/product/975]**, which all have motor driver control lines that double as programming lines. Version 1.2 could cause motors connected to these devices to briefly kick during programming. Version 1.3 will not drive motors during programming.

Note that version 1.2 (and hence also version 1.3) fixed a serious bug in the programmer's version 1.1 firmware. This bug causes the programmer to pull too much current while connected to the target device, which in turn can damage the programmer.

You can determine your firmware version number by following the steps listed in **Section 5.c** (Configuring Your Programmer for AVR Studio). The programmer used in the following screen shot has firmware version 1.0 ("Fw: 1.0"):



From this step, you can type **x** to exit without changing any version numbers.

If you have firmware version 1.1, 1.2, or 1.3, please download this file:

pgm02b_firmware_v1.4.pgm **[http://www.pololu.com/file/download/pgm02b_firmware_v1.4.pgm?file_id=0J126]** **(47k pgm)**

and follow the firmware update procedure detailed in the rest of this section.

8.a. Upgrade Preparations

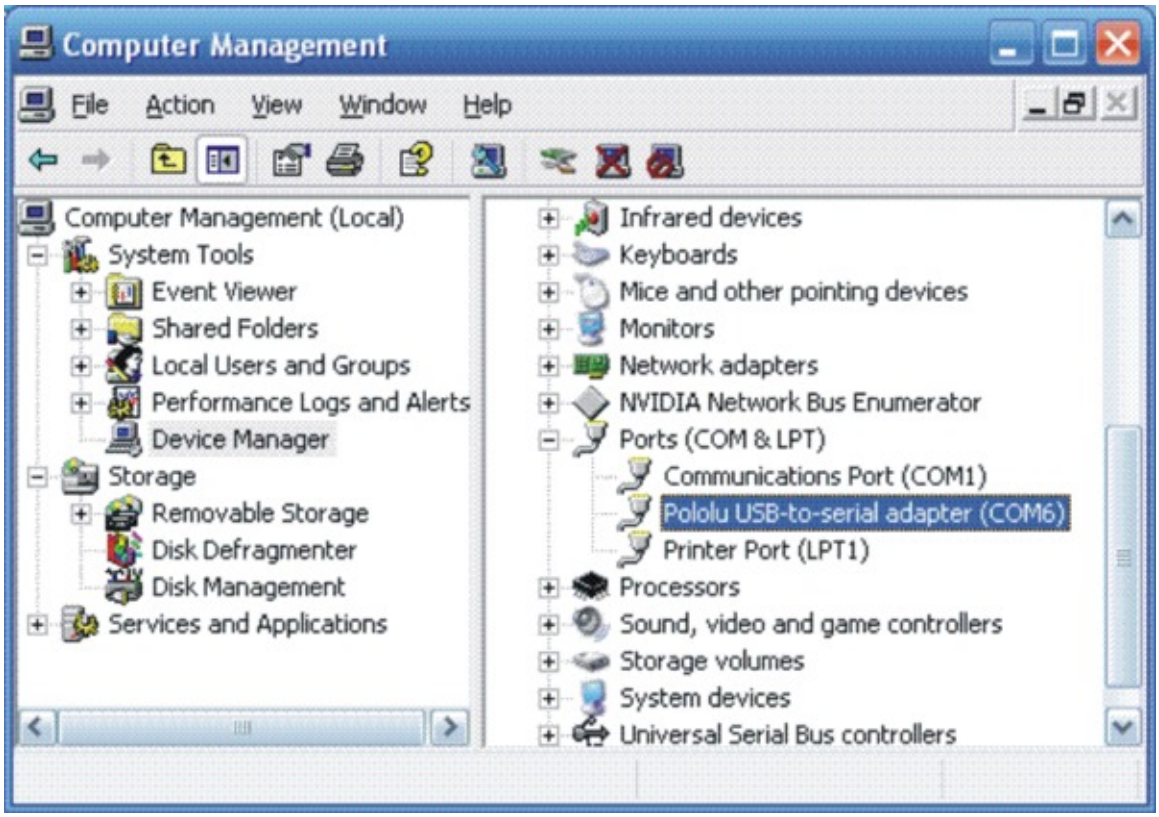
1. Install a suitable terminal program

To perform the firmware update, you will need a terminal program capable of sending binary files. We recommend Tera Term Pro v2.3, available for free at **http://hp.vector.co.jp/authors/VA002416/teraterm.html [http://hp.vector.co.jp/authors/VA002416/teraterm.html]**. Installation instructions are included in the downloaded .zip file.

2. Close any programmer serial port connections

Close any existing COM port connections you have that might interfere with the upgrading process. For example, if you have AVR Studio open, make sure the AVRISP programming window is closed as this window ties up the COM port your programmer is using.

3. Determine your COM port

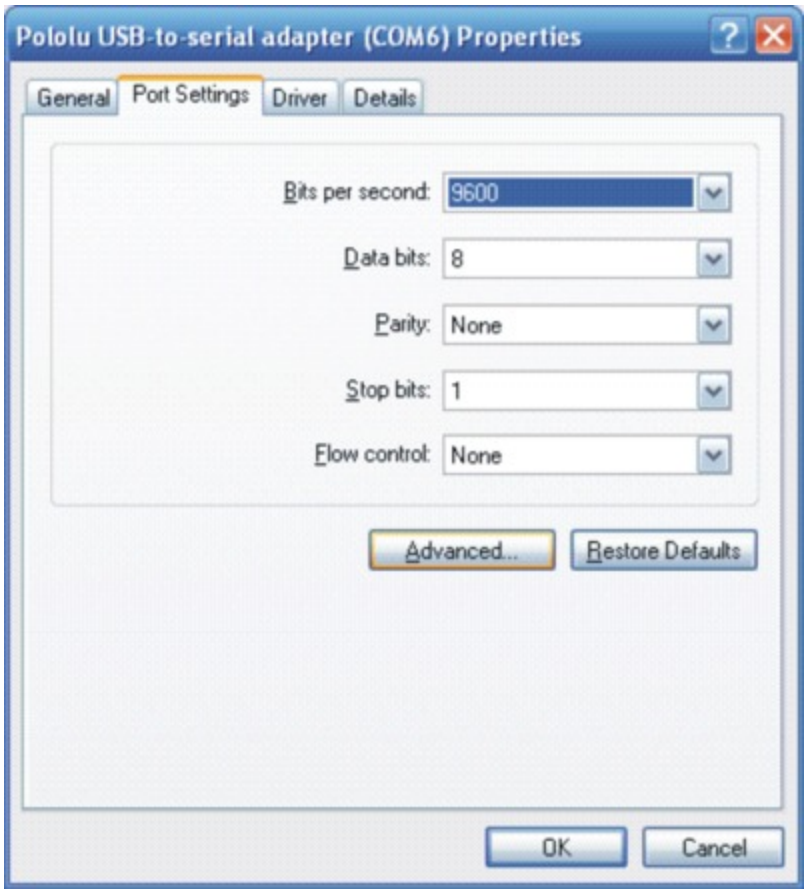


Using the Windows device manager to identify the programmer's COM port

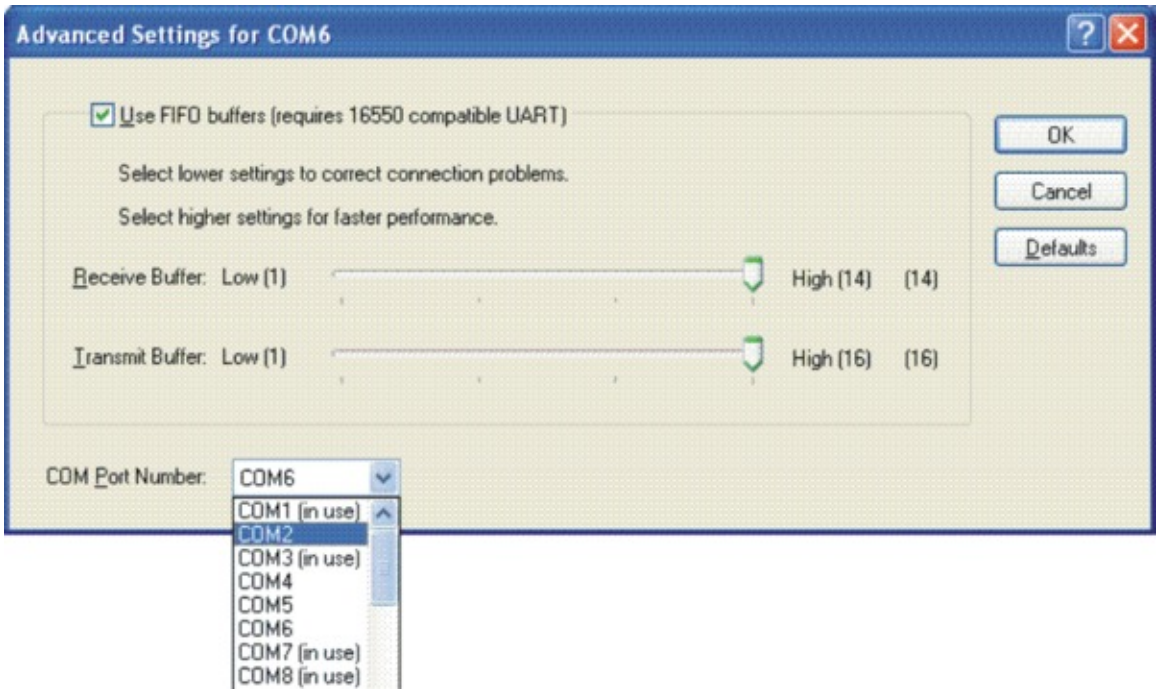
Connect your programmer to your computer and bring up your computer's device manager. One way to accomplish this is through the hardware tab of the System control panel. Another way is to right-click on "My Computer" and select "manage," which should bring up a window like the one shown above. Select "Device Manager" on the left and then expand the "Ports (COM & LPT)" list on the right. Make note of the COM port associated with "Pololu USB-to-serial adapter". (If you have another Pololu device that uses the CP2102 USB adapter, you might see a different device description.) In our example below you can see the port is COM6. If your port is COM4 or less, or if your terminal program can use higher port numbers, please move on to **Section 8.b**; otherwise, continue with the following step.

4. Make sure your port is COM1 through COM4 (TeraTerm only)

Tera Term cannot connect to COM6, so we need to change the COM port number. Right-click the “Pololu USB-to-serial adapter (COMx)” and select “Properties”. Select the “Port Settings” tab of the resulting dialog box and click the “Advanced…” button.



At the bottom of the next dialog box is a drop-down list labeled “COM Port Number”. When you select this list you can see all the ports available to you. You need to select a port from COM1 to COM4 that is not marked as “(in use)”. In the example below, valid choices would be COM2 or COM4; we select COM2.



If COM1 through COM4 are all in use, you have three options:

1) find a way to free up one of those ports by disconnecting something

2) find another terminal program that will let you send binary files to ports greater than COM4

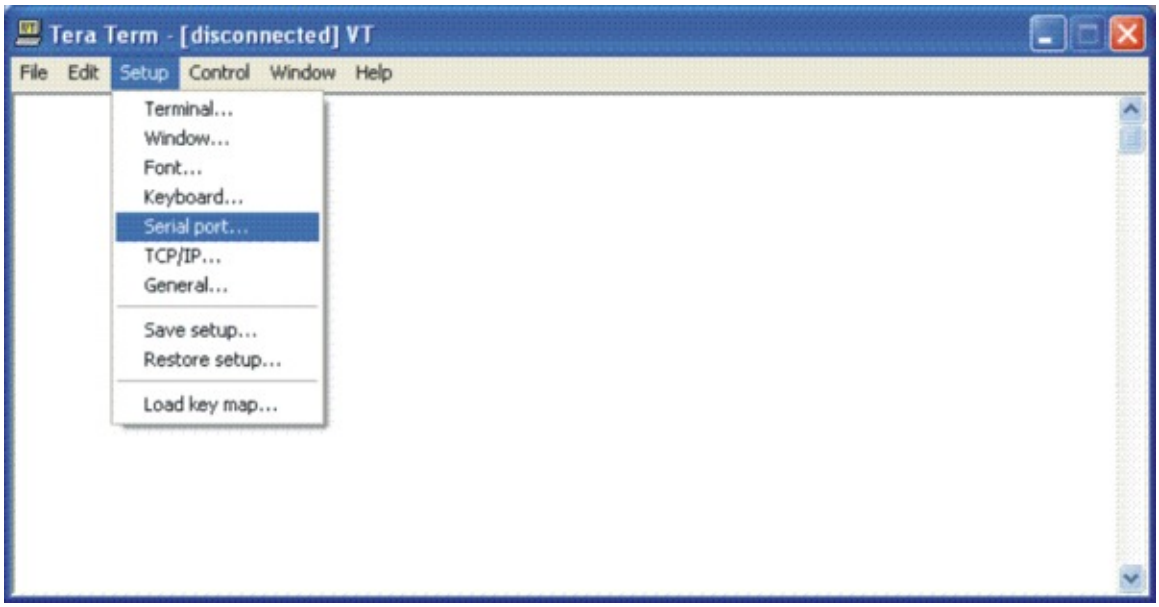
3) reassign your programmer to an “in use” port that you know nothing is currently connected to.

Note that if you choose option 3, you might run into problems in the future if you ever simultaneously connect your programmer and the other device that is configured to use that port. Once you have chosen your new port, click OK on both the Advanced and Properties windows and close the Computer Management window. If you re-open Computer Management window, you should see your serial adapter mapped to your newly selected port.

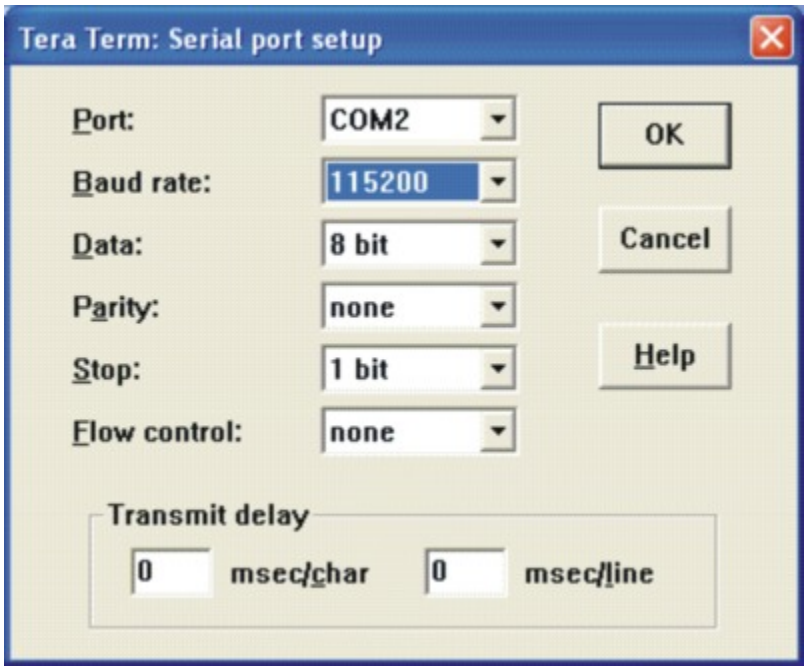
8.b. Uploading the New Firmware

1. Run Tera Term Pro v2.3

You will be greeted by a “TeraTerm: New connection” dialog box. Click cancel, then go to the “Setup” menu and select “Serial port…”.



Configure the settings as shown below, substituting your own COM port. The baud rate should be 115200 with 8-bit data, no parity, one stop bit, and no flow control. You do not need any transmit delays. The only settings you should have to change from their default values are the port and the baud rate. **Do not click OK yet.**



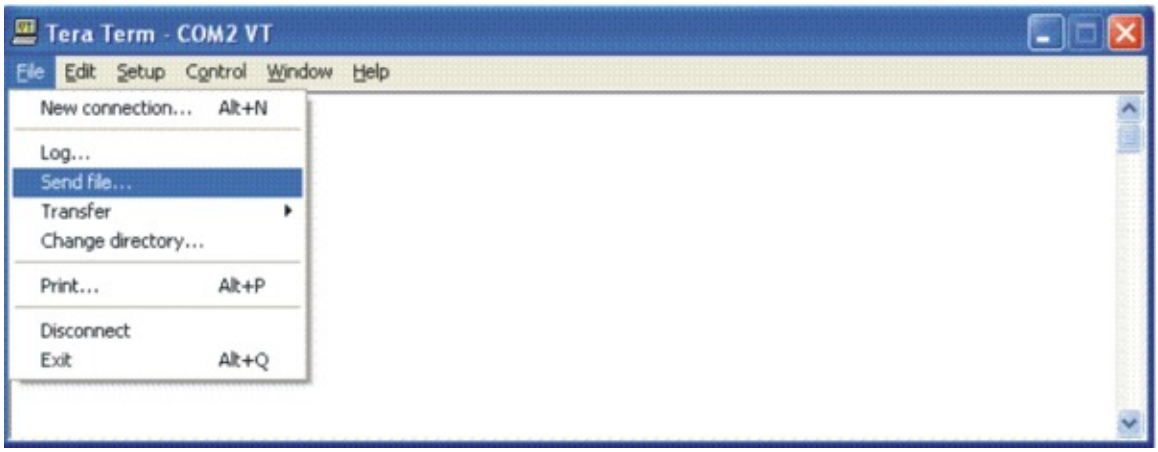
2. Enter the programmer's firmware update mode

When you connect your programmer to your computer via the USB cable, you should see both the red and green status LEDs light for approximately five seconds. During this time, the programmer is waiting for the proper input sequence that will put it into firmware update mode. The procedure for entering firmware update mode is as follows:

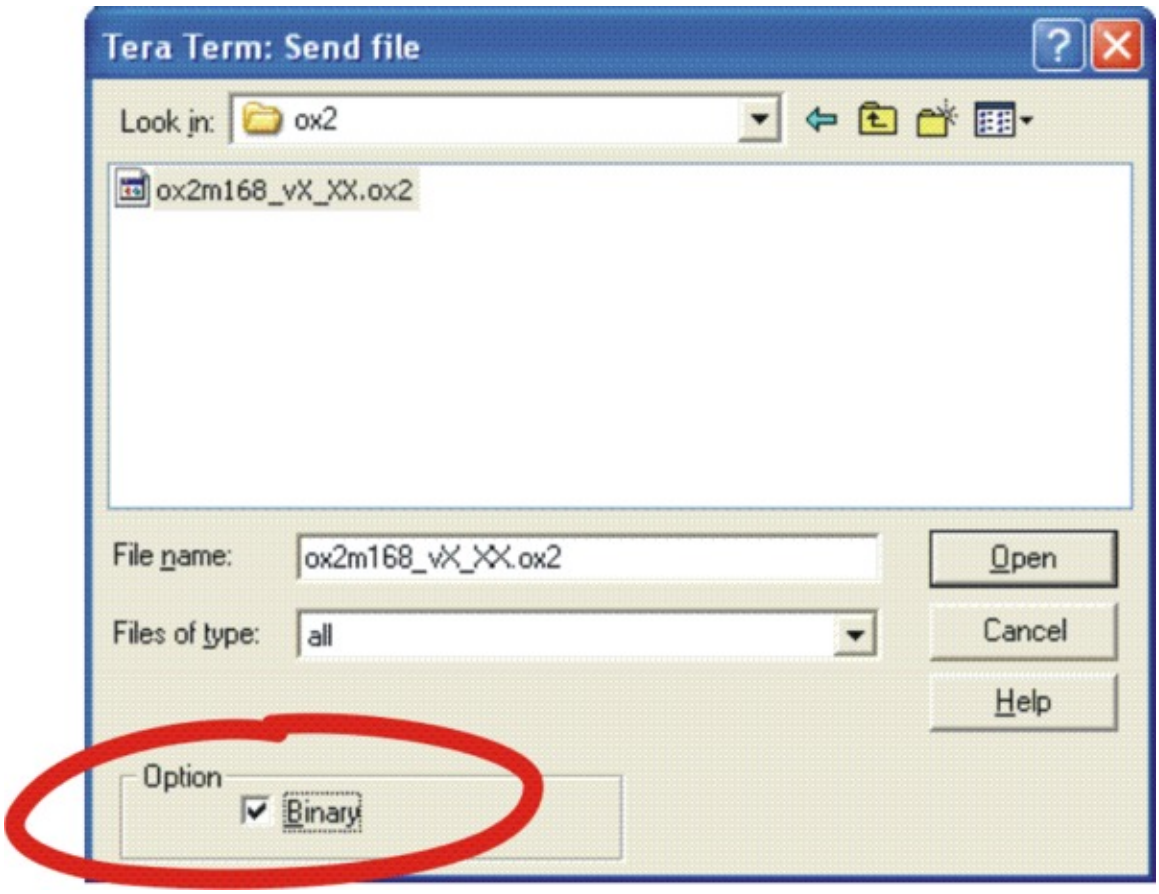
- Bring up the Tera Term serial port setup dialog as described in part 1 above, but do not connect yet.
- Plug your programmer into your computer via the USB cable and wait for the green **USB** status LED to light. It usually takes around 0.5 s for the USB connection to be established.
- Once the green USB status LED is lit and while both red and green programmer status LEDs are still lit, click **OK** in the Tera Term serial port setup dialog to establish a connection and type **fwbootload**. This string must be in all lowercase. With every successfully entered letter, the 5-second timer resets, so the biggest difficulty is typing that first "f" before the first 5-second period expires and the two status LEDs turn off.
  As you type the string, you will see the uppercase version of the letters you type echoed back, so as you type "fwbootload", you will see "FWBOOTLOAD". If you enter a letter incorrectly, a "?" will be echoed back to you and both status LEDs will turn off immediately. If this happens you will need to disconnect from Tera Term, unplug your programmer, and repeat the process again.
  If you enter the string successfully, you will see the bootloader version number echoed back to you (probably "1.2"). Note that this bootloader version number is not the same as the the programmer's firmware version number. This version number is the version of the small piece of code (called a "bootloader") that is reprogramming the programmer's flash.
- At this point you can type **x** to exit firmware upgrade mode or **s** to load the new firmware. If you type **s**, there will be a brief delay as the current firmware is erased, after which you will see **S** echoed back. Now the programmer is waiting for you to upload a binary data file containing the new firmware. This data file should have a **.pgm** extension.

3. Loading the new firmware

From the “File” menu, select “Send file….”



Navigate to the directory that has the .pgm file you wish to upload and select the file. Before you click “Open,” **make sure you have checked the binary checkbox option in the lower left part of the window**. If you fail to check this box, the upload will fail.



When you click “Open”, you should see periods streaming across your terminal window . Each period represents the successful transmission of a packet. You should also see the red status LED below the reset button flickering as the firmware update progresses. When the upload is complete, the programmer will transmit ***SUCCESS** to indicate success. If there is a problem, the programmer will transmit either **STARTERR** or **CHKSUMERR**, most likely followed by a stream of *****'s and a few **c**'s. The most likely cause for such a problem would be the failure to transmit the firmware upgrade file as a binary file. If repeating the steps in this section again do not lead to a successful firmware update, please see **Section 2** for information on how to contact us for support.