



NetBurner FPGA Blade Board (NBPKX500-100CR) Manual

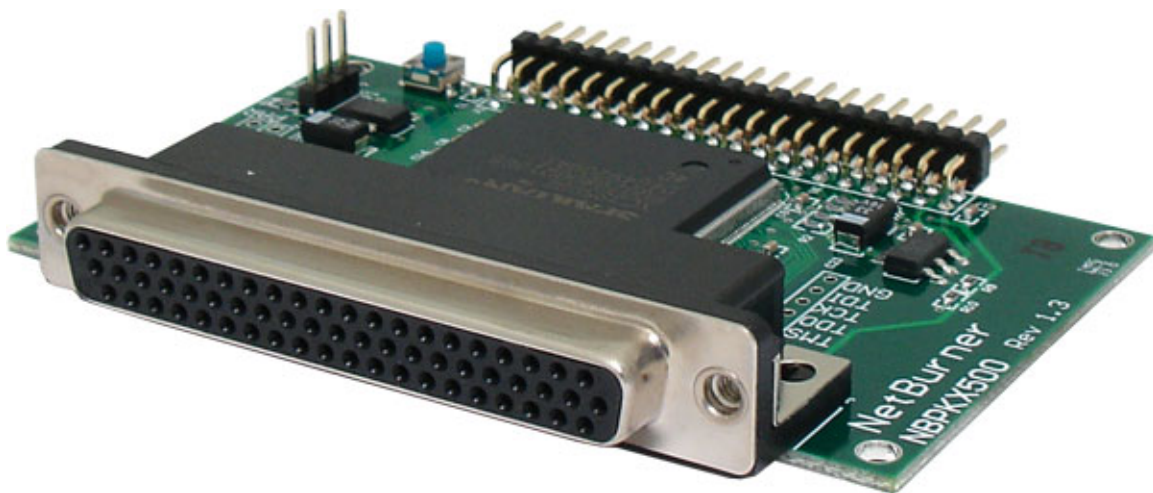


Table of Contents

1	Introduction.....	3
1.1	Overview.....	3
1.2	Features	3
1.3	Requirements	4
1.4	Design Outline	4
2	Programming the FPGA	4
2.1	Create the Verilog and User Constraints Files.....	4
2.2	Compile the source files using Xilinx tools.....	4
2.3	Making a bitstream file	9
2.4	Convert the bit file to a cpp to be used by your project.....	9
2.4.1	Using NBEclipse.....	9
2.4.2	Using the command line	10
2.5	Add the source files to your project.....	10
2.6	Call LoadCode on startup	11

Revision History

Rev.	Date	Comments
1.0	6/2010	Initial release
1.1	2/2011	Revised introduction

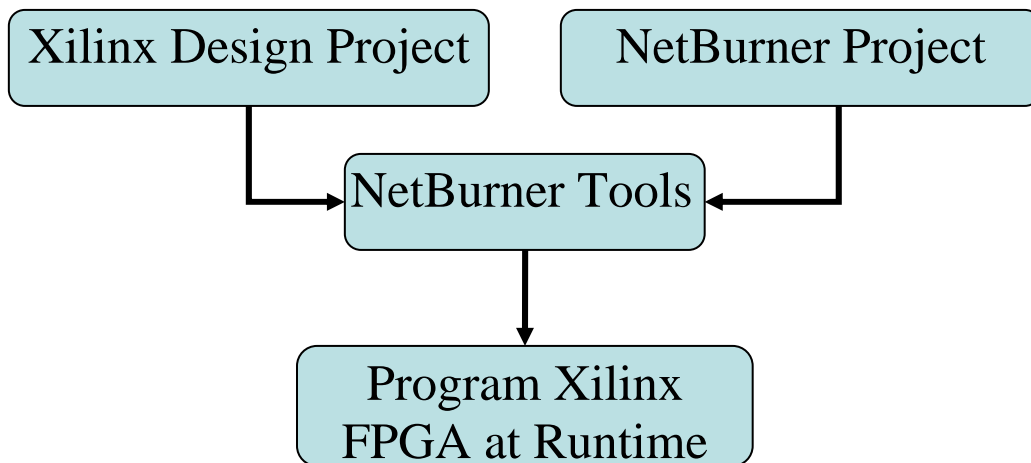
1 Introduction

Special Note:

This documentation will detail the process of creating, building, and loading code to an FPGA running on the FPGA Blade Board. It is not an instructional document on Verilog or VHDL programming, and only covers how to load your source into the NetBurner tools. If you are using the Xilinx Design Suite, you must download the software directly from Xilinx.

1.1 Overview

The FPGA blade board is development hardware that enables a developer to quickly interface a PK70 with a Spartan 3E FPGA. Using NetBurner tools and this guide, learn how Xilinx binary files can be loaded directly in to a NetBurner project. The NetBurner project can then load the binary file on to the FPGA at runtime, keeping your application and FPGA blade board in sync.



1.2 Features

- Hardware layout featuring access to the Xilinx Spartan 3E FPGA
- Parallel interface between the Spartan 3E and a NetBurner PK70 device
- High Density 62 pin connector (DB-62HD)
- Program the FPGA anytime with the JTAG connector or at runtime from a NetBurner application
- Code examples demonstrating how to load an FPGA binary file at runtime

1.3 Requirements

The following hardware and software is required to use this guide

- NetBurner NNDK version 2.4 or greater
- NetBurner PK70 device
- NetBurner FPGA Blade Board
- Xilinx ISE Design Suite

1.4 Design Outline

Creating a project for the Xilinx FPGA consists of three major steps:

1. Compiling the Verilog source in to a BIT file
2. Converting the BIT file in to a C++ source file. This file can be used by any NetBurner project.
3. Call a special function which loads the FPGA source from the NetBurner application.

This document will go over the steps needed to be taken to complete this process. Source files will be provided to create a simple GPIO demo.

2 Programming the FPGA

2.1 Create the Verilog and User Constraints Files

The first step in building for the FPGA blade board is to create your Verilog and User Constraints file. An example Verilog source file and user constraints file has been included with the board. To follow along in this Users Guide, you should use these examples, XilinxBladeGpio.v and XilinxBlade.ucf.

2.2 Compile the source files using Xilinx tools

To compile the source files, you must have the Xilinx tools installed. In this example, I used the Xilinx ISE Design Suit, 12.1. Different versions of the tools may produce different results.

To start, click on the start menu and select Xilinx ISE Design suite 12.1 → ISE Design Tools → Project Navigator. You should now be at the main ISE screen (Figure 2.2-1)

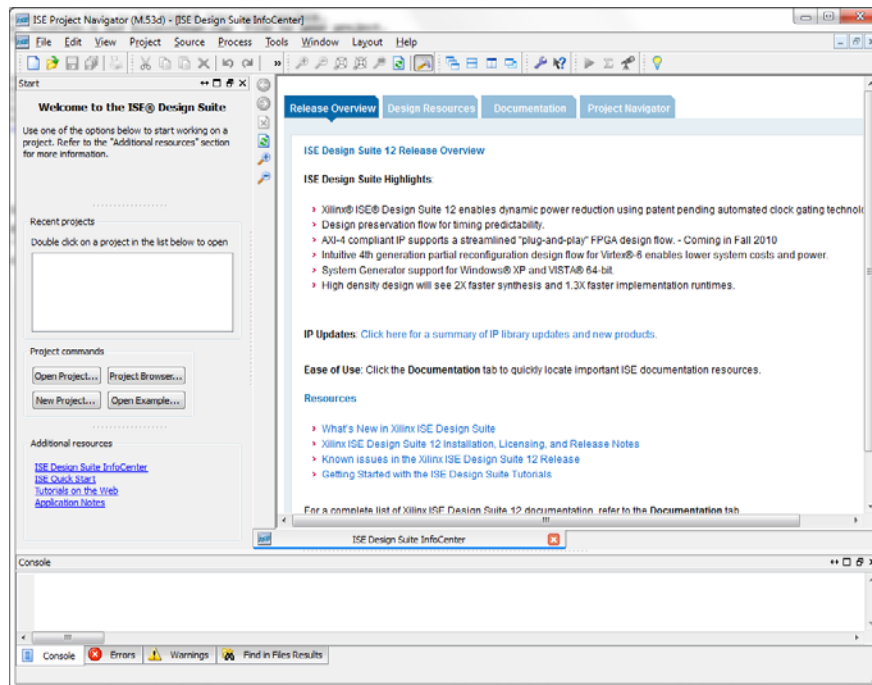


Figure 2.2-1

Click on the New Project button to start a new project (Figure 2.2-2)

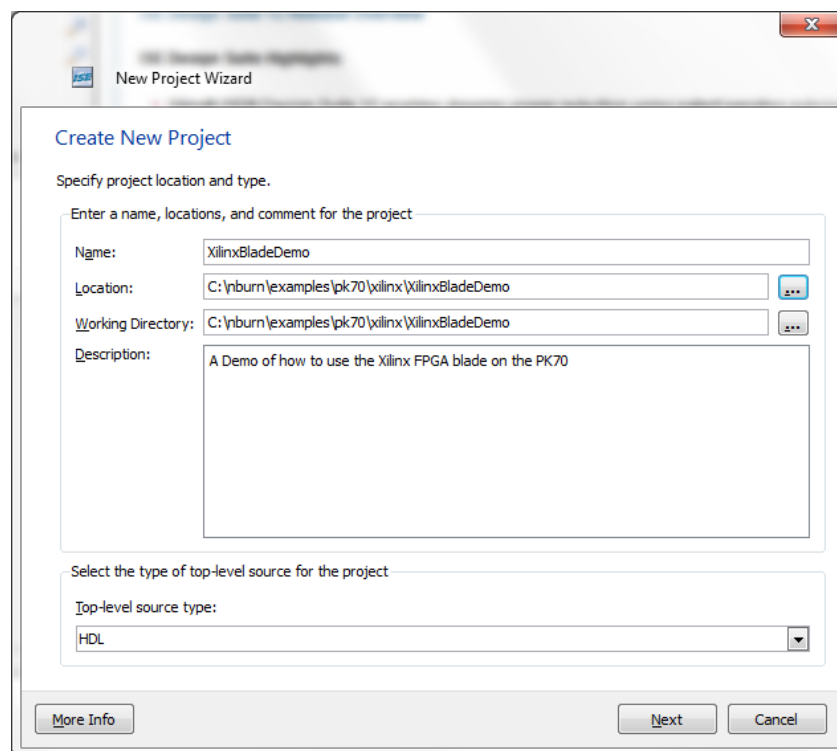


Figure 2.2-2

After naming your project, click next to bring up the project settings. (Figure 2.2-3) Fill in the device type as XC3S500E in the PQF208 package.

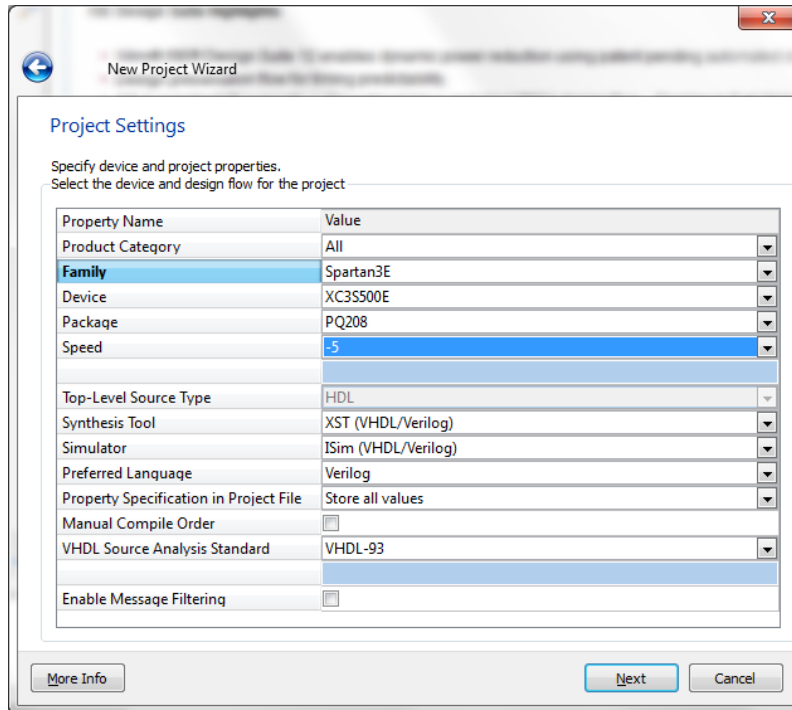


Figure 2.2-3

Click next to proceed to the project summary page. (Figure 2.2-4) Verify that the proper device and package is selected and click finish to create the project.

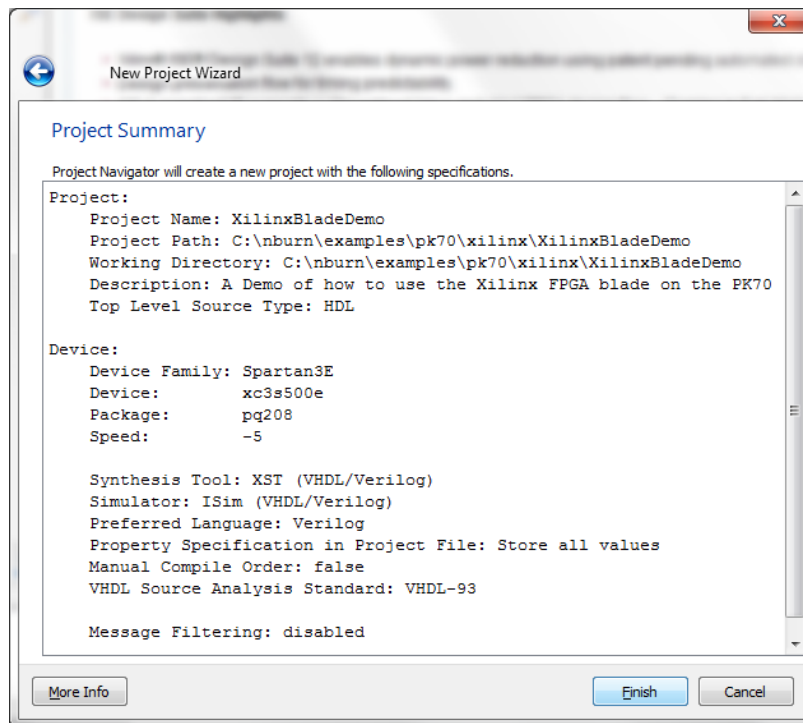


Figure 2.2-4

Once the project has been created, you must add the project files that we are building in this examples, XilinxBladeGpio.v and XilinxBlade.ucf. Begin by copying them into your project folder. Next, add them to the project by clicking on the add source project button. (Figure 2.2-5)

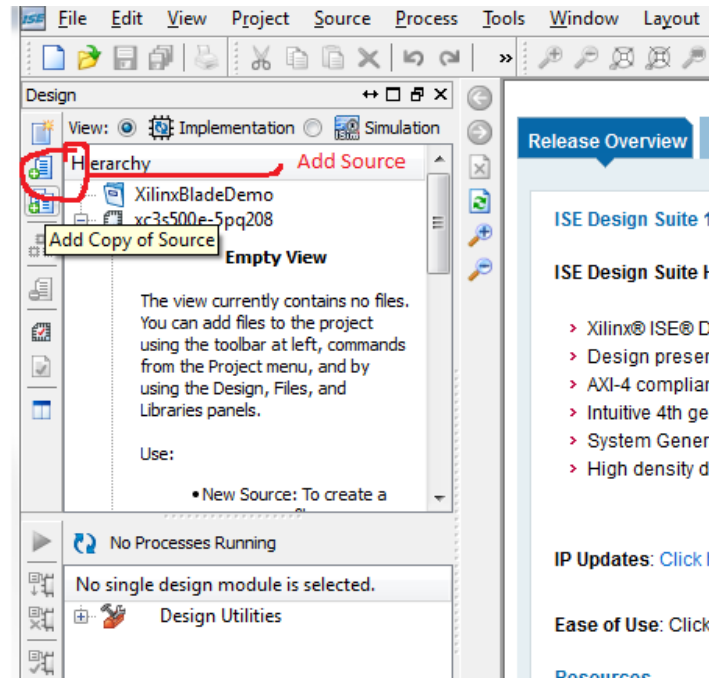


Figure 2.2-5

Clicking on the add source button brings up a file dialog. After adding the source, you should be back in the project summary screen. (Figure 2.2-6)

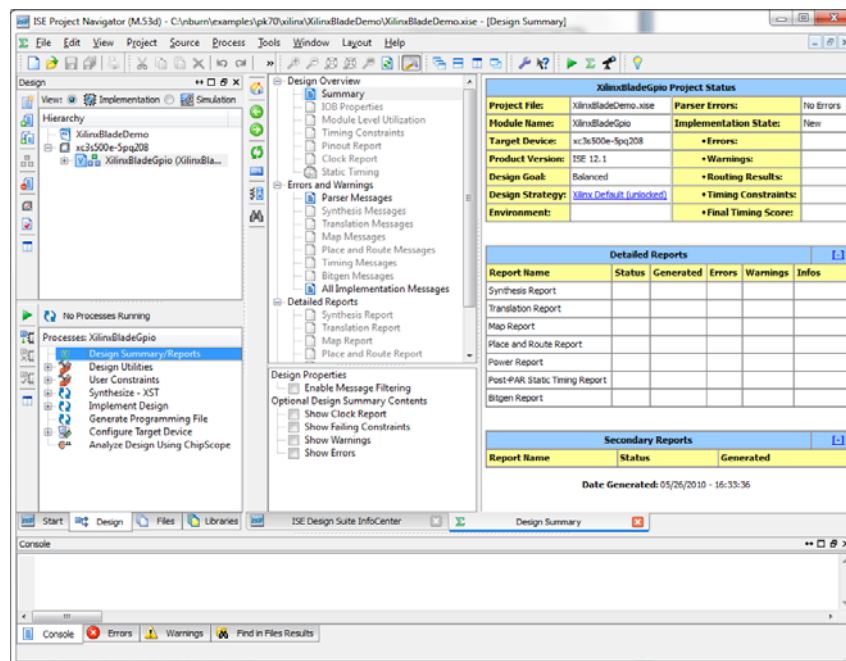


Figure 2.2-6

Click on the green arrow next to No Processes Running text to start the build. (Figure 2.2-7)

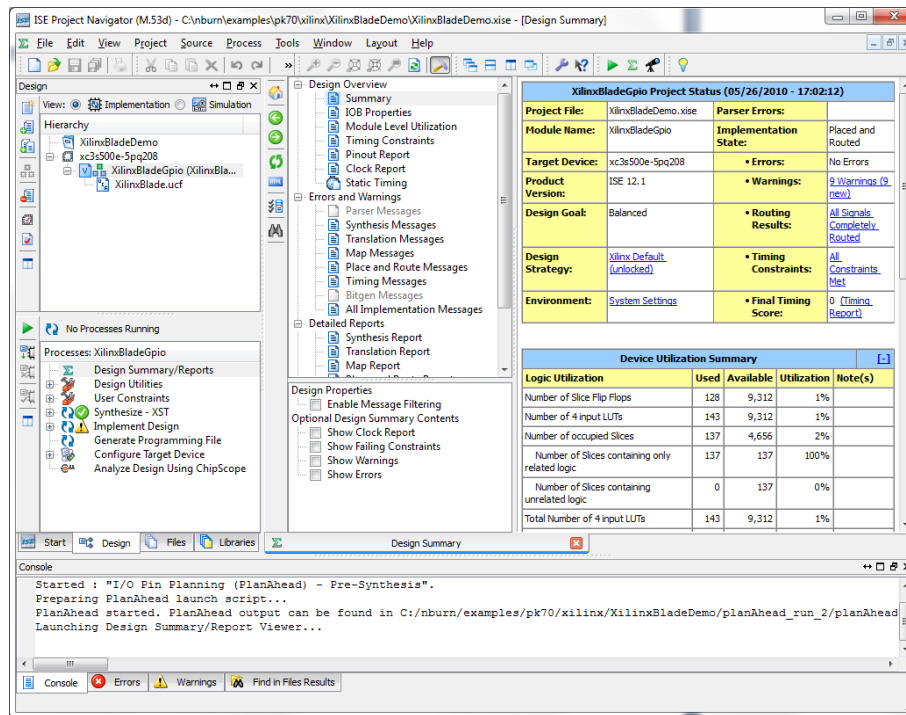


Figure 2.2-7

You may get some warning about drivers on the debug connector and a misplaced clock. It's safe to ignore these warnings.

2.3 Making a bitstream file

Create the bitstream file by opening the implement design tree item in the process window. Right click on Generate Programming File and choose run. (Figure 2.3-1)

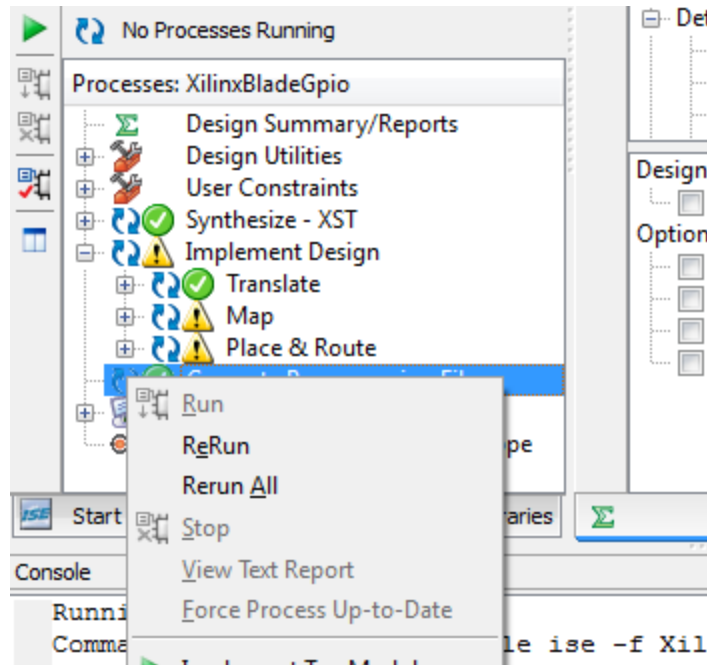


Figure 2.3-1

This creates the bit file, xilinxbladegpio.bit

2.4 Convert the bit file to a cpp to be used by your project

To convert the bit file into a cpp that can be compiled and linked in to your project, you must use two utilities, promgen and compfile. We will go over how to do this in both NBEclipse and building on the command line.

2.4.1 Using NBEclipse

NBEclipse will automatically convert and use the bit file. Simply drop the file in to your project. NBEclipse will recognize bit files and use the promgen and compfile utilities to build the cpp source file XilinxImage.cpp, which will be used by your project.

If you need to change the command line options used by promgen or compfile, right click on your project and select Properties. Under C/C++ Build, select Settings. You will find the options for promgen and compfile here.

2.4.2 Using the command line

To convert the bit file manually, you will need to call promgen to convert the file from a bit to a bin, and then compfile to convert from bin to a cpp.

Convert the bit file to a prom file:

```
promgen -w -p bin -c FF -o BinPromImage -u 0 xilinxbladegpio.bit -s 65536
```

Convert the bin file to a CPP file we can link to our project:

```
compfile BinPromImage.bin XilinxData XilinxSize XilinxImage.cpp
```

You can use promgen --help and compfile --help to examine the various command line options you can use to affect the build.

2.5 *Add the source files to your project*

Three source files must now be added to your NetBurner project to finish the build. First, add the cpp file that was generated in the last section, (2.4) XilinxImage.cpp. If you are following along in NBEclipse, this file will automatically be included, and can be found in your release directory.

Next, add the files LoadCode.cpp and LoadCode.h. These files were included in the FPGA Blade Board release files. These will allow your application to load the FPGA source while your application is running.

2.6 Call LoadCode on startup

Finally, you will need to tell the application to load the Xilinx FPGA with the stored code. This is done with the LoadCode function. You will also have to set up any required chipselects for talking to the device.

```
BOOL result = LoadCode();
```

The final results, if you are using the example files provided would be as follows:

```
Waiting 1sec to start 'A' to abort
Configured IP = 10.1.1.73
Configured Mask = 255.255.255.0
MAC Address= 00:03:f4:02:f8:2d
About to Load Code
Took 2 ticks
Load Code Result = 1
Locations readback      123ABC
In the example project this should be 123ABC
Wait....hit a key
```

So if what we readback matches what it should be in the line below, the FPGA loaded correctly.

Disclaimers and Copyrights

NetBurner, Inc. makes no representations or warranties with respect to the contents of our manuals and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, NetBurner, Inc. reserves the right to revise our manuals, make changes, and/or discontinue them without notice. Every effort has been made to ensure that all information is correct, but NetBurner, Inc. is not responsible for inadvertent errors.

NetBurner manuals contain links to third-party web sites that are not under the control of NetBurner, and NetBurner is not responsible for the content on any linked site. If you access any third-party sites listed in any of our manuals, then you do so at your own risk. NetBurner provides these links only as a convenience, and the inclusion of the link does not imply that NetBurner endorses or accepts any responsibility for the content on those third-party sites.

Under copyright laws, the documentation for the all NetBurner Manuals, PDFs, and Guides may not be copied, photocopied, reproduced, translated or reduced to any electronic medium or machine-readable form, in whole or in part, without prior written consent of NetBurner, Inc. NetBurner and the NetBurner logo are trademarks of NetBurner, Inc.

Life Support Disclaimer

NetBurner's products both hardware and software (including tools) are not authorized for use as critical components in life support devices or systems, without the express written approval of NetBurner, Inc. prior to use.

As used herein: (1) Life support devices or systems are devices or systems that (a) are intended for surgical implant into the body or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user. (2) A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.