Table of Contents

TABLE OF CONTENTS ................................................................................................................................. 5

CHAPTER 1: ABOUT THIS MANUAL .................................................................................................................. 8

1 MANUAL CONTENTS........................................................................................................................................ 8
   ABOUT THIS MANUAL ...................................................................................................................................... 9
   QUICK START GUIDE ...................................................................................................................................... 9
   BOARD HARDWARE ...................................................................................................................................... 9
   CONTROLLER SOFTWARE .......................................................................................................................... 9
   REFERENCE DESIGN ................................................................................................................................... 9
   ORDERING INFORMATION ........................................................................................................................ 9

2 ADDITIONAL RESOURCES ........................................................................................................................... 9

3 CONVENTIONS................................................................................................................................................ 10
   3.1 TYPOGRAPHICAL .................................................................................................................................... 10
   3.2 CONTENT ................................................................................................................................................ 12
      3.2.1 File names ......................................................................................................................................... 12
      3.2.2 Physical orientation and Origin ........................................................................................................ 12
      3.2.3 Part Pin Names .................................................................................................................................. 12
      3.2.4 Schematic Clippings ....................................................................................................................... 12
      3.2.5 Terminology ..................................................................................................................................... 12

CHAPTER 2: QUICK START GUIDE ................................................................................................................... 14

1 PROVIDED MATERIALS .................................................................................................................................. 14

2 ESD WARNING ............................................................................................................................................. 15

3 POWER-ON INSTRUCTIONS .......................................................................................................................... 15
   3.1 MEMORY AND HEATSINKS .................................................................................................................... 16
   3.2 PREPARE CONFIGURATION FILES ...................................................................................................... 16
   3.3 CONNECT CABLES .................................................................................................................................. 17
   3.4 VIEW CONFIGURATION FEEDBACK OVER RS232 ............................................................................... 17
      3.4.1 Watch the configuration status output .............................................................................................. 17
      3.4.2 Interactive configuration ................................................................................................................... 20
      3.4.3 Read temperature sensors ............................................................................................................... 20
      3.4.4 Shared Serial port ............................................................................................................................ 21
   3.5 CHECK LED STATUS LIGHTS ................................................................................................................ 21

4 USING THE REFERENCE DESIGN WITH THE PROVIDED SOFTWARE ............................................................ 25
   4.1 OPERATING THE USB CONTROLLER PROGRAM ................................................................................. 25
   4.2 COMMUNICATING TO THE USER DESIGN OVER THE SERIAL PORT .............................................. 27
   4.3 USING AETEST TO RUN HARDWARE TESTS ....................................................................................... 27
      4.3.1 AETest, Windows98 version .............................................................................................................. 28
      4.3.2 AETest, DOS version ........................................................................................................................ 28
      4.3.3 AETest on Linux or Solaris .............................................................................................................. 28
      4.3.4 Use AETest ...................................................................................................................................... 28
   4.4 MOVING ON ........................................................................................................................................... 29

CHAPTER 3: CONTROLLER SOFTWARE ........................................................................................................... 32

1 USB CONTROLLER ....................................................................................................................................... 32
   1.1 MENU OPTIONS ...................................................................................................................................... 32
      1.1.1 File Menu ....................................................................................................................................... 32
1.1.2 Edit Menu ........................................................................................................... 33
1.1.3 FPGA Configuration Menu ............................................................................... 33
1.1.4 Settings/Info Menu ............................................................................................ 34
1.2 PROGRAMMER’S GUIDE....................................................................................... 35
  1.2.1 Cypress CY7C68013A .................................................................................. 35
  1.2.2 Configuration FPGA ....................................................................................... 36
  1.2.3 Configuration Register Map ............................................................................ 36
  1.2.4 Vendor Request List ....................................................................................... 37
  1.2.5 USB Reference Design Control ....................................................................... 38
  1.2.6 Main Bus accesses ......................................................................................... 38
  1.2.7 Configuration .................................................................................................. 39
  1.2.8 Readback ......................................................................................................... 40
2 PCI ATEST APPLICATION ....................................................................................... 40
  2.1 FUNCTIONALITY ............................................................................................... 40
  2.2 RUNNING ATEST .............................................................................................. 41
  2.3 COMPILING ATEST ........................................................................................... 43
    2.3.1 Compiling ATest for DOS .............................................................................. 41
    2.3.2 Compiling ATest for Windows XP ................................................................. 42
3 UPDATING THE FIRMWARE .................................................................................. 42
  3.1 UPDATING THE MCU (FLASH) FIRMWARE ...................................................... 42
  3.2 UPDATING THE SPARTAN (EEPROM) FIRMWARE ........................................ 44

CHAPTER 4: HARDWARE .......................................................................................... 47
1 OVERVIEW ............................................................................................................. 47

2 CONFIGURATION CIRCUIT .................................................................................... 49
  2.1 OVERVIEW ......................................................................................................... 49
  2.2 THE SPARTAN 2 FPGA ...................................................................................... 49
    2.2.1 Spartan Configuration ................................................................................... 50
    2.2.2 CompactFlash ............................................................................................. 51
    2.2.3 MCU communication .................................................................................. 52
    2.2.4 PCI communication ..................................................................................... 52
    2.2.5 RS232 .......................................................................................................... 52
    2.2.6 IC .................................................................................................................... 53
  2.3 CONFIGURATION OPTIONS .............................................................................. 53
    2.3.1 Jog ............................................................................................................... 53
    2.3.2 CompactFlash ............................................................................................. 55
    2.3.3 USB ............................................................................................................. 57
    2.3.4 PCI ............................................................................................................... 58
  2.4 FPGA CONFIGURATION PROCESS .................................................................. 60
    2.5 MCU ................................................................................................................ 61
      2.5.1 RS232 ....................................................................................................... 61
      2.5.2 Clocks ....................................................................................................... 62
      2.5.3 LEDs ......................................................................................................... 63
      2.5.4 Memory space ........................................................................................... 63
      2.5.5 USB .......................................................................................................... 66
      2.5.6 CompactFlash ........................................................................................... 67
3 CLOCKING ............................................................................................................ 67
  3.1 GLOBAL CLOCK SYNTHESIS ......................................................................... 69
  3.2 USER CLOCK .................................................................................................... 70
4 RESET TOPOLOGY .................................................................................................... 70
5 POWER .................................................................................................................... 72
  5.1 SWITCHING POWER SUPPLIES ..................................................................... 74
  5.2 SECONDARY POWER SUPPLIES ..................................................................... 74
    5.2.1 DDR2 Termination Power ............................................................................ 74
  5.3 HEAT DISSIPATION ............................................................................................ 75
6 FPGA INTERCONNECT ........................................................................................... 76
Chapter 1: About This Manual

Welcome to DN8000K10PSX Logic Emulation Board

Congratulations on your purchase of the DN8000K10PSX LOGIC Emulation Board. If you are unfamiliar with Dini Group products, you should read Chapter 2, Quick Start Guide to familiarize yourself with the user interfaces the DN8000K10PSX provides.

Figure 1 DN8000K10PSX
1 Manual Contents

This manual contains the following chapters:

About This Manual
List of available documentation and resources available. Reader’s Guide to this manual

Quick Start Guide
Step-by-step instructions for powering on the DN8000K10PSX, loading and communicating with a simple provided FPGA design and using the board controls.

Board Hardware
Detailed description and operating instructions of each individual circuit on the DN8000K10PSX

Controller Software
A summary of the functionality of the provided software. Implementation details for the remote USB board control functions and instructions for developing your own USB host software.

Reference Design
Detailed description of the provided DN8000K10PSX reference design. Implementation details of the reference design interaction with DN8000K10PSX hardware features.

Ordering Information
Contains a list of the available options and available optional equipment. Some suggested parts and equipment available from third party vendors.

2 Additional Resources

For additional information, go to http://www.dinigroup.com. The following table lists some of the resources you can access from this website. You can also directly access these resources using the provided URLs.
### Support Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description/URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserDN8000K10PSX User Guide</td>
<td>This is the main source of technical information. The manual should contain most of the answers to your questions</td>
</tr>
<tr>
<td>Dini Group Web Site</td>
<td>The web page will contain the latest manual, application notes, FAQ, articles, and any device errata and manual addenda. Please visit and bookmark: <a href="http://www.dinigroup.com">http://www.dinigroup.com</a></td>
</tr>
<tr>
<td>Virtex 4 User Guide</td>
<td>Xilinx publication UG070</td>
</tr>
<tr>
<td></td>
<td>Most of your questions regarding usage and capabilities of the Virtex 4 devices will be answered here, including readback, boundary scan, configuration, and debugging</td>
</tr>
<tr>
<td>E-Mail</td>
<td>You may direct questions and feedback to the Dini Group using this e-mail address: <a href="mailto:support@dinigroup.com">support@dinigroup.com</a></td>
</tr>
<tr>
<td>Phone Support</td>
<td>Call us at <strong>858.454.3419</strong> during the hours of 8:00am to 5:00pm Pacific Time.</td>
</tr>
<tr>
<td>FAQ</td>
<td>The download section of the web page contains a document called <strong>DN8000K10PSX Frequently Asked Questions (FAQ)</strong>. This document is periodically updated with information that may not be in the User’s Manual.</td>
</tr>
</tbody>
</table>

Figure 2 Support Resources

### 3 Conventions

This document uses the following conventions. An example illustrates each convention.

#### 3.1 Typographical

The following typographical conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning or Use</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Courier font</td>
<td>Messages, prompts, and program files that the system displays</td>
<td>speed grade: - 100</td>
</tr>
<tr>
<td>Courier bold</td>
<td>Literal commands that you enter in a syntactical statement</td>
<td>ngdbuild design_name</td>
</tr>
<tr>
<td>Convention</td>
<td>Meaning or Use</td>
<td>Example</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td><strong>Garamond bold</strong></td>
<td>Commands that you select from a menu</td>
<td><em>File ➔ Open</em></td>
</tr>
<tr>
<td></td>
<td>Keyboard shortcuts</td>
<td><em>Ctrl+C</em></td>
</tr>
<tr>
<td><strong>Italic font</strong></td>
<td>Variables in a syntax statement for which you must supply values</td>
<td><em>rgdbuild design_name</em></td>
</tr>
<tr>
<td></td>
<td>References to other manuals</td>
<td>See the <em>Development System Reference Guide</em> for more information.</td>
</tr>
<tr>
<td></td>
<td>Emphasis in text</td>
<td>If a wire is drawn so that it overlaps the pin of a symbol, the two nets are <em>not</em> connected.</td>
</tr>
<tr>
<td><strong>Braces [ ]</strong></td>
<td>An optional entry or parameter. However, in bus specifications, such as bus[7:0], they are required.</td>
<td><em>rgdbuild [option_name] design_name</em></td>
</tr>
<tr>
<td><strong>Braces { }</strong></td>
<td>A list of items from which you must choose one or more</td>
<td>*lowpwr = {on</td>
</tr>
<tr>
<td>**Vertical bar</td>
<td>**</td>
<td>Separates items in a list of choices</td>
</tr>
<tr>
<td><strong>Vertical ellipse</strong></td>
<td>Repetitive material that has been omitted</td>
<td><em>IOB #1: Name = QOUT”</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>IOB #2: Name = CLKin’</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>-</em></td>
</tr>
<tr>
<td><strong>Horizontal ellipse</strong></td>
<td>Repetitive material that has been omitted</td>
<td><em>allow block block_name</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>loc1 loc2 ... locn</em></td>
</tr>
<tr>
<td><strong>Prefix “0x” or suffix “h”</strong></td>
<td>Indicates hexadecimal notation</td>
<td><em>Read from address 0x00110373, returned 4552494h</em></td>
</tr>
<tr>
<td><strong>Letter “#” or “_n”</strong></td>
<td>Signal is active low</td>
<td><em>INT# is active low</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>fpga_inta_n is active low</em></td>
</tr>
</tbody>
</table>
3.2 Content

3.2.1 File names
Paths to documents included on the User CD are prefixed with “D:\”. This refers to your CD drive’s root directory.

3.2.2 Physical orientation and Origin
By convention, the board is oriented as show on page 3, with the “top” of the board being the edge near Headers A and B, and the edge with the optical module connectors. The “right” edge is near the SMA connectors, the “left” side is the side with the PCI bezel. “topside” refers to the side of the PWB with FPGAs soldered to it, “backside” is the side with the daughtercard connectors. The reference origin of the board is the center of the lower PCI bezel mounting hole.

3.2.3 Part Pin Names
Pin names are given in the form \(<X><Y><Z>\); The \(<X>\) is one of: U for ICs, R for resistors, C for capacitors, P or J for connectors, FB or L for inductors, TP for test points, MH for mounting structures, FD for fiducials, BT for sockets, DS for diodes, F for fuses, HS for mechanics, PSU for power supply modules, Q for discreet semiconductors, RN for resistor networks, X for oscillators, Y for crystals. \(<Y>\) is a number uniquely identifying each part from other parts of the same X class on the same PWB. \(<Z>\) is the pin or terminal number or name, as defined in the datasheet of the part.Datasheets for all standard and optional parts used on the DN8000K10PSX are included in the Document library on the provided User CD.

3.2.4 Schematic Clippings
Partial schematic drawings are included in this document to aid quick understanding of the features of the DN8000K10PSX. These clippings have been modified for clarity and brevity, and may be missing signals, parts, net names and connections. Unmodified Schematics are included in the User CD as a PDF. Please refer to this document. Use the PDF search feature to search for nets and parts.

3.2.5 Terminology
Abbreviations and pronouns are used for some commonly used phrases.

MGT and RocketIO are used interchangeably. MGT is multi-gigabit transceiver. RocketIO is the Xilinx trademark on their multi gigabit transceiver hardware.

MCU is the Cypress FX2 Microcontroller, U39
Chapter 2: Quick Start Guide

The Dini Group DN8000K10PSX is the user-friendliest board available with multiple Virtex 4 FPGAs. However, due to the number of features and flexibility of the board, it will take some time to become familiar with all the control and monitoring interfaces equipped on the DN8000K10PSX. Please follow this quick start guide to become familiar with the board before starting your ASIC emulation project.

1 Provided Materials

Examine the contents of your DN8000K10PSX kit. It should contain:

- DN8000K10PSX board
- One CompactFlash card with reference design
- USB CompactFlash card reader
- RS232 IDC header cable to female DB9
- USB cable
- CD ROM containing:
  - Virtex 4 Reference Design
  - User manual PDF
  - Board Schematic PDF
  - USB program (usbcontroller.exe)
  - PCI program (Aetestdj.exe)
  - Source code for USB program, PCI program and DN8000K10PSX firmware
2 ESD Warning

The DN8000K10PSX is sensitive to static electricity, so treat the PCB accordingly. The target markets for this product are engineers that are familiar with FPGAs and circuit boards. However, if needed, the following web page has an excellent tutorial on the “Fundamentals of ESD” for those of you who are new to ESD sensitive products:

http://www.esda.org/basics/part1.cfm

There are two large grounded metal rails on the DN8000K10PSX.

The DN8000K10PSX has been factory tested and pre-programmed to ensure correct operation. You do not need to alter any jumpers or program anything to see the board work. A reference design is included on the provided CD and CompactFlash card.

The 200-pin connectors are not 5V tolerant. According to the Virtex 4 datasheets, the maximum applied voltage to these signals is VCCO + 0.5V (3.0V while powered on). These connections are not buffered, and the Virtex 4 part is sensitive to ESD. Take care when handling the board to avoid touching the daughtercard connectors.

3 Power-On Instructions

The image below represents your DN8000K10PSX. You will need to know the location of the following parts referenced in this chapter.

Figure 3 DN8000K10PSX configuration controls

To begin working with the DN8000K10PSX, follow the steps below.
### 3.1 Memory and heatsinks

There should be an active heatsink installed on each FPGA on the DN8000K10PSX and a fan over the power supply units. Virtex 4 FPGAs are capable of dissipating 15W or more, so you should always run them with heatsinks installed.

The DN8000K10PSX comes packaged without memory installed. If you want the Dini Group reference design to test your memory modules, you can install them now in the 1.8V DDR2 DIMM sockets.

![Figure 4 FPGA Names](image)

The socket DIMMB is connected to FPGA B. The socket can accept any capacity DDR2 Sodimm module. Note that DDR1 modules will not work in these slots since they are a completely different pin-out.

### 3.2 Prepare configuration files

The DN8000K10PSX reads FPGA configuration data from a CompactFlash card. To program the FPGAs on the DN8000K10PSX, FPGA design files (with a .bit file extension) put on the root directory of the CompactFlash card file using the provided usb card reader.

The DN8000K10PSX ships with a 128MB CompactFlash card preloaded with the Dini Group reference design.

1. Insert the provided CompactFlash card labeled “Reference Design” into your usb card reader. Make sure the card contains the files:

   - FPGA_A.bit
   - FPGA_B.bit
   - FPGA_C.bit
   - main.txt

   The files FPGA_A-C.bit are files created by the Xilinx program bitgen, part of the ISE 8.1 tools. The file main.txt contains instructions for the DN8000K10PSX configuration
circuitry, including which FPGAs to configure, and to which frequency the global clock
tenetworks should be automatically adjusted.

2. Insert the CompactFlash card labeled “Reference Design” into the DN8000K10PSX’s
CompactFlash slot.

### 3.3 Connect cables

The configuration circuitry can accept user input to control FPGA configuration or provide
feedback during the configuration process. The configuration circuitry IO can also be used to
transfer data to and from the user design.

1. Use the provided ribbon cable to connect the MCU RS232 port, P7 under FPGA C
(not the FPGA serial port) to a computer serial port to view feedback from the
configuration circuitry during FPGA configuration. Pin one of the header is marked
with a white dot, and the number “1”. The red wire in the cable should be on the pin-1
side of the header. Run a serial terminal program on your PC (On Windows you can use
HyperTerminal
Start->Programs->Accessories->Communications->HyperTerminal) and make sure the
computer’s serial port is configured with the following options:

- Bits per second: 19200
- Data bits: 8
- Parity: None
- Stop Bits: 1
- Flow control: None
- Terminal Emulation: VT100

2. Use the provided USB cable to connect the DN8000K10PSX to a Windows computer
(Windows XP is recommended).

3. Plug an ATX power supply into P5, or plug the DN8000K10PSX into a PCI slot. **Do
not use a separate external power supply in P5 if you are using the
DN8000K10PSX in a PCI slot.** Turn on the ATX power supply.

When the DN8000K10PSX powers on, it automatically loads Xilinx FPGA design files (ending
with a .bit extension), found on the CompactFlash card in the CompactFlash slot into the
FPGAs.

### 3.4 View configuration feedback over RS232

As the DN8000K10PSX powers on, your RS232 terminal (connected to P7) will display useful
information about the Configuration process.

#### 3.4.1 Watch the configuration status output
No USB cable detected, rebooting from FLASH...please wait

Setting ACLK...
N: 01 M: 000001000
DONE
Setting BCLK...
N: 01 M: 000001000
DONE
Setting DCLK...
N: 01 M: 000001000
DONE

= == DN8000K10PSX MCU FLASH BOOT == ==

-- FPGAS STUFFED --
A B
d

-- COMPACTFLASH INFO --
MAKER ID: EC
DEVICE ID: 75
SIZE: 32 MB

-- FILES FOUND ON COMPACTFLASH CARD
FPGA_B.BIT
FPGA_A.BIT
MAIN~1.TXT
MAIN.TXT

-- CONFIGURATION FILES --
FPGA A: FPGA_A.BIT
FPGA B: FPGA_B.BIT

-- OPTIONS --
Message level set to default: 2
Sanity check is set to default: ON
N: 00 M: 000001010
DONE
Setting BCLK...
N: 01 M: 000001100
DONE
Setting DCLK...
N: 01 M: 000001000
DONE
Setting R1CLK...
N: 01 M: 000001000
DONE
Setting R2CLK...
N: 01 M: 000001000
DONE

******************************************************************************

The global clocks GCLK0, GCLK1, GCLK2 (The RS232 port prints “ACLK”, “BCLK”, “DCLK”) are frequency-configurable. The M binary sequence represents the multiplication applied to the installed crystal. The N represents the division applied. U6, U14, U20, U31 and the ICS8442AY datasheet.

The MCU detects which FPGAs are present

The MCU detects if a CompactFlash card is present

The MCU tries to access the CompactFlash card. If the MCU is not successful in reading the files on the CompactFlash card, be sure you have not formatted the card in Windows. Windows uses a non-standard format for media cards and will make the card unreadable. You can download a format utility from dinigroup.com to repair your incorrectly-formatted SM card.

The MCU reads the contents of the file MAIN.TXT and executes each instruction line.

Here the MCU is setting the clocks according to instructions in MAIN.TXT

The MCU is configuring FPGA A according to instructions in MAIN.TXT

The sanity check option reads the design (“.bit”) file headers and verifies that the design is compiled for the same type of FPGA that the MCU detects on your DN8000K10PSX. If the design and FPGA do not match, the MCU will reject the file and flash the Error LED. You may need to disable to sanity check option (See Chapter X, section X) if you want to encrypt or compress your configuration
Performing Sanity Check on Bit File

BIT FILE ATTRIBUTES

FILE NAME: FPGA_B.BIT
FILE SIZE: 003A943B bytes
PART: 4vlx100f151317:05:01
DATA: 2005/07/19
TIME: 17:05:01
Sanity check passed

........DONE WITH CONFIGURATION OF FPGA: B

TEMPERATURE SENSORS

A YES
B YES
FPGA Temperature Alarm Threshold: 80 degrees C

DN8000K10PSX MAIN MENU (Jul 27 2005 10:38:05)
1.) Configure FPGAs using "MAIN.TXT"
2.) Interactive configuration menu
3.) Check configuration status
4.) Change MAIN configuration file
5.) List files on CompactFlash
6.) Display CompactFlash text file
7.) Change RS232 PPC Port
8.) Set FPGA Address
9.) Write to FPGA at current address
a.) Read from FPGA at current address
g.) Display FPGA Temperatures
h.) Set FPGA Temperature Alarm Threshold

ENTER SELECTION:

The MCU is configuring FPGA B according to instructions in MAIN.TXT

The MCU is setting the temperature threshold to cause a board reset.

You should see the DN8000K10PSX MCU main menu. If the reference design is loaded in the Virtex-4 FPGAs, then you should see the above on your terminal. Try pressing 3 to see if the configuration circuit was successful in programming the FPGAs.

The easiest way to verify your FPGAs are configured is to look at DS19, DS21, DS23 located next to each FPGA. When the green LEDs are lit, the FPGA under it is successfully configured.
3.4.2 Interactive configuration
If you want to put multiple designs on a single CompactFlash card, you can use the interactive configuration menu to select which .bit file to use on each FPGAs. Select menu option 2.

```
ENTER SELECTION: 2

--- INTERACTIVE CONFIGURATION MENU ---

1) Select bit files to configure FPGA(s)
2) Set verbose level (current level = /)
3) Enable sanity check for bit files
M) Main Menu

Enter Selection:
```

Figure 6 Interactive Config Menu

3.4.3 Read temperature sensors
The DN8000K10PSX is equipped with temperature sensors to measure and monitor the temperature on the die of the Virtex 4 FPGAs. According to the Virtex 4 datasheet, the maximum recommended operating temperature of the die is 85°C degrees. If the microcontroller measures a temperature above 80 degrees, it will reset the DN8000K10PSX.

If you think your DN8000K10PSX is resetting due to temperature overload, you can use the temperature monitor menu to measure the current junction temperature of each FPGA.

```
ENTER SELECTION: g

-- FPGA TEMPERATURES (Degrees Celsius [+/- 4]) --
B 29
-- Set FPGA Temperature Alarm Threshold --
(degrees C, decimal values, range [1-127])
Old Threshold: 80
New Threshold: 85
Threshold Updated: 85 Degrees C
```

Figure 7 Temperature Threshold Menu

The Virtex 4 FPGA can operate as hot as 120°C degrees before damaging the part, although timing specifications are not guaranteed. The MCU allows you to change the reset threshold, although we recommend improving your heat dissipation to maintain a low junction temperature. The Xilinx ISE tools allow changing the temperature of the die to calculate accurate timing information.
3.4.4 Shared Serial port

The DN8000K10PSX has one serial port (P6) for user use. This single port is shared so that any FPGA can access it through its RX and TX signals (MB[36] and MB[37] respectively). You do not need to use the RS232 MCU interface to change the FPGA “RS232 MUX”, as this will have no effect for the DN8000K10PSX card. It is important to note, however, that because the RX232 TX line (MB[37]) is shared between all FPGA’s, it is up to the user to avoid contention. Only one FPGA can drive this signal at a time.

3.5 Check LED status lights

The DN8000K10PSX has many status LEDs to help the user confirm the status of the configuration process.

1. Check the power voltage indication LEDs to confirm that all voltage rails of the DN8000K10PSX are present. From the left, the LEDs indicate the presence of 1.2V, 1.8V, 2.5V, 3.3V and 5V. Green lit LED’s on the voltage present LEDs indicate the rails are greater than ~1.7V. If this LED is not lit green, the DN8000K10PSX might not function properly.

2. Check the Spartan FPGA status LED, DS43. This LED indicates that the Spartan II FPGA has been configured. If this LED is not lit soon after power on, then there may be a problem with the firmware on the DN8000K10PSX. This LED off or blinking may indicate a problem with one of the board’s power supplies.
3. Check the FPGA A status LED, DS20 to the upper right of FPGA A. This green LED is lit when FPGA A is configured and operational. This light should be on if you loaded the reference design from the CompactFlash card.

4. Check the FPGA B status LED, DS22 above and to the right of FPGA B. This light should be lit green if your DN8000K10PSX was installed with the FPGA B option, and the reference design is loaded.

5. Check the FPGA C status LED, DS24 to the upper right of FPGA C. This green LED will light if you have the FPGA C option and the FPGA is configured.

6. Check the FPGA A User LEDs to the right of FPGA A. If you have successfully loaded the Dini Group’s DN8000K10PSX reference design, these should be lit yellow.

7. Check the FPGA C User LEDs on the right and left side of FPGA C. If the reference design is loaded, these will light yellow.

8. If you suspect one or more FPGAs did not configure properly, the easiest way to determine the cause of the error is to connect a terminal to the RS232 port (P7) and try to configure again. Configuration feedback will be presented over this port. P7 is located along the bottom edge of the board, under FPGA C. Pin one of this header is marked with a white dot.

You should also notice the Fans mounted above the 3 Virtex 4 FPGAs and the Fan mounted above the power supplies spinning.

<table>
<thead>
<tr>
<th>Assembly Number</th>
<th>Signal</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS5</td>
<td>1.2V_PRESENT</td>
<td>The 5.0V power rail is present (above ~1.0V)</td>
</tr>
<tr>
<td>DS6</td>
<td>1.8V_PRESENT</td>
<td>The 1.8V power rail is present (above ~1.0)</td>
</tr>
<tr>
<td>DS7</td>
<td>2.5V_PRESENT</td>
<td>The 2.5V power rail is present (above ~1.7)</td>
</tr>
<tr>
<td>DS8</td>
<td>3.3V_PRESENT</td>
<td>The 3.3V power rail is present (above ~1.7)</td>
</tr>
<tr>
<td>DS9</td>
<td>5.0V_PRESENT</td>
<td>The 5.0V power rail is present (above ~1.7)</td>
</tr>
<tr>
<td>LED Code</td>
<td>Description</td>
<td>Status Information</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>DS25, DS26, DS27, DS28</td>
<td>MCU LEDs</td>
<td>MCU status codes. One or more blinking LEDs indicates an error.</td>
</tr>
<tr>
<td>DS43</td>
<td>Spartan DONE</td>
<td>Indicates the configuration FPGA is configured. This should always be on</td>
</tr>
<tr>
<td>DS53, DS54, DS55, DS56</td>
<td>Spartan LED</td>
<td>These are used for debugging. Ignore these.</td>
</tr>
<tr>
<td>DS19</td>
<td>FPGA A DONE</td>
<td>FPGA A is configured</td>
</tr>
<tr>
<td>DS23</td>
<td>FPGA C DONE</td>
<td>FPGA C is configured</td>
</tr>
<tr>
<td>DS21</td>
<td>FPGA B DONE</td>
<td>FPGA B is configured</td>
</tr>
<tr>
<td>DS10</td>
<td>ETHN_LINK10</td>
<td>The Ethernet PHY detects a 10Base-T link</td>
</tr>
<tr>
<td>DS11</td>
<td>ETHN_LINK100</td>
<td>The Ethernet PHY detects a 100Base-T link</td>
</tr>
<tr>
<td>DS12</td>
<td>ETHN_DUPLEX</td>
<td>The Ethernet PHY is configured in full-duplex mode</td>
</tr>
<tr>
<td>T5 (Green)</td>
<td>ETHN_LINK1000</td>
<td>The Ethernet PHY detects a 1000Base-T link</td>
</tr>
<tr>
<td>T5 (Yellow)</td>
<td>ETHN_ACT</td>
<td>The Ethernet PHY detects activity on the RJ45 port</td>
</tr>
<tr>
<td>DS58, DS59</td>
<td>PCL_ACT</td>
<td>These LEDs are disabled by default and may not be installed</td>
</tr>
<tr>
<td>DS13</td>
<td>USB_ACT</td>
<td>This LED blinks when there is USB activity</td>
</tr>
<tr>
<td>DS52</td>
<td>CF_ACT</td>
<td>This LED blinks when there is activity on the CompactFlash card.</td>
</tr>
<tr>
<td>DS57</td>
<td>PCI_ACT</td>
<td>This LED blinks when there is PCI activity (configuration or MainBus)</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------</td>
<td>--------------------------------------------------------------------</td>
</tr>
<tr>
<td>DS20, DS22, DS24, DS45</td>
<td>Config Error</td>
<td>These LEDs light on some subset of possible configuration errors. (Currently none)</td>
</tr>
<tr>
<td>DS14</td>
<td>SYS_RSTn</td>
<td>This LED lights when the board is in reset. This can occur if there is a power problem or if the user holds the “hard reset” button. Check this LED if the board fails to appear over USB or otherwise does nothing.</td>
</tr>
<tr>
<td>DS29, DS31, DS33, DS36</td>
<td></td>
<td>User LEDs from FPGA A</td>
</tr>
<tr>
<td>DS15, DS16, DS17, DS18</td>
<td></td>
<td>User LEDs from FPGA B</td>
</tr>
<tr>
<td>DS30, DS32, DS34, DS37</td>
<td></td>
<td>User LEDs from FPGA B</td>
</tr>
<tr>
<td>DS38, DS39, DS41, DS46</td>
<td></td>
<td>User LEDs from FPGA C</td>
</tr>
<tr>
<td>DS35, DS40, DS42, DS47, DS48, DS49, DS50, DS51</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 9 DN8000K10PSX LEDs
4 Using the Reference Design with the Provided Software

To communicate with the reference design (or user design) on the emulation board, the DN8000K10PSX provides three options out of the box.

- USB
- PCI
- RS232

The USB and PCI interfaces allow configuration of the FPGAs and bulk data transfer to and from the User design. The RS232 interface allows low-speed data transfers to and from the User design, and control and monitoring of the configuration process.

This section will get you started and show you how to operate the provided software. For detailed information about the reference design and implementation details, see Chapter X, The Reference Design.

4.1 Operating the USB controller program

Use the provided USB monitoring software to verify that the design is loaded into the FPGAs.

1. Insert the CDROM that came with your DN8000K10PSX into the CDROM drive of your computer.

2. Connect the USB cable to your DN8000K10PSX and a Windows XP PC. (Before or after the DN8000K10PSX has powered on)

3. When you connect the USB cable to your PC for the first time, Windows detects the DN8000K10PSX and asks for a driver. The board should identify itself as a “DiNi Prod FLASH BOOT”. When the new device detected window appears, select the option "install from a list" -> select "search for the best driver in these locations". Select "include the location in the search" and browse to the product CD in D:\USB_Software_Applications\driver\windows_wdm\ ->select "finish"

4. After Windows installs the driver, you will be able to see the following device in the USB section of Windows device manager: “DiniGroup DN8000K10PSX FLASH boot”.

5. Run the USB controller application found on the product CD in D:\USB_Software_Applications\USBController\USBController.exe
6. This window will appear showing the current state of the DN8000K10PSX. Next to each FPGA a green light will appear if that FPGA is configured successfully. The above window shows the USB Controller connected to a DN8000K10PSX with a single FPGA in the B position. If you have the reference design loaded and a DDR2 SODIMM installed, you can use the USB Controller to run tests of the SODIMM. From the FPGA Memory menu, select Test DDR.

7. Clear the FPGAs of their configurations. Right-click on an FPGA and select from the popup menu, “Clear FPGA”. The green light above the FPGA on the GUI and on the board should stop shining green.

8. Configure an FPGA using the USB Controller program. Right-click on an FPGA and select Configure FPGA via USB from the popup menu. The program will open a dialog box for you to select the configuration file to use for configuration. Browse to the provided user’s CD D:\FPGA_Reference_Designs\Programming_Files\DN8000K10PSX\MainTest\LX160\fpga_a.bit

   If you are configuring an LX200 or FX60 devices you should select a bit file from the LX200 or FX60 directories instead. If you are configuring FPGA B or FPGA C, you should select fpga_b.bit or fpga_c.bit instead.
FPGA B cleared successfully.
FPGA A cleared successfully.
Doing a sanity check...Sanity Check passed. Configuring FPGA B via USB...please wait.
File D:\dn_BitFiles\DN8000K10PSX\MainTest\LX100\fpga_b.bit transferred.
Configured FPGA B via USB

Figure 11 USB Controller Log Output

9. The message box below the DN8000K10PSX graphic should display some information about the configuration process

The USB Controller program also allows you to easily configure and transfer data to and from the user design on the emulation board. More information is provided in Chapter X, “The USB program”

4.2 Communicating to the User Design over the Serial Port

You may want to communicate with your design over the user serial port (P6). Only one FPGA can use P6 at a time. The port is shared between all FPGA’s which connect to it on the dedicated MainBus lines MB[36] (RX) and MB[37] (TX). You do not need to change the “RS232 MUX” settings for the DN8000K10PSX- this function has no effect.

Connect an RS232 cable to P6, the FPGA RS232. It is located right next to the configuration RS232 port, P7. If you have the reference design loaded, the FPGA RS232 port runs at 19200 bps, 8 bit, no parity. On the computer’s terminal, the reference design is programmed to loopback the input to the output. If on the terminal you can read your own output, then the reference design was able to capture the RS232 signal and generate an RS232 signal that your computer could capture.

If you are familiar with previous Dini Group products, the reference design test outputs could be read from this serial port. On the DN8000K10PSX, you must use the AETEST application to read the results of self-test.

4.3 Using AETEST to run hardware tests

AETest is the program that you can use to verify the hardware on the DN8000K10PSX, as well as to demonstrate the reference design function. The following instructions assume you have a PC running the Windows XP operating system. The user CD includes a Windows version of the AETest program. If you plan to use the DN8000K10PSX in stand-alone mode, connect the DN8000K10PSX to your WindowsXP computer and use aetest_usb in D:\USB_Software_Applications\USB_CMD_Line_AETEST_USB\aesub_wdm.exe
If the computer asks for a driver, click “Have Disk” and browse to D:\USB_Software_Applications\driver\windows_wdm\dndevusb.inf
If you are going to use the DN8000K10PSX in a PCI slot, turn off the computer, insert the DN8000K10PSX into an unused PCI slot, and turn the computer on. If the operating system
asks for a driver, click “Have Disk” and browse to
D:\PCI_Software_Applications\Aetest\wdmdrv\drv\dndev.inf
Then run the PCI version of the AETest application at
D:\PCI_Software_Applications\Aetest\aetest\aetest_wdm.exe

4.3.1 AETest, Windows98 version
If you are using Windows 98 on your PC, you must use a different set of drivers to access the
DN8000K10PSX. Drivers have been provided for win98 are in
D:\PCI_Software_Applications\Aetest\win98drv

Due to the poor USB support in Windows 98, Dini Group discontinued the win98 USB driver.

4.3.2 AETest, DOS version
These DOS instructions are outdated. If you need DOS mode, contact support@dinigroup.com

The Aetest application will also run under DOS. The DOS version of AETest will not run
under Windows’s DOS emulation mode. You must boot into DOS using a boot disk.

1) Create a windows boot floppy disk. The easiest way to do this is to format a disk
using WindowsXP with the “Create an MS-DOS startup disk” option checked.

2) D:\AETest\aetest_floppy
   contains some files you need to add to the boot disk. Copy the contents of this
directory to the disk. Add the program
   D:\AETest\DJGPP\CWSDPMI.EXE
to the floppy. The DOS version of AETest requires CWSDPMI.EXE to access
the PCI bus.

3) Plug DN8000K10PSX into the PCI slot.

4) Boot from the floppy

5) Run AETESTDJ.EXE

4.3.3 AETest on Linux or Solaris
To use the AETest application on Linux or Solaris, you must compile the source code included
on the User CD. Instructions for compiling AETest are found in chapter 3.

4.3.4 Use AETest
The Aetest application should display it’s main menu.
Run one of the tests. Choose option 1. Remember, the FPGA you test has to be loaded with the reference design, or the test will fail.

For more information on the AETEST program, see Chapter 3.

4.4 Moving On

Congratulations! You have just programmed the DN8000K10PSX and learned all of the features that you must know to start your emulation project. If you are new to Xilinx FPGA, you might want move to chapter 4, introduction to ISE and Virtex 4 and start adding your Verilog code to the reference design. You can start your design using the UCF files provided
with the reference design, or generate IO placement from the customer netlist of the DN8000K10PSX provided on the user CD.
Chapter 3: Controller Software

1 USB Controller

USBController application is used to communicate with the DN8000K10PSX.

All USBController source code is included on the CD-ROM shipped with the DN8000K10PSX. The USBController can be installed on Windows 98/ME/2000/XP. There is a command line version called AETEST_USB that can be installed on Linux and Solaris.

The USBController Application contains the following functionality:
- Verify Configuration Status
- Configure FPGA(s) over USB
- Configure FPGAs via CompactFlash card
- Clear FPGA(s)
- Reset FPGA(s)
- Set Global clocks frequency
- Set RocketIO CLK Frequency
- Update MCU FLASH firmware

The following function interface with the Dini Group reference design.
- Read/Write to FPGA(s) – see the chapter Reference Design for a description of the Main bus interface.
- Test DDRs/FLASH/Reisters/FPGA Interconnect

1.1 Menu Options

1.1.1 File Menu

The File Menu has the following 2 options:

a. Open – opens a file with the selected text editor (notepad by default). To change the text editor see Settings/Info Menu section

b. Exit – Closes the USBController application
1.1.2 **Edit Menu**
The Edit Menu performs the basic edit commands on the command log in the bottom half of the USBController window.

1.1.3 **FPGA Configuration Menu**
The FPGA Configuration Menu has the following options:

(1) Configure via USB (individually) – After selecting this option a window will pop and ask which FPGA you want to configure and then what bitfile you want to configure the selected FPGA with. The status of the FPGA configuration will detailed in the log window and the DN8000K10PSX will be updated after the bitfile has been transferred.

(2) Configure via USB using file – This option allows the user to configure more than one FPGA over USB at a time. To use this option you must create a setup file that contains information on which FPGA(s) should be configured and what bitfiles should be used for each FPGA. The file should be in the following format, the first character of each line represents which FPGA you want configured (a-f or A-F), this letter should be followed by a colon and then the path to the bitfile to use for this FPGA. The path to the bitfile is relative to the directory where this setup file is, or you can use the full path. Below is an example of an accepted setup file:

```
A: fpga_a.bit
B: fpga_b.bit
C: fpga_c.bit
```

(3) Configure via CompactFlash Card – This option allows the user to use a CompactFlash card to configure the FPGAs. Please section [Creating Configuration File “main.txt”](#) for information on what files should be on the CompactFlash card to use this option.

(4) Clear All FPGAs – This option will deconfigure all FPGAs.

(5) Reset – This options sends an active low reset (active for approx. 20ns) to all FPGAs on the signal called RESET_FPGASn which is connected to the following I/O pins:

- FPGA A: AK19
- FPGA B: K21
- FPGA C: AG18
1.1.4 Settings/Info Menu

The Settings/Info Menu has the following options:

(1) Set FPGA RocketIO CLK Frequency – When the DN8000K10PSX is first powered up the RocketIO CLK inputs to the FPGAs are inactive. The RocketIO CLK Inputs are connected to the following FPGA Differential CLK inputs on all FPGAs: F21/G21 and AT21/AU21. This menu option allows the user to specify what frequency the RocketIO CLKs should be set at for each FPGA. The supported frequency range is 31.25MHz – 700MHz. After selecting this option, a pop-up window will ask which FPGA’s RocketIO Frequency you want to set (or you can choose to set all to the same frequency), and then what frequency you want. Check the log window to verify what frequency the CLKs were actually set at.

(2) Set Global clock frequencies

The clocks on the DN8000K10PSX are automatically adjusted to the user’s desired frequency by reading the setup file on the CompactFlash card. If you wish to change the frequency after power-on, or do not want to use a CompactFlash card, you can set the frequency in the USB program.

GCLK0) Generated from a 25MHz crystal. Available frequencies are:

<table>
<thead>
<tr>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>31.25</td>
</tr>
<tr>
<td>34.375</td>
</tr>
<tr>
<td>37.5</td>
</tr>
<tr>
<td>40.625</td>
</tr>
<tr>
<td>43.75</td>
</tr>
<tr>
<td>46.875</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>53.125</td>
</tr>
<tr>
<td>56.25</td>
</tr>
<tr>
<td>59.375</td>
</tr>
<tr>
<td>62.5</td>
</tr>
<tr>
<td>65.625</td>
</tr>
<tr>
<td>68.75</td>
</tr>
<tr>
<td>71.875</td>
</tr>
<tr>
<td>75</td>
</tr>
<tr>
<td>78.125</td>
</tr>
<tr>
<td>81.25</td>
</tr>
<tr>
<td>84.375</td>
</tr>
<tr>
<td>87.5</td>
</tr>
<tr>
<td>93.75</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>106.25</td>
</tr>
<tr>
<td>112.5</td>
</tr>
<tr>
<td>118.75</td>
</tr>
<tr>
<td>125</td>
</tr>
<tr>
<td>131.25</td>
</tr>
<tr>
<td>137.5</td>
</tr>
<tr>
<td>143.75</td>
</tr>
<tr>
<td>150</td>
</tr>
<tr>
<td>156.25</td>
</tr>
<tr>
<td>162.5</td>
</tr>
<tr>
<td>168.75</td>
</tr>
<tr>
<td>175</td>
</tr>
<tr>
<td>187.5</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>212.5</td>
</tr>
<tr>
<td>225</td>
</tr>
<tr>
<td>237.5</td>
</tr>
<tr>
<td>250</td>
</tr>
<tr>
<td>262.5</td>
</tr>
<tr>
<td>275</td>
</tr>
<tr>
<td>287.5</td>
</tr>
<tr>
<td>300</td>
</tr>
<tr>
<td>312.5</td>
</tr>
<tr>
<td>325</td>
</tr>
<tr>
<td>337.5</td>
</tr>
<tr>
<td>350</td>
</tr>
<tr>
<td>375</td>
</tr>
<tr>
<td>400</td>
</tr>
<tr>
<td>425</td>
</tr>
<tr>
<td>450</td>
</tr>
<tr>
<td>475</td>
</tr>
<tr>
<td>500</td>
</tr>
<tr>
<td>525</td>
</tr>
<tr>
<td>550</td>
</tr>
<tr>
<td>575</td>
</tr>
<tr>
<td>600</td>
</tr>
<tr>
<td>625</td>
</tr>
<tr>
<td>650</td>
</tr>
<tr>
<td>675</td>
</tr>
<tr>
<td>700</td>
</tr>
</tbody>
</table>

GCLK1) Generated from a 14.318 Mhz crystal. Supported frequencies are:

<table>
<thead>
<tr>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>32.22</td>
</tr>
<tr>
<td>34.01</td>
</tr>
<tr>
<td>35.80</td>
</tr>
<tr>
<td>37.58</td>
</tr>
<tr>
<td>39.37</td>
</tr>
<tr>
<td>41.16</td>
</tr>
<tr>
<td>42.95</td>
</tr>
<tr>
<td>44.74</td>
</tr>
<tr>
<td>46.53</td>
</tr>
<tr>
<td>48.32</td>
</tr>
<tr>
<td>50.11</td>
</tr>
<tr>
<td>51.90</td>
</tr>
<tr>
<td>53.69</td>
</tr>
<tr>
<td>55.48</td>
</tr>
<tr>
<td>57.27</td>
</tr>
<tr>
<td>59.06</td>
</tr>
<tr>
<td>60.85</td>
</tr>
<tr>
<td>62.64</td>
</tr>
<tr>
<td>64.43</td>
</tr>
<tr>
<td>66.22</td>
</tr>
<tr>
<td>68.01</td>
</tr>
<tr>
<td>69.80</td>
</tr>
<tr>
<td>71.59</td>
</tr>
<tr>
<td>73.38</td>
</tr>
<tr>
<td>75.17</td>
</tr>
<tr>
<td>76.96</td>
</tr>
<tr>
<td>78.75</td>
</tr>
<tr>
<td>80.54</td>
</tr>
<tr>
<td>82.33</td>
</tr>
<tr>
<td>84.12</td>
</tr>
<tr>
<td>85.91</td>
</tr>
<tr>
<td>89.49</td>
</tr>
<tr>
<td>93.07</td>
</tr>
<tr>
<td>96.65</td>
</tr>
<tr>
<td>100.2</td>
</tr>
<tr>
<td>103.8</td>
</tr>
<tr>
<td>107.4</td>
</tr>
<tr>
<td>111.0</td>
</tr>
<tr>
<td>114.5</td>
</tr>
<tr>
<td>118.1</td>
</tr>
<tr>
<td>121.7</td>
</tr>
<tr>
<td>125.3</td>
</tr>
<tr>
<td>128.9</td>
</tr>
<tr>
<td>132.4</td>
</tr>
<tr>
<td>136.0</td>
</tr>
<tr>
<td>139.6</td>
</tr>
<tr>
<td>143.2</td>
</tr>
<tr>
<td>146.8</td>
</tr>
<tr>
<td>150.3</td>
</tr>
<tr>
<td>153.9</td>
</tr>
<tr>
<td>157.5</td>
</tr>
<tr>
<td>161.1</td>
</tr>
<tr>
<td>164.7</td>
</tr>
<tr>
<td>168.2</td>
</tr>
<tr>
<td>171.8</td>
</tr>
<tr>
<td>179.0</td>
</tr>
<tr>
<td>186.1</td>
</tr>
<tr>
<td>193.3</td>
</tr>
<tr>
<td>200.5</td>
</tr>
<tr>
<td>207.6</td>
</tr>
<tr>
<td>214.8</td>
</tr>
<tr>
<td>221.9</td>
</tr>
<tr>
<td>229.1</td>
</tr>
<tr>
<td>236.2</td>
</tr>
<tr>
<td>243.4</td>
</tr>
<tr>
<td>250.6</td>
</tr>
<tr>
<td>257.7</td>
</tr>
<tr>
<td>264.9</td>
</tr>
<tr>
<td>272.0</td>
</tr>
<tr>
<td>279.2</td>
</tr>
<tr>
<td>286.4</td>
</tr>
<tr>
<td>293.5</td>
</tr>
<tr>
<td>300.7</td>
</tr>
<tr>
<td>307.8</td>
</tr>
<tr>
<td>315.0</td>
</tr>
<tr>
<td>322.2</td>
</tr>
<tr>
<td>329.3</td>
</tr>
<tr>
<td>336.5</td>
</tr>
<tr>
<td>343.6</td>
</tr>
<tr>
<td>358.0</td>
</tr>
<tr>
<td>372.3</td>
</tr>
<tr>
<td>386.6</td>
</tr>
<tr>
<td>400.9</td>
</tr>
<tr>
<td>415.2</td>
</tr>
<tr>
<td>429.5</td>
</tr>
<tr>
<td>443.9</td>
</tr>
<tr>
<td>458.2</td>
</tr>
<tr>
<td>472.5</td>
</tr>
<tr>
<td>486.8</td>
</tr>
<tr>
<td>501.1</td>
</tr>
<tr>
<td>515.4</td>
</tr>
<tr>
<td>529.8</td>
</tr>
<tr>
<td>544.1</td>
</tr>
<tr>
<td>558.4</td>
</tr>
<tr>
<td>572.7</td>
</tr>
<tr>
<td>587.0</td>
</tr>
<tr>
<td>601.4</td>
</tr>
<tr>
<td>615.7</td>
</tr>
<tr>
<td>630.0</td>
</tr>
<tr>
<td>644.3</td>
</tr>
<tr>
<td>658.6</td>
</tr>
<tr>
<td>672.9</td>
</tr>
<tr>
<td>687.3</td>
</tr>
</tbody>
</table>

GCLK2) Generated from a 16.0 Fundamental crystal. Supported frequencies:
(3) Change Text Editor – This option allows the user to select a text editor to use (the default editor is notepad).

(4) FPGA Stuffing Information – This option will display the type of FPGAs that are stuffed on the DN8000K10PSX.

(5) MCU Firmware Version – This option will display the MCU Firmware version in the log window.

(6) BOARD/SPARTAN Version – This option will display the Board Version along with the Spartan (Config Fpga) Version.

1.2 Programmer's Guide
This section contains information to help the development of your own USB software for use with the DN8000K10PSX. If you do not need to develop your own USB control software, or modify the Dini Group USB controller, you can skip this section. All of the code for AETest and USB Controller (windows only) is provided on the user CD. Precompiled Windows drivers are also provided for Windows XP. You should also read the section Hardware: Configuration Section before attempting to modify the USB software.

The source code for the Windows USB Controller (GUI) is provided in D:\USB_Software_Applications\USBController along with the Microsoft Visual Studio 6 project used to compile. Visual Studio 6 or later is required to compile this program.

1.2.1 Cypress CY7C68013A
A Cypress Microcontroller (MCU) with built-in USB support provides the USB interface of the DN8000K10. All communication with the DN8000K10PSX over USB is initiated by the host (PC) and consists either of a USB vendor request (See USB specification and Cypress datasheet) or a USB bulk transfer.

Vendor requests can contain short (512Byte) messages in either direction, and cause the MCU to execute code. In response to most vendor requests, the MCU will modify or read values in the Configuration memory space (see next section).
Since vendor requests can contain only a limited amount of data, USB Bulk transfers are used to send configuration data to the DN8000K10. The MCU is too slow to process USB 2.0 data at full speed, and so the bulk transfer data is sent to external pins on the Cypress MCU (see Cypress datasheet) and to the configuration FPGA (next section). Currently, this data is only used to configure FPGAs, and so the data is sent to the SelectMap pins of the Virtex 4 FPGAs.

To begin communication with the DN8000K10, the USB Controller program creates a USB connection object in the host operating system, by opening Vendor ID 0x1234 product ID 0x1234. (For the purposes of updating the firmware, the DN8000K10 can come up in “EPROM” mode, where it loads a program capable of connecting over USB to a host, downloading firmware and writing it to the MCU flash memory, U201. The check the MCU makes on reset to determine which mode it should start in is the firmware update switch, S1 #4. This EPROM code is stored in the EPROM DIP installed in U203. When the MCU is in this mode, it registers itself to the operating system as Vendor ID 0x1234, product ID 0x1233. For firmware update instructions, see USB Software: Firmware Update. For information about the MCU boot up sequence, see Hardware: Configuration Circuit: MCU.)

The source code for the MCU firmware (“Flash”) is provided in D:\ Config_Section_Code\MCU\FLASH
The compiler used to compile these files is Keil Studios MicroVision 2.11. Dini Group no longer encourages users to modify the firmware, but provides this code for informational and historical reasons. If you need special behaviour, please contact support@dinigroup.com and request a modification.

1.2.2 Configuration FPGA
The MCU unit controls all of the configuration circuits on the DN8000K10, but it does not have sufficient IO to access all of the configuration signals. For IO expansion, the MCU’s external memory bus is connected to a Spartan 2 FPGA. This FPGA provides a memory-mapped interface to all of its IO. This bus is called the ‘Config Bus’.

The configuration FPGA is connected to all of the configuration signals of the Virtex 4 FPGAs, the temperature sensors, status LEDs, CompactFlash card, reset buttons, RS232 port, clock synthesizer control signals, global clock multiplexer control signals, FPGA clock inputs, the Main Bus.

The source code for the Configuration FPGA is provided in D:\ Config_Section_Code\ConfigFPGA
The code was compiled using Xilinx ISE version 7.1i SP4. Dini Group does not encourage users to modify the “Spartan” firmware. The code is provided for informational purposes. If you need special behaviour, contact support@dinigroup.com and request it to become a standard feature.

1.2.3 Configuration Register Map
The DN8000K10PSX firmware is updated constantly to add compatibility for new products and add features. The information in this section may change after this manual is printed. The memory space of the MCU is 16 bits wide.
This table describes registers within the Configuration FPGA that are accessible from the memory space of the MCU.

<table>
<thead>
<tr>
<th>REGISTER</th>
<th>ADDRESS</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPC_RS232_12SELECT</td>
<td>DF0D</td>
<td>On the DN8000K10PSX, this register has no effect</td>
</tr>
<tr>
<td>PPC_RS232_34SELECT</td>
<td>DF0E</td>
<td>On the DN8000K10PSX, this register has no effect</td>
</tr>
<tr>
<td>FPGA_ERROR</td>
<td>DF14</td>
<td></td>
</tr>
<tr>
<td>HOLD_DONES</td>
<td>DF22</td>
<td></td>
</tr>
<tr>
<td>FPGA_FREQ_H</td>
<td>DF24</td>
<td></td>
</tr>
<tr>
<td>FPGA_FREQ_SEL</td>
<td>DF25</td>
<td></td>
</tr>
<tr>
<td>FPGA_FREQ_L</td>
<td>DF26</td>
<td></td>
</tr>
<tr>
<td>MCU_STUFFING1</td>
<td>DF27</td>
<td></td>
</tr>
<tr>
<td>MCU_STUFFING2</td>
<td>DF28</td>
<td></td>
</tr>
<tr>
<td>ACLK_N_VAL</td>
<td>DF29</td>
<td>Sets the divider value of GCLK0</td>
</tr>
<tr>
<td>ACLK_M_VAL</td>
<td>DF30</td>
<td>Sets the multiplier of GCLK0</td>
</tr>
<tr>
<td>BCLK_N_VAL</td>
<td>DF31</td>
<td>Sets the divider value of GCLK1</td>
</tr>
<tr>
<td>BCLK_M_VAL</td>
<td>DF32</td>
<td>Sets the multiplier of GCLK1</td>
</tr>
<tr>
<td>DCLK_N_VAL</td>
<td>DF33</td>
<td>Sets the divider value of GCLK2</td>
</tr>
<tr>
<td>DCLK_M_VAL</td>
<td>DF34</td>
<td>Sets the multiplier of GCLK2</td>
</tr>
<tr>
<td>BOARD_VERSION_DUP</td>
<td>DF46</td>
<td>Disables Main Bus interface</td>
</tr>
<tr>
<td>FPGA_COMMUNICATION</td>
<td>DF39</td>
<td></td>
</tr>
<tr>
<td>CHECKSUM</td>
<td>DF45</td>
<td>A checksum of USB configuration data</td>
</tr>
<tr>
<td>TEMP_SENSOR_A</td>
<td>DF50</td>
<td>Temperature of FPGA A</td>
</tr>
<tr>
<td>TEMP_SENSOR_B</td>
<td>DF51</td>
<td>Temperature of FPGA A</td>
</tr>
<tr>
<td>TEMP_SENSOR_C</td>
<td>DF52</td>
<td>Temperature of FPGA A</td>
</tr>
<tr>
<td>SERIAL_NUM_ADDR</td>
<td>DFFA</td>
<td></td>
</tr>
<tr>
<td>SPARTAN_MKS_VERSN_ADDR</td>
<td>DFFB</td>
<td></td>
</tr>
<tr>
<td>SPARTAN_VERSION_ADD</td>
<td>DFFD</td>
<td></td>
</tr>
<tr>
<td>BOARD_VERSION_NEW</td>
<td>DFFE</td>
<td></td>
</tr>
<tr>
<td>BOARD_VERSION</td>
<td>DFFF</td>
<td></td>
</tr>
</tbody>
</table>

1.2.4 Vendor Request List

The USB Program is updated constantly to add compatibility to new products and to add features. There may be changes to the application after this manual is printed that affect this section.

The following table describes the USB interface presented to the host by the MCU microcontroller. The USB Device identification numbers are Vendor: 0x1234, Device: 0x1234.

<table>
<thead>
<tr>
<th>Vendor Request Name</th>
<th>ID Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VR_UPLOAD</td>
<td>0xc0</td>
<td>Does nothing on DN8000K10</td>
</tr>
<tr>
<td>VR_DOWNLOAD</td>
<td>0x40</td>
<td>Downloads data to the Cypress EPROM, or to RAM</td>
</tr>
<tr>
<td>VR_ANCHOR_DLD</td>
<td>0xa0</td>
<td></td>
</tr>
</tbody>
</table>
1.2.5 USB Reference Design Control

1.2.6 Main Bus accesses

The USB Controller control the DN8000K10 reference design using USB vendor requests and bulk transfers that access the configuration FPGA’s registers. These registers cause “Main Bus” transactions with the user FPGAs. All Main Bus transactions are initiated by the configuration FPGA. To see a specification of the Main Bus interface, see Reference Design.

To request a Main Bus interface write transaction, the USB Controller program sends a USB bulk write to EP2 (endpoint 2). The first byte contains a code, either 0x00 or 0x01, determining whether the next 4 bytes contain an address or a datum. If this byte is a 0x00, the next 4 bytes in the bulk transfer are stored into an address register. All data transferred to and from the main bus is LSB first. The address 0x12345678 should be sent as a bulk transfer of 5 bytes: 0x00, 0x78, 0x56, 0x34, 0x12. To send a datum, send the code 0x01, followed by 4 bytes, LSB first. When the DN8000K10 receives a data word, it sends it onto the main bus interface to the address in the address register. It then increments the address register. Therefore, to send two words over main bus, 0x00000001 to address 0x00000001 and 0x00000002 to address 0x00000002, the USB Controller would send the following 15 bytes to USB EP2:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VR_EEPROM</td>
<td>0xa2 Loads (uploads) EEPROM</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
</tbody>
</table>

1.2.5 USB Reference Design Control

1.2.6 Main Bus accesses

The USB Controller control the DN8000K10 reference design using USB vendor requests and bulk transfers that access the configuration FPGA’s registers. These registers cause “Main Bus” transactions with the user FPGAs. All Main Bus transactions are initiated by the configuration FPGA. To see a specification of the Main Bus interface, see Reference Design.

To request a Main Bus interface write transaction, the USB Controller program sends a USB bulk write to EP2 (endpoint 2). The first byte contains a code, either 0x00 or 0x01, determining whether the next 4 bytes contain an address or a datum. If this byte is a 0x00, the next 4 bytes in the bulk transfer are stored into an address register. All data transferred to and from the main bus is LSB first. The address 0x12345678 should be sent as a bulk transfer of 5 bytes: 0x00, 0x78, 0x56, 0x34, 0x12. To send a datum, send the code 0x01, followed by 4 bytes, LSB first. When the DN8000K10 receives a data word, it sends it onto the main bus interface to the address in the address register. It then increments the address register. Therefore, to send two words over main bus, 0x00000001 to address 0x00000001 and 0x00000002 to address 0x00000002, the USB Controller would send the following 15 bytes to USB EP2:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VR_EEPROM</td>
<td>0xa2 Loads (uploads) EEPROM</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
</tbody>
</table>

1.2.5 USB Reference Design Control

1.2.6 Main Bus accesses

The USB Controller control the DN8000K10 reference design using USB vendor requests and bulk transfers that access the configuration FPGA’s registers. These registers cause “Main Bus” transactions with the user FPGAs. All Main Bus transactions are initiated by the configuration FPGA. To see a specification of the Main Bus interface, see Reference Design.

To request a Main Bus interface write transaction, the USB Controller program sends a USB bulk write to EP2 (endpoint 2). The first byte contains a code, either 0x00 or 0x01, determining whether the next 4 bytes contain an address or a datum. If this byte is a 0x00, the next 4 bytes in the bulk transfer are stored into an address register. All data transferred to and from the main bus is LSB first. The address 0x12345678 should be sent as a bulk transfer of 5 bytes: 0x00, 0x78, 0x56, 0x34, 0x12. To send a datum, send the code 0x01, followed by 4 bytes, LSB first. When the DN8000K10 receives a data word, it sends it onto the main bus interface to the address in the address register. It then increments the address register. Therefore, to send two words over main bus, 0x00000001 to address 0x00000001 and 0x00000002 to address 0x00000002, the USB Controller would send the following 15 bytes to USB EP2:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VR_EEPROM</td>
<td>0xa2 Loads (uploads) EEPROM</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
</tbody>
</table>

1.2.5 USB Reference Design Control

1.2.6 Main Bus accesses

The USB Controller control the DN8000K10 reference design using USB vendor requests and bulk transfers that access the configuration FPGA’s registers. These registers cause “Main Bus” transactions with the user FPGAs. All Main Bus transactions are initiated by the configuration FPGA. To see a specification of the Main Bus interface, see Reference Design.

To request a Main Bus interface write transaction, the USB Controller program sends a USB bulk write to EP2 (endpoint 2). The first byte contains a code, either 0x00 or 0x01, determining whether the next 4 bytes contain an address or a datum. If this byte is a 0x00, the next 4 bytes in the bulk transfer are stored into an address register. All data transferred to and from the main bus is LSB first. The address 0x12345678 should be sent as a bulk transfer of 5 bytes: 0x00, 0x78, 0x56, 0x34, 0x12. To send a datum, send the code 0x01, followed by 4 bytes, LSB first. When the DN8000K10 receives a data word, it sends it onto the main bus interface to the address in the address register. It then increments the address register. Therefore, to send two words over main bus, 0x00000001 to address 0x00000001 and 0x00000002 to address 0x00000002, the USB Controller would send the following 15 bytes to USB EP2:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VR_EEPROM</td>
<td>0xa2 Loads (uploads) EEPROM</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
<tr>
<td>VR_RAM</td>
<td>0xa3 Loads (uploads) external ram</td>
</tr>
<tr>
<td>VR_SETI2C_ADDR</td>
<td>0xa4</td>
</tr>
<tr>
<td>VR_GETI2C_TYPE</td>
<td>0xa5 8 or 16 byte address</td>
</tr>
<tr>
<td>VR_GET_FLASH_REV</td>
<td>0xa6 Returns a revision code of the DN8000K10 MCU firmware</td>
</tr>
<tr>
<td>VR_GET_FPGA_INFO</td>
<td>0xa7</td>
</tr>
<tr>
<td>VR_RENUM</td>
<td>0xaa The Cypress MCU behaves as if it were removed and reconnected to USB.</td>
</tr>
<tr>
<td>VR_DB_FX</td>
<td>0xa9 Force use of double byte address EEPROM (for FX)</td>
</tr>
</tbody>
</table>
Note that the number of bytes sent to EP2 must be divisible by 5.

To request a main bus read operation, the USB Controller sends a USB bulk write to EP2 to set the address register, as described in the above paragraph. Then, the USB Controller sends a bulk read to EP6 (endpoint 6), with the USB bulk request SIZE field set to the number of bytes requested. The number requested must be divisible by 4. After the bulk read is complete, the address register is incremented by SIZE/4. Read and write transactions use the same

Before starting a USB read, or series of reads, you should set the size of the Cypress USB read buffer to be equal to the size of the bulk transfer. This can be accomplished using the VR_SET_EP6TC (0xBB) vendor request described in the Vendor Requests section. If this step is skipped, you may experience slow USB response, or even system instability, depending on the operating system.

1.2.7 Configuration
To access the 16 FPGA configuration interface (SelectMap), a USB interface is provided using Vendor Requests and bulk transfers. The basic configuration process is as follows:

USB Controller sends VR_SETUP_CONFIG (see Vendor Requests) with data representing which FPGA to configure. (F0 is 0x01, F1 is 0x02, F2 is 0x03…)

MCU on receiving this vendor request sets the PROG signal of the selected FPGA. This resets the FPGA and clears any configuration data it may already have. This Vendor request also selects the FPGA, so that SelectMap bus activity only affects the selected FPGA. Bulk transfers initiated after this command are interpreted as SelectMap transfers, rather than Main Bus transfers. (See Main Bus access above). This will be so until vendor request VR_SETUP_END is called.

USB Controller sends a bulk write USB request to EP2. Each byte of data in the bulk write is sent to the selected FPGA over the SelectMap bus, and the FPGA signal CCLK is pulsed once for each byte of data sent. For more on the SelectMap interface, see Hardware: Configuration: SelectMap. Note that the LSBit in the USB transaction is sent to the LSBit in the SelectMap interface, so bit swapping as described in the Virtex 4 Configuration Guide UG071 is not required. A standard .bit file from Xilinx bitgen can be transferred in binary over this USB interface to correctly configure an FPGA on the DN8000K10.

Unless the HOLDDONES option has been activated, the Virtex 4 FPGA will activate, following the activation command imbedded in the .bit stream file. The DONE signal will go high, lighting the green LED next to the FPGA labeled “FPGA Done”.
The USB Controller sends a vendor request VR_SETUP_END. This request deselects the FPGA, so that further bulk requests are interpreted as Main Bus transactions. See Main Bus accesses.

1.2.8 Readback
Not recommended over SelectMap. Suggest Xilinx ChipScope Pro, which work over JTAG.

2 PCI AETEST Application

AETEST utility program can test and verify the functionality of the DN8000K10PSX Logic Emulation board, and provide data transfer to and from the User design.

All AETEST source code is included on the CD-ROM shipped with your DN8000K10PSX Logic Emulation kit. AETEST can be installed on a variety of operating systems, including:

- DOS and Windows 95/98/ME using DPMI (DOS Protected Mode Interface)
- Windows 98/ME using a VxD driver
- Windows 2000/XP (Windows WDM)
- Windows NT
- Linux
- Solaris

2.1 Functionality
The AETEST utility program contains the following tests:

- PCI Test
- Memory Tests (SRAM & DDR)
- FLASH Test
- Daughter Card Test (with or without cables)
- BAR Memory Range Tests

AETEST also provides the user with the following abilities:

- Recognize the DN8000K10PSX
- Display Vendor and Device ID
- Set PCI Device and Function Number
- Display all configured PCI devices
- Various loops for PCI device-function and ID numbers
- Write and Read Configuration DWORD
- Write DWORD, Read DWORD and Write/Read DWORD (Same Address)
- BAR Memory Fill, Write and Display
- Configure/Save BAR’s from/to a file

2.2 Running AETEST

2.3 Compiling AETEST

There are two versions of AETest, one that controls the DN8000K10PSX from a PCI bus, and the other that controls the DN8000K10PSX over a USB connection. The source for the PCI version is found on the User CD

D:\aetest\aetest

The source for the usb version is found on the User CD

D:\aetest_usb\aetest

You will likely want to interface to your ASIC emulation project using PCI. You may want to start your controller software by modifying and recompiling AETEST.

2.3.1 Compiling AETest for DOS

The DOS version of AETest requires DJGPP. You can find it at

http://www.delorie.com/djgpp/

Follow the installation instructions for DJGPP. The download comes with an utility to set your environment variables correctly.

D:\AETest\aetest

Contains the source code for AETest. Copy this directory to your hard drive. Open the file Makefile. This file must be edited to define which operating system you wish to target.

Uncomment the line

#DESTOS = DOS_DJGPP

and the line

include Makefile.make

In a DOS shell, run make
2.3.2 Compiling AETest for Windows XP
AETest for Windows requires visual studio to compile.

Copy the directory on the User CD
D:\AETest\aetest
to your local machine. Open the file “Makefile” and uncomment the lines
#DESTOS = WIN_WDM
and
include Makefile.make
Run make.

3 Updating the Firmware

Dini Group may release firmware bug fixes or added features to the DN8000K10PSX. If a firmware update is released you will need to

There are two firmware files that Dini Group may release, the first is a Micro controller (MCU) software update that is stored in a flash memory. This update can be accomplished easily from within the USBController application.

The second update that may be required is a Spartan FGPA core update. The configuration data for the Spartan FPGA is contained in a Xilinx configuration PROM. This update can be accomplished with the Xilinx JTAG programming program, iMpact.

3.1 Updating the MCU (flash) firmware

To protect against accidental erasure, the MCU firmware cannot be updated unless the board is put in firmware update mode during power-on. Find Switch S6 (“User Reset”) on the DN8000K10PSX.
Hold down the “User reset” Switch while the DN8000K10PSX powers on. Or alternately, while holding down the “User reset” switch, press the “Hard reset” button next to it. The DN8000K10PSX samples the user reset button on power on to boot into firmware update mode.

Open the USB Controller program. If the DN8000K10PSX powered on in firmware update mode, there will be an “Update Flash” button near the top of the USB Controller window. Click on this button.

When the Open… dialog box appears, navigate to the Firmware image file supplied by Dini Group. The file name should be “flash_flp.hex”. Press OK.

The USB Controller should take about 10 seconds while the firmware update is taking place. (The user controls may freeze up while this is occurring) When the download is complete, the Log window should print, “Update Complete”

Power cycle the board.
3.2 Updating the Spartan (EEPROM) firmware

Connect a Xilinx Parallel IV configuration cable to the parallel port of your computer. You can also use a Xilinx Platform USB cable. The Parallel IV cable requires external power to operate, so you may need to connect the keyboard connector power adapter. When the Parallel IV cable has power, the status LED on Parallel IV turns amber.

Use a 2mm IDC cable to connect the Parallel IV cable to the DN8000K10PSX connector J14. This connector is labeled “Firmware”

![Figure 16 Firmware Update Header](image)

Power on the DN8000K10PSX. When the Parallel IV cable is connected to a header, the status light turns green.

Open the Xilinx program Impact, usually found at Start->programs->Xilinx ISE 7.1->Accessories->impact

Choose the menu option File->Initialize Chain. (You may need to create a new project for this menu option to be available)

Impact should detect 2 devices in the JTAG chain. Xc18v02 and Xc2s200. For each item in the chain Impact will direct you to select a programming file for each. For the xc18v02 device, select the Spartan Firmware update file provided by Dini Group. This file should be named prom.mcs. Hit Open. Impact will then ask for a programming file to program the xc2s200. Press Bypass.
To program the prom. Right-click on the prom and select Program… from the popup menu. In the options dialog that follows, the options “Erase before programming” should be selected, and “Verify” should be deselected. Press OK. The programming process takes about 35 seconds over the parallel port.

Power cycle the DN8000K10PSX. The new firmware is now loaded. You can close impact and disconnect the Parallel IV cable.
Chapter 4: Hardware

1 Overview

The DN8000K10PSX was designed to maximize the number of useful gates in your emulation project running at speed by providing the densest interconnect possible. To achieve this goal, the DN8000K10PSX is equipped with the highest-capacity FPGAs available today, the Xilinx Virtex 4 LX200. The FPGAs on the DN8000K10PSX are in the largest, 1513-ball package to give the user extremely high IO count, for high bandwidth and low-latency interconnect between FPGAs. Three hundred eighty nine differential links between FPGAs A and B allow for as much as 189 Gb/s communication between the two FPGAs.

In order to support enough bandwidth to deliver real time data to your design at speed, the DN8000K10PSX is equipped with an optional Xilinx Virtex 4 FX100 with RocketIO Multi-Gigabit Tranceivers. Serial connections over Fibre, Coax ribbon cable, and Coax SMA cables allow for a total aggregate 150 Gb/s off-board communication.

Monitoring your design and supplying test vectors is simple with an onboard PCI interface bridge chip, capable of full 66Mhz, 64-bit PCI while reserving 100% of the FPGA fabric for user logic. The PCI interface is taken care of. You will be able to communicate to the DN8000K10PSX into a PCI slot right out of the box.

To allow you to connect the FPGA to the resources that will be on your end product, the DN8000K10PSX also has highspeed expansion capabilities.

Below is a block diagram of the DN8000K10PSX
Figure 18: DN8000K10PSX Block Diagram
The following sections describe in detail each circuit on the DN8000K10PSX. Note that Schematics appearing in this section are illustrative and may have had details omitted or have been modified for clarity and brevity. If you need to probe, modify or design around the DN8000K10PSX you will need to examine the complete schematics. This have been provided for you in PDF form on the User CD.

## 2 Configuration Circuit

### 2.1 Overview

The goal of the configuration circuit on the DN8000K10PSX is to allow the user to configure his FPGAs using any host interface. The configuration system on the DN8000K10PSX allows configuration over PCI, USB, JTAG, or automatic configuration from a CompactFlash card.

The circuit is designed to provide an easy configuration solution that will work out-of-the-box for most users. For special configuration requirements, the configuration circuitry is programmable. The verilog code for the configuration FPGA and the C code for the microcontroller are both provided on the reference CD. The C code for the PCI controller program and USB Windows GUI controller program are also included on the User CD.

### 2.2 The Spartan 2 FPGA

The configuration circuitry of the DN8000K10PSX is built around a Xilinx Spartan II Fpga. The SelectMap interface of the user FPGAs is connected directly to the general purpose IOs of the Spartan 2, allowing the maximum flexibility of configuration. The Spartan 2 also shares connectivity with the three user FPGAs over a 40-bit Main bus, allowing fast transfers from a computer to the user design over PCI or USB. The Spartan 2 FPGA also provides IO expansion for the Cypress Microcontroller. The Spartan II FPGA comes preloaded with a core that provides a way to program the Virtex 4 FPGAs over PCI, USB and CompactFlash.
The Spartan FPGA is connected to the Cypress microcontroller’s address and data buses, and the control registers within the Spartan II FPGA that control FPGA configuration are memory-mapped into the MCU’s address space.

Figure 19 Spartan II IO Connections

### 2.2.1 Spartan Configuration

The Spartan 2 FPGA is configured from a Xilinx serial prom. The Spartan’s configuration mode is hard-wired into Master Serial mode. After power up, the Spartan automatically clocks an external PROM, U41, which programs the FPGA over the serial configuration data pin DIN.

A green LED, DS24, lights when the DONE pin is high. This signal is driven by the Spartan 2 FPGA when it is configured and running.

Both the Spartan and the serial prom are connected in a JTAG chain attached to J14. This header is used when performing firmware updates to update the PROM.
As soon as the Spartan II FPGA is configured, it resets the Cypress microcontroller. Pull-downs on the PROG pin of FPGAs A B and C ensure that the FPGAs cannot be active unless the Spartan II is successfully configured.

### 2.2.2 CompactFlash

The CompactFlash card interface is connected to the IOs of the Spartan 2 FPGA.

The CompactFlash data bus, D[0-7], also connects to the microcontroller. Currently the MCU connection is not used. The Microcontroller is able to read from the CompactFlash interface by accessing the Spartan’s memory-mapped data over the MCU memory interface for the purposes of reading instructions from CompactFlash cards.
For instructions on creating a CompactFlash card for configuring the DN8000K10PSX, see the section Configuration Options: CompactFlash.

### 2.2.3 MCU communication
The MCU communicates to the Spartan 2 FPGA over its external memory interface, pins D0:7 and A0:15. The Spartan 2 is assigned the address range 0xDF00 to 0xDFFF in the Microcontrollers memory space.

The 480Mbs data rate of USB 2.0 is too fast for the microcontroller to control, so the MCU’s hardware passes USB bulk transfer data to the MCU GPIF interface. These signals, SM[0-7] and GPIF_CTL, GPIF_RDY, connect to the Spartan FPGA. The SM[0-7] signals also connect to the CompactFlash card socket, although the MCU does not communicate with the CompactFlash interface directly. The MCU_IFCLK signal provides a clock for this interface. The signal is driven from the Spartan 2 FPGA.

### 2.2.4 PCI communication
To enable configuration over PCI, the Spartan 2 is connected to a subset of the Quicklogic 5064 PCI interface. Due to the available number of IOs on the Spartan 2 FPGA, only 32 of the 64 data bits in the Quicklogic interface are connected to IOs on the Spartan 2. Also, the Spartan 2 does not connect to any of the signal required for DMA operation over PCI.

From the PCI host perspective, the address range BAR0 is directed to the Spartan 2 FPGA, and BAR1-BAR7 is directed towards the Virtex 4 FPGA A.

Since FPGA A and the Spartan 2 both can access the Quicklogic 5064 back-end, only one must communicate with the QL5064 at a time. The signals SP_PCI_REQ and LX_PCI_ACK are used to control communication with the QL5064.

### 2.2.5 RS232
The DN8000K10PSX has two RS232 headers. One (P7) is used by the microcontroller unit to provide configuration feedback and control. The other (P6) is connected to the Spartan 2 FPGA. The Spartan 2 FPGA passes the RX and TX lines through to MB[36] and MB[37], respectively. It is up to user designs to avoid contention on the TX line (MB[37]) - only one FPGA may transmit on RS232 at a time, and all other FPGAs must tristate the TX line during the transmission.

Since RS232 uses a 12V signal levels, the RS232 signals from the SpartanII are first buffered through a voltage translation buffer shown below.
2.2.6 IIC

There is a single IIC bus on the DN8000K10PSX connecting all IIC enabled chips on the board. On this bus are three MAX1617A temperature sensing chips (U3, U4, U24), two DDR2 SODIMM sockets, and a serial eprom. The temperature sensors on the IIC bus are polled about once per second by the MCU to read the temperature of each FPGA.

2.3 Configuration Options

The DN8000K10PSX allows FPGA configuration from any of four methods.

When a Virtex 4 FPGA is configured, the DONE pin on the FPGA is pulled high. The DN8000K10PSX has a green LED attached to the DONE signal of each to indicate the state of the DONE pin on the three Virtex 4 FPGAs and on the SpartanII configuration FPGA.

2.3.1 Jtag

Jtag is the only configuration method on the DN8000K10PSX that does not use the Virtex 4 SelectMap configuration interface. When programming the user FPGAs over a JTAG cable plugged into J11, the DN8000K10PSX configuration circuitry is not used.

A JTAG connection is required to use some Xilinx configuration tools like ChipScope, and readback from Impact. Configuration over JTAG is slower than SelectMap. You can still use the
CompactFlash or USB interfaces to control clock settings if you plan to configure through JTAG.

To configure using JTAG, we recommend using Xilinx Parallel cable IV, or Xilinx platform USB cable. The Xilinx program. You should set the configuration speed of your JTAG cable to 4MHz or below.

The JTAG signals TMS is bussed to all three Virtex 4 FPGAs. TDO connects to FPGA A, the TDO of FPGA A is connected to TDI of FPGA B, the TDO of FPGA B connects to the TDI of FPGA C and TDO of FPGA C is connected to the TDI of J11. TCK is buffered and passed to each FPGA in a point-to-point fashion.

The INITn signal is not used.
If you ordered your DN8000K10PSX with one or more FPGAs not installed (Option FPGA A NONE, FPGA B NONE, or FPGA C NONE) then a bypass resistor is installed connecting the TDI pin to the TDO pin of the uninstalled FPGA. This is so the JTAG chain will remain intact when FPGAs are missing.

### 2.3.2 CompactFlash

When the DN8000K10PSX powers on, the microcontroller reads the contents of any CompactFlash card that is in the CompactFlash slot. The microcontroller by default opens a file on the root directory named “Main.txt” if it exists. This file contains instructions for the configuration circuitry to configure the Virtex 4 FPGAs.

To create a CompactFlash card to control the DN8000K10PSX configuration, insert the CompactFlash card into a card reader (provided) and connect it to a PC. Create a file on the root directory of the card and call it “Main.txt”

In main.txt, write a series of configuration commands, separated each by a new line. A valid command is one of the following:

```
// <comment>
FPGA A:<filename>
FPGA B:<filename>
FPGA C:<filename>
CLOCK FREQUENCY: <clockname> N <number> M <number>
SANITY CHECK: <yn>
VERBOSE LEVEL: <level>
MEMORY MAPPED: 0x<SHORTADDR> 0x<BYTE>
MAIN BUS 0x<WORDADDR> 0x<WORDDATA>
```
<comment> can be any string of characters except for newline.
<filename> can be the name of a file on the root directory of the CompactFlash Card.
<number> can be any one or two digit positive integer in decimal
<clockname> can be [G0,G1,G2] G0 is GCLK0, G1 is GCLK1, G2 is GCLK2.
<yn> can be the letter y or the letter n
<level> can be 0,1,2 or 3
<SHORTADDR> is a 2-digit hex number (16 bits)
<BYTE> is a 1-digit hex number (8 bits)
<WORDADDR> 4-digit (32 bit) hex number representing a main bus address
<WORDDATA> 4-digit (32 bit) hex number containing data for a main bus transaction

The following table describes the function of each of the available main.txt commands.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>// &lt;comment&gt;</td>
<td>The MCU performs no operation and moves to the next command.</td>
</tr>
<tr>
<td>VERBOSE LEVEL: &lt;level&gt;</td>
<td>This command will set the amount of output the MCU will produce over the RS232 port during configuration. When level is set to 0, the MCU will produce only error output. Before this command is executed, the level is set to the default value 3.</td>
</tr>
<tr>
<td>FPGA A:&lt;filename&gt;</td>
<td>The Virtex 4 FPGA “A” will be configured with the file named by &lt;filename&gt;</td>
</tr>
<tr>
<td>FPGA B:&lt;filename&gt;</td>
<td>The Virtex 4 FPGA “B” will be configured with the file named by &lt;filename&gt;</td>
</tr>
<tr>
<td>FPGA C:&lt;filename&gt;</td>
<td>The Virtex 4 FPGA “C” will be configured with the file named by &lt;filename&gt;</td>
</tr>
<tr>
<td>SANITY CHECK: &lt;yn&gt;</td>
<td>If &lt;yn&gt; is set to y, then the MCU will examine the headers in the .bit files on the CompactFlash card before using them to configure each FPGA. If the target FPGA annotated in the .bit file header is not the same type as the FPGA the MCU detects on the board, it will reject the file and flash the error LED. Before this command is executed, &lt;yn&gt; is set to the default value y.</td>
</tr>
<tr>
<td></td>
<td>If you want to encrypt of compress your bit files, you will need to set &lt;yn&gt; to n. Encrypting bit files is not supported or recommended by Dini Group. Previous revisions of Xilinx parts have been vulnerable to permanent damage caused by bugs in the encryption circuitry.</td>
</tr>
<tr>
<td>MAIN BUS 0x&lt;WORDADDR&gt; 0x&lt;WORDDATA&gt;</td>
<td>Writes data in &lt;WORDDATA&gt; to the address on the main bus interface at &lt;WORDADDR&gt;. This command only makes sense in the context of the Dini Group reference design, unless your design implements a compatible controller on the main bus pins. The Specification for this interface is in the</td>
</tr>
</tbody>
</table>
An example main.txt file:

```
VERBOSE LEVEL:0
   // This will prevent the MCU output over RS232 to speed up configuration
FPGA A:a.bit
   // this will load the configuration a.bit into FPGA A
CLOCK FREQUENCY: G0 N 4 M 10
   // This will cause GCLK0 frequency to be
   // 25*10=250 / 4 = 62.5Mhz
MAIN BUS: 0x0000 0x0001
   // Writes to a register in FPGA A.
```

Even if you are not planning to configure your Virtex 4 FPGAs using a CompactFlash card, you may want to leave a CompactFlash card in the socket to automatically program your global and rocketIO clock. (Clocks may also be programmed using the provided USB application, or over the MCU RS232 terminal.)

### 2.3.3 USB

The USB interface on the DN8000K10PSX is provided by the Cypress microcontroller unit. The Cypress microcontroller is programmed to interrupt when it receives a USB vendor request.

When the MCU receives over USB a Bulk Transfer type request, it does not interrupt. The raw data contained in the bulk transfer is driven out on the GPIF pins of the MCU (the SM[0-7] signals) to the Spartan 2. The data is clocked out using the MCU_IFCLK clock signal to the Spartan 2. As long as the signal GPIF_CTL is held high by the MCU, the Spartan 2 clocks MCU_IFCLK to receive the USB data.
When data is written to the Spartan 2 from a bulk transfer over the MCU’s GPIF interface, the Spartan 2 either writes that data onto the SelectMap interface of the Vitex4 FPGAs, or onto the Main bus using the Main Bus interface described in the Reference Design chapter.

The control register FPGA_SELECT within the Spartan 2 determine to which interface this data is routed to.

2.3.4 PCI

The PCI interface on the DN8000K10PSX is primarily used directly by the Virtex 4 FPGA. The Quicklogic 5064 implements the PCI interface and delivers the data directly over the QL interface to FPGA A. However, when the host machine makes a read or write transaction to the DN8000K10PSX on a BAR0 address over PCI, the data is instead delivered to the Spartan 2 configuration FPGA. This allows the Spartan 2 to configure FPGAs, change other configuration settings, or communicate to FPGAs B and C over PCI.

The program AETest, supplied on the User CD along with the AETest source code, allows the user to configure FPGAs and change configuration settings over PCI. When used with the Dini Group reference design, or if the user has created a compatible Main Bus interface, the AETest program can also communicate directly to the FPGA A, B and C designs.

2.3.4.1 Configuration

To configure an FPGA over PCI, the host program writes the following instructions to BAR0 of the DN8000K10PSX.

First, the host instructs the Spartan 2 FPGA to “select” an FPGA for configuration. Write one word of data to BAR0, 0x0208. The data 0x11 represents FPGA A, 0x12 is FPGA B, 0x13 is FPGA C.

Next, the host instructs the Spartan 2 FPGA to assert the PROGn signal (Reset) of the selected FPGA’s selectmap interface. This causes the FPGA to unconfigure regardless of the reconfiguration setting made by the bit file’s reconfigure setting. The PROGn signal must be asserted once before the FPGA is configured for the first time. Write a word to the address BAR0, 0x208. The data word 0x11 represents FPGA A, 0x12 is FPGA B, 0x13 is FPGA C.

When the FPGA is read to configure, it pulls the selectmap signal INTn low. The configuration process should pause until this occurs. To read the current value of INTn on the selected FPGA, read from the BAR0 address 0x208. Bit 5 represents the value of the selected INTn signal.

After the INTn signal is detected, the Host should de-assert PROGn (Reset). Write to BAR0 address 0x208 the data word representing the selected FPGA. 0x11 is FPGA A, 0x12 is FPGA B, 0x13 is FPGA C.

The configuration stream for the FPGA is then sent to BAR0, address 0x210, one byte at a time. Some time during the configuration stream byte loading process, a startup sequence is sent
to the FPGA and the FPGA becomes operational. This startup sequence is contained within the bit file.

To determine if the selected FPGA is currently configured (ie configuration was successful), read from BAR0 address 0x208. The bit 5 contains the state of the DONE FPGA pin, the bit 6 contains the state of the FPGA INIT signal.

By convention, the host program should leave the Spartan in the FPGA deselected state. To deselect the FPGA, write to BAR0, address 0x208 the data 0x10. (FPGA SELECT NONE)

2.3.4.2 Config Space

PCI can also be used to control other configuration functions on the DN8000K10, such as temperature sensors and clocks. This is done by altering the data in the XDATA memory space of the configuration MCU.

To write to the MCU address space, access the DN8000K10’s BAR0 at the address MCU_BAR_ADDR 0x258. Send a 32-bit word of data. This data is decoded as follows:

- Bits 31-16: address in the XDATA space. (only addresses 0xDF00-0xDFFF reside in the Spartan 2)
- Bits 15-8: Ignored
- Bits 7-0: The Data to write

To read from the MCU address space, access the DN8000K10’s BAR0 at the following 32-bit address:

- Bits 31-24: The DN8000K10PSX’s BAR0
- Bits 23-16: the lower 8 bits of XDATA address you would like to read. This corresponds to addresses 0xDF00-0xDFFF of the XDATA address. (Only addresses 0xDF00-0xDFFF reside in the Spartan 2)
- Bits 15-0: 0x0260

2.3.4.3 Main Bus Space

PCI can also be used to send information to and from your Virtex 4 user design through the Spartan 2 FPGA. Communication directly to the user design can also be accomplished from FPGA A by communicating directly with the QL PCI backend interface. This method is an order of magnitude faster, and allows the use of advanced PCI features like DMA. See Hardware: PCI interface. Communication with the FPGAs through the Spartan occurs using the Main Bus interface. For information on the main bus interface see Reference Design: Main Bus Interface.
To write to the main bus interface, write to BAR0 address QLPCI_REG_MBADDR with the 32-bit value representing the main bus address you would like to write to. Then, write a second PCI write to address QLPCI_REG_MBWRDATA with 32-bit data representing the data that you would like to write to main bus. After the Spartan 2 has received a write to both the MBADDR and MBWRDATA registers, it will write to the main bus interface.

To read from the main bus interface, first write to BAR0 address QLPCI_REG_MBADDR with the 32-bit value representing the main bus address you would like to read from. Then, read from BAR0, QLPCI_REG_MBRDDATA. The returned value will be the value read off the main bus at the selected address. When an error has occurred (No FPGA responded to the read request) the Spartan will return the value 0xABCDABCD.

2.4 FPGA configuration Process
For information regarding the JTAG interface and configuration, See Xilinx publication UG071, Virtex 4 configuration guide.

When configuring over PCI, USB or CompactFlash, the FPGAs are configured over the Virtex 4 SelectMap bus.

All SelectMap signals are connected directly to the Spartan2 FPGA. The SelectMap signals are:

D[0-7] SelectMap data signals.

PROGRAM_B Active low asynchronous reset to the configuration logic. This will cause the FPGA to become unconfigured. The documentation refers to this signal as PROGn

DONE After the FPGA is configured, it is driven high by the FPGA.

INIT Low indicates that the FPGA configuration memory is cleared. After configuration, this could indicate an error.

RDWR_B Active low write enable. The Documentation refers to this signal as RDWR

BUSY When busy is high, the SelectMap configuration stream must stop until BUSY goes low.

CS_B SelectMap chip select. The documentation refers to this signal as CSn

CCLK Signals D[0:7], DONE, RDWR_B and CS_B are clocked on CCLK

Each Virtex 4 FPGA has a complete set of SelectMap signals connected point-to-point to the Spartan 2, except for FPGA B and C, who share signals D[0-7]. All signals are 2.5V CMOS signals except for D[0-7] of FPGA A (Signals SELECTMAP_3V_D[0-7]), which are 3.3V CMOS.
All commands required to configure a Virtex 4 FPGA are created and embedded in the .bit files created by the Xilinx Bitgen program. The DN8000K10PSX does not interact with the SelectMap interface other than to reset the FPGA using the PROGn-INTn-PROGn resetr sequence described in UG071, and to copy a bit stream file unaltered to the FPGA over the data pins D[7-0]. Select map commands can be issued to the Virtex 4 FPGA from the host using the same interface used to configure and FPGA.

After a Virtex 4 FPGA is configured, it asserts the signal DONE. On the DN8000K10PSX, these signals have an LED attached to each DONE signal placed near the upper corner of each FPGA.

FPGA A’s LED is DS18, B is DS14, C is DS16

If your Virtex 4 FPGA design is failing to produce the intended (or any) results, you should check the DONE light above the FPGA to make sure it is configured correctly. The design files created by Xilinx bitgen software contain a CRC check, so if the Virtex 4 FPGA detects a CRC failure, there was a transmission error during configuration and the DONE light will not glow. The DN8000K10PSX microcontroller also checks the design files you send to make sure they are compiled for the FPGAs that are installed on your board. If they are not, then the microcontroller unit halts the configuration process. As a result, when the DONE light goes on, you will know that the configuration process was successful.

2.5 MCU

The operation of the Spartan II is monitored and controlled by a Cypress CY7C68013 microcontroller. The microcontroller also has a USB 2.0 interface that can be used to monitor the board, control configuration, or transfer data to and from the user FPGA design. Basic operation can be controlled over an RS232 link from a computer terminal.

2.5.1 RS232

The primary method of user interaction with the DN8000K10PSX configuration circuitry is the MCU’s RS232 port (P7). The Cypress CY7C68013 has two RS232 pins that are buffered through a 12V voltage translation buffer for use with a standard computer serial port.
The RS232 port will be able to communicate with a standard PC serial port set to 19200 baud, 8 data bits, no parity, no handshaking. When you connect a computer terminal to the port and power on the DN8000K10PSX, the firmware loaded on the microcontroller unit will display a menu on the terminal. This menu will allow you to control the basic configuration options of the DN8000K10PSX including configuration and clock frequencies.

### 2.5.2 Clocks

The Cypress CY7C68013 is also responsible for configuring the global clocks and RocketIO clock of the DN8000K10PSX. The Cypress CY7C68013 MCU reads the file “main.txt” from the CompactFlash card in the socket (J24), and follows the users clock configuration commands.

The 3 ICS8442 clock synthesizers on the DN8000K10PSX used for generating the global clocks, GCLK0, GCLK1 and GCLK1, share a serial configuration bus connected to the MCU.
to program them. The ICS8442 frequency synthesizers are capable of multiplying and dividing the reference frequencies provided by their reference crystals. The MCU loads the user’s desired multiplication “M” value, and division, “N” value into the settings registers in the ICS8442 chip.

2.5.3 LEDs
The MCU is connected to 4 red LEDs that are visible from outside the PC case when the DN8000K10PSX is plugged into a PCI slot. The LEDs flash a status code during and after configuration.

All four flashing LEDs means there has been an error configuring at least one FPGA.

2.5.4 Memory space
The XDATA memory space of the MCU is partitioned into four sections.

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000 - 0x1FFF</td>
<td>internal data/program memory</td>
</tr>
<tr>
<td>0x2000 - 0xCFFF</td>
<td>external SRAM</td>
</tr>
<tr>
<td>0xDFF0 - 0xDFFF</td>
<td>memory mapped registers (no external memory accesses)</td>
</tr>
<tr>
<td>0xE000 - 0xFFFF</td>
<td>reserved by MCU, RD/WR strobes not active in this region</td>
</tr>
</tbody>
</table>

The internal data memory region is mapped to an internal SRAM in the Cypress MCU. When the microcontroller code calls memory access from this region, the external Address and Data busses are not used. After power on reset, the MCU reads from the IIC Eprom connected to the MCU_EPROM signals and fills this internal memory before allowing the PC to run. The code in this section of memory contains core functions of the Dini Group firmware, like setting up the interrupt registers, communicating with USB, and allowing firmware updates.

The external SRAM is used for heap data.

The memory mapped register region (The DF region) contains registers in the Spartan 2 FPGA that control FPGA configuration.

The program memory space of the MCU is directly mapped to the external Flash memory.

When the Cypress MCU is reset (which happens after the Spartan 2 is configured), it loads its boot code into its 8kB of internal memory from a serial EEPROM (U13). The code in the EPROM instructs the MCU to execute code located on the FLASH memory (U19). The code in the EEPROM and FLASH is located on the user CD.
Communication over the MCU memory bus to the Spartan 2 is synchronized to the 24Mhz MCU_CLK (X3). For information regarding the timing of transactions on this bus, see the Cypress CY7C68013 user manual.

The Configuration FPGA is connected to the MCU_DATA[7:0] signals, the MCU_ADDR[15:0] signals and the MEM_OE signal, allowing it to decode address accesses of the MCU. The Configuration FPGA is programmed to respond to accesses in the XDATA address space in the address range of 0xDF00 to 0xDFFF.

Communication over the MCU memory bus to the Config FPGA is synchronized to the 24Mhz MCU_CLK (X3). For information regarding the timing of transactions on this bus, see the Cypress CY7C68013 user manual.

The following registers implemented in the Configuration FPGA are accessible as part of the MCU’s XDATA address space.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>XDATA Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>DF00</td>
<td>Used when reading from SM but not configuring</td>
</tr>
<tr>
<td>COMMAND</td>
<td>DF01</td>
<td>Commands for the SM</td>
</tr>
<tr>
<td>Symbol</td>
<td>Address</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>ROW_LADDR</td>
<td>DF02</td>
<td>Holds lower 8-bits of SM address</td>
</tr>
<tr>
<td>ROW_HADDR</td>
<td>DF03</td>
<td>Holds upper 8-bits of SM address</td>
</tr>
<tr>
<td>ROW_XADDR</td>
<td>DF04</td>
<td>Holds extra bits of SM address</td>
</tr>
<tr>
<td>NUM_BYTES_0</td>
<td>DF05</td>
<td>Holds lower 8-bits of the number of bytes to read</td>
</tr>
<tr>
<td>NUM_BYTES_1</td>
<td>DF06</td>
<td>Holds upper bits of number of bytes to read in</td>
</tr>
<tr>
<td>BITS_1</td>
<td>DF07</td>
<td>BIT7: mcu_fpga_config_rd BIT6:</td>
</tr>
<tr>
<td>BITS_2</td>
<td>DF08</td>
<td>BIT4: FPGA_DONE BIT3 CPLD_idle BIT2:</td>
</tr>
<tr>
<td>SM_SIGNALS</td>
<td>DF09</td>
<td></td>
</tr>
<tr>
<td>MCU_XADDR</td>
<td>DF0A</td>
<td>Address register for upper FLASH/SRAM bits</td>
</tr>
<tr>
<td>MCU_CNTL</td>
<td>DF0B</td>
<td>Address register for upper FLASH/SRAM bits</td>
</tr>
<tr>
<td>FPGA_SELECT</td>
<td>DF0C</td>
<td>FPGA_select[5:0] = bits 5:0</td>
</tr>
<tr>
<td>PPC_RS232_ABSELECT</td>
<td>DF0D</td>
<td>No effect on DN8000K10PSX</td>
</tr>
<tr>
<td>PPC_RS232_CDSELECT</td>
<td>DF0E</td>
<td>No effect on DN8000K10PSX</td>
</tr>
<tr>
<td>FPGA_CNTRL</td>
<td>DF0F</td>
<td>bits[1:0] = 01 (write address), 10 (data write), 11</td>
</tr>
<tr>
<td>FPGA_BE</td>
<td>DF10</td>
<td>select byte in addr, read, and data bytes</td>
</tr>
<tr>
<td>FPGA_RD_DATA</td>
<td>DF11</td>
<td></td>
</tr>
<tr>
<td>FPGA_WR_DATA</td>
<td>DF12</td>
<td></td>
</tr>
<tr>
<td>FPGA_ADDR</td>
<td>DF13</td>
<td></td>
</tr>
<tr>
<td>FPGA_ERROR</td>
<td>DF14</td>
<td></td>
</tr>
<tr>
<td>GPIF_DATA</td>
<td>DF20</td>
<td></td>
</tr>
<tr>
<td>GPIF_ERROR</td>
<td>DF21</td>
<td></td>
</tr>
<tr>
<td>HOLD_DONES</td>
<td>DF22</td>
<td></td>
</tr>
<tr>
<td>FPGA_FREQ_H</td>
<td>DF24</td>
<td></td>
</tr>
<tr>
<td>FPGA_FREQ_SEL</td>
<td>DF25</td>
<td></td>
</tr>
<tr>
<td>FPGA_FREQ_L</td>
<td>DF26</td>
<td></td>
</tr>
<tr>
<td>MCU_STUFFING1</td>
<td>DF27</td>
<td></td>
</tr>
<tr>
<td>MCU_STUFFING2</td>
<td>DF28</td>
<td></td>
</tr>
<tr>
<td>SERIAL_CLK_CTRL_0</td>
<td>DF29</td>
<td></td>
</tr>
<tr>
<td>SERIAL_CLK_CTRL_1</td>
<td>DF30</td>
<td></td>
</tr>
<tr>
<td>MB80_1_CTRL0</td>
<td>DF36</td>
<td></td>
</tr>
<tr>
<td>MB80_1_CTRL1</td>
<td>DF37</td>
<td></td>
</tr>
<tr>
<td>MB80_2_CTRL0</td>
<td>DF38</td>
<td></td>
</tr>
<tr>
<td>MB80_2_CTRL1</td>
<td>DF39</td>
<td></td>
</tr>
<tr>
<td>FPGA_COMMUNICATION</td>
<td>DF40</td>
<td></td>
</tr>
<tr>
<td>MB80_3_CTRL</td>
<td>DF41</td>
<td></td>
</tr>
<tr>
<td>MB64_1_CTRL</td>
<td>DF42</td>
<td></td>
</tr>
<tr>
<td>MB64_2_CTRL</td>
<td>DF43</td>
<td></td>
</tr>
<tr>
<td>MB64_3_CTRL</td>
<td>DF44</td>
<td></td>
</tr>
<tr>
<td>CPLD_CS_N_CTRL</td>
<td>DF45</td>
<td></td>
</tr>
<tr>
<td>CPLD_DATA</td>
<td>DF46</td>
<td></td>
</tr>
<tr>
<td>CPLD_ADDR</td>
<td>DF47</td>
<td></td>
</tr>
</tbody>
</table>
These registers can be written to from the USB interface. See *USB Software: Programmers Guide*.

### 2.5.5 USB

The Cypress CY7C68013 has a built-in USB 2.0 interface. The USB type B connector on the DN8000K10PSX (J12) is connected directly to the USB pins on the Cypress MCU.

![USB Transient Protection](image)

The USB protocol is completed by the Cypress CPU.

The Cypress receives a 24Mhz clock from an oscillator (X3). The Cypress internally multiplies this clock to 480Mhz for USB 2.0 and 48Mhz for GPIF operation. The core runs at 24Mhz along with the external memory interface. Communication over this external memory interface is clocked using the MCU_IFCLK signal driven from the MCU at 48Mhz. (The Spartan communicates over main bus with the Virtex 4 FPGAs using a separate 48Mhz oscillator (X1) and distributes this clock to each FPGA including itself)
2.5.6 CompactFlash

The CompactFlash card socket pins are bussed among the Cypress MCU GPIF pins, the Spartan 2 FPGA IOs, and the CompactFlash card socket. After reset, the MCU uses this connection to look for and read the contents of the file main.txt on the CompactFlash card. The main.txt file contains instructions for configuring the user design into the three Virtex 4 FPGAs.

After reading the configuration instructions, the MCU reads the headers of the user’s FPGA design (“.bit”) files and verifies that they target the correct type of FPGA that are installed on your DN8000K10PSX. This will prevent damage to the FPGA from an incorrect or corrupt .bit file. This behavior can be turned off.

If this check is passed, MCU uses its memory mapped interface with the SpartanII to instruct the SpartanII to read the CompactFlash card and configure the Virtex 4 FPGAs over SelectMap bus.

3 Clocking

The clocking circuitry on the DN8000K10PSX is designed for high-speed operation. The flexible clock design should meet the most difficult clocking needs, allowing 4 totally asynchronous, clock sources for each FPGA.

All clocks operating above 100Mhz are fully differential, LVDS signaled, low skew, low jitter clocks.
Figure 30 DN8000K10PSX Clocking

From the above diagram, the global clocks are listed here.

GCLK0, GCLK1, GCLK2. These global clocks are supplied by ICS8442 frequency synthesizers. They are configured from the MCU to output a user-specified frequency from 31 to 700Mhz. They are each distributed to FPGAs A B and C.

GCLK0 (SMA), GCLK0 can be configured to drive a clock from the SMA inputs, J16, J17, onto the global clock net.

GCLK1 can be configured to drive a clock onto the global network from the MegArray 400pin daughtercard headers (Signal DCCCLK)

GCLK2 can be configured to drive a clock on the global network from FPGA A (FBCLK signal)

Sysclk – this 48Mhz, single-ended clock is driven from the configuration FPGA at a fixed frequency. It is delivered to FPGAs A, B, C and the configuration FPGA. This clock is used by the Dini Group reference design to clock the Main Bus interface. If you implement USB communication using the MainBus interface, you must use this clock.
PCLK – This single-ended, 75Mhz fixed clock is delivered to the Configuration FPGA, the Quicklogic PCI bridge, and FPGA A. It is used to clock the QL PCI back-end interface. It can also be used for the user design in FPGA A.

DCACLK, DCBCLK (not shown in diagram) A signal driven from the dughtercard to FPGA C only.

DDR_REF – each FPGA drives this clock to an external buffer. To compensate for the buffer delay, this buffer drives back DDR_FB to the FPGA. The Dini Group Reference design uses this clock to run the DDR2 reference design.

UCLK (not shown in diagram) – Two SMAs connect to global clock input pins on FPGA A. (J9, J10). These can be used as a clock input, or just as test IOs.

### 3.1 Global Clock Synthesis

The three main global clocks are driven by ICS8442 clock synthesizers, each capable of producing frequencies of 700Mhz (There is a 500Mhz input limit on Virtex-4 FPGAs) The clock synthesizers can be programmed from a CompactFlash card, from the GUI application, or PCI application. We recommend leaving a CompactFlash card programmed with clock settings installed in the DN8000K10PSX even if you are not configuring FPGAs in this way.

Each ICS8442 has an interal multiplication PLL that can operate between 250 and 700 Mhz. With 1, 2, 4, or 8x division on the output, the possible output frequencies are 31.25 – 700Mhz. (Virtex 4 FPGAs are limited by a 500Mhz clock input max). VCO_SEL can be used to disable the PLL, so G0, G1, G2 can operate at frequencies below this 31Mhz minimum (NOT IMPLEMENTED YET CONTACT SUPPORT)
To program the frequency of these synthesizers, you can use the main.txt configuration file on CompactFlash. Add a line in Main.txt:

**CLOCK FREQUENCY: G0 M 25 N 8**

This line causes the 8442 synthesizer associated with GCLK0 to multiply its 25Mhz reference by 25, then divide by 8. Note that the reference frequency times the multiplication factor is 625Mhz, which is in the PLL range of 250-700Mhz.

Each global clock is delivered to the FPGA as an LVDS, differential clock. The IO input on this clock should be configured as a differential clock input (the IBUFGDS primitive).

The example below shows the Verilog instantiation of this module, using the GCLK0 signal.

```verilog
wire aclk_ibufds;
IBUFGDS ACLK_IBUFG (.O(aclk_ibufg), .I(GCLK0p), .IB(GCLK0n)) ;
```

The signal `aclk_ibufds` should then be fed to either a BUFG or a DCM before being used as an internal clock for FPGA logic.

### 3.2 User Clock

The DN8000K10PSX has two SMA pairs specifically designed for accepting a clock input. The pair on the upper left (J9, J10) connects directly to clock input pins on FPGA A. The signals are named CLK_SMAP, CLK_SMAPn

![Image of DN8000K10PSX board](image)

Use a differential input method, like an extended LVDS differential input clock buffer.

### 4 Reset Topology

The DN8000K10PSX is protected from undervoltage and over temperature by a reset circuit. When the board powers on, a voltage monitor waits until all voltages are above their minimum voltage levels, then deasserts reset. The Spartan 2 distributes the reset signal to all FPGAs and the Microcontroller unit, so until the Spartan 2 is configured, reset remains asserted.
The user may also assert reset by pressing S5, “Hard reset” This will trigger the reset signal “SYS_RSTn” which is monitored by the Spartan FPGA. When SYS_RST is asserted, the Spartan FPGA resets the Virtex 4 FPGAs, causing them to lose their configuration data and deactivate. The Spartan also causes a reset on the Microcontroller unit, which will cause the microcontroller to reload configuration instructions from the CompactFlash card. USB contact will be lost with the USB host, and the DN8000K10PSX will have to re-enumerate.

There is a RED LED next to the HARD RESET button, DS14, that will light when the board is in reset. If the DN8000K10PSX is not responding to USB or RS232, then look at this LED and see if it is on. If it is, then there could be a problem with the power being supplied to the board.

There is a second button, S6 called “Soft Reset”. When this button is pressed, the signal “RESET_FPGAs” is asserted. This signal is sent to the Virtex 4 FPGAs on a user IO pin, and could be used by the user design as a reset signal. This signal is also asserted to all FPGAs after any FPGA becomes configured. RESET_FPGAs is an asynchronous signal.

Each FPGA is also connected to a temperature monitor. The Virtex 4 FPGA can easily overheat if a heatsink and fan are not used. The recommended operating temperature for the Virtex 4 is 85 degrees C. The absolute maximum temperature for operation is 125 degrees C. If
at any time the junction temperature of the Virtex 4 exceeds 85 degrees, the Microcontroller will reset the FPGAs, causing them to lose their configuration data. An overheating FPGA could be the result of a misconfiguration, a clock that is set incorrectly, or an inadequate heatsink unit. The heatsink and fan assembly that comes with the DN8000K10PSX is appropriate for dissipating the amount of heat energy available through a PCI slot without the auxiliary power connector (25W total for the card). If you are operating the DN8000K10PSX at very high speeds in stand-alone mode and you are causing heat overload resets, you may need to install a larger heatsink, or increase the system airflow.

This circuit shows the MAX1617 temperature monitor. The IIC bus is connected to the Cypress microcontroller.

5 Power

The DN8000K10PSX gets its power from the 5.0V and 3.3V rails of the PCI card edge connector. It can also be operated in stand-alone mode with a 4-pin “Molex” “Hard drive power” connector. You can operate the board with the external power connector connected and the board plugged into PCI, but be careful that the hard drive connector is generated from the same power supply as the PCI host, or else there will be a DC connection between two opposing power supplies as the DN8000K10PSX becomes a high-current path.

The PCI slot is capable of sourcing 25W. The DN8000K10PSX is capable of drawing much more power than this. In this case, the 4-pin “hard drive” power connector may be required.
The main rails of the DN8000K10PSX are:

- 1.2V – This is the main power supply rail used for the internal digital logic of Virtex 4 FPGAs.

- 1.8V – This is used for IO signaling and internal logic of DDR2 SDRAM memory. It is also used to supply some Gigabit optical modules, and is used as a low-power current source to supply RocketIO isolated power rails.

- 2.5V – This is used to power FPGA interconnect with low-power LVDS. It is also used as the analog power supply on the Virtex 4 FPGAs.

- 3.3V – This voltage supplies the LVDS clock distribution trees. It is also used to power the LVTTL interfaces of the Cypress microcontroller, and Quicklogic 5064 PCI bridge.

- 5.0V – This voltage is used to supply power to the 1.2, 2.5 and 1.8V switching power supplies. It also powers the FPGA cooling fans, some Gigabit optical modules, and the PCI signaling. If the PCI slot isn’t providing enough power, then a Hard Drive 4-pin power cable can be connected to the board (from the same ATX power supply) to reduce the voltage droop on 5V. Please note that the board is capable of exceeding the 25W limit of the PCI connector (depending on the desity of the FPGAs utilized, and the operating frequency).

The DN8000K10PSX also has these secondary rails:

- 0.9V – This voltage is used to terminate the SSTL18 signaling of the DDR2 memory module.

- +12V – This rail is passed directly from the PCI edge connector and ATX power connector to the MegArray expansion header.

- -12V – This rail is passed directly from the PCI edge connector and ATX power connector to the MegArray expansion header.

- +VIO_DCC0,1,2 These 1.2V power rails supply the IO current of the FPGA to the daughtercard headers. There is a separate power rail for each of the three Virtex-4 banks connected to the header. These regulators are only capable of sourcing current, so a daughtercard may choose to increase the voltage of these nets with no negative effects. With a resistor change, these regulators can be modified to output higher currents if required.

There are test points for measuring the voltage levels of each rail near the top left of the DN8000K10PSX. Each rail is monitored by a voltage monitor circuit, and will cause a reset if any of the primary supplies drop 5% or more below their setpoints.
There are also LEDs next to each testpoint to indicate the presence of each voltage rail. These LEDs do not indicate that a rail is within 5% of its setpoint, only that the rail is present.

5.1 Switching power supplies
The main power rails for the Virtex 4 FPGAs are produced on board with three 20A switching power supplies, one for each of 1.8V, 2.5V, and 1.2V.

Each power supply is protected with a 15A fuse on the inputs. These should never trip, unless there is a hardware failure.

Each of the primary power rails (5.0, 3.3, 2.5, 1.8, 1.2) is monitored for undervoltage. If the voltage monitor circuit detects a low voltage, it will hold the board in reset until the supply is back within 5% of its setpoint.

5.2 Secondary Power Supplies
The secondary power supplies are derived from a primary supply.

5.2.1 DDR2 Termination Power
DDR2 memory modules use the SSTL18 signaling standard. Properly terminating SSTL18 requires a termination power supply of 0.9V. Since as much as 1.6 Amps of termination current are needed, a switching power supply is required.
The ML6554 produces up to 3A of the required 0.9V termination power rail along with a stable 0.9V reference voltage supply.

5.3 Heat dissipation

Virtex 4 FPGAs are capable of drawing incredible amounts of current from their 1.2V and 2.5V power supplies. According to Xilinx online power estimator tool, a fully utilized FPGA running at 300Mhz can draw more than 30W of power. With this much power used in each FPGA, the DN8000K10PSX can dissipate 75 or more Watts of heat. For all but the most trivial designs, a heatsink must be used with the Virtex 4 FPGA. The DN8000K10PSX comes with a forced air heatsink rated at 4 degrees per Watt. Since the maximum operating junction temperature of a Virtex 4 FPGA is 85 degrees, assuming an ambient temperature of 50 degrees (the inside of your computer case) the most amount of energy dissipated by the FPGA using the standard fan is $85 - 30 / 4 = 27.5W$. This should be sufficient for many applications. If you intend to operate the Virtex 4 FPGA at very high speeds, or are getting overheating issues with your design, you will need to install a larger heatsink.

Above: The FPGA temperature monitor circuit. The MAX1617’s IIC bus is connected to the Cypress MCU.
6 FPGA interconnect

The DN8000K10PSX was designed to maximize the amount of interconnect between the two primary Virtex 4 FPGAs A and B. This interconnect was routed as tightly coupled differential LVDS to provide the best immunity to power supply and crosstalk noise so that your interconnect can operate at the full switching speed of the output buffers. Following Xilinx recommendations, the interconnect on the DN8000K10PSX was designed to operate at 1Gb/s for every LVDS pair. (Note 1Gb/s operation requires the fastest speed-grade part, LX200 –12) In order to achieve such breakneck speeds, you will need to operate the busses of signals using a source-synchronous clocking scheme. The interconnect signals on the DN8000K10PSX have been optimized to operate in “lanes” There are 7 lanes between FPGAs A and B, three between B and C and two between FPGAs A and C. Each lane has a differential LVDS source-synchronous clock in each direction. For a complete pinout of the Virtex 4 FPGA interconnect, you can use the customer netlist provided on the User CD.

Clocking incoming data at high speeds required the used of the each input’s delay buffer to align each bit. The incoming clock needs to be adjusted and used to clock the inputs within its lane. This process can be automated by the use of the new Virtex 4 feature IDelayCTL.

Above: Cooling fan power connector.
For detailed description of the required user design to achieve 1Gbs operation, see Xilinx Application note XAPP704, “High Speed SDR LVDS Transceiver”.

Synchronous clocking and single-ended signaling are still possible on the DN8000K10PSX, you are not required to use highspeed serial design techniques. Single ended interconnect is recommended for signaling below 133Mhz. Because of the DN8000K10PSX’s excellent low-skew clocking network, global synchronous clocking should work fine for your interconnect at speeds lower than 300Mhz. The source synchronous clock signals can also be used as single ended or differential interconnect, or to forward clocks from one FPGA to another.

The maximum total interconnect counts between FPGAs

- A to B 248
- B to C 248
- A to C 31

Some of these interconnects are not available when the DN8000K10PSX comes installed with smaller FPGAs (SX55, LX60, LX40)

- If FPGA A is a small FPGA, then AB is limited to 124 signals max.
- If FPGA B is a small FPGA, then AB is limited to 186 signals max, BC is limited to 186 signals max.
- If FPGA C is a small FPGA, then BC is limited to 186 signals max.
7 Memory interface

There are two standard 200-pin DDR2 SODIMM module sockets on the DN8000K10PSX. These sockets are supplied with 1.8V power and keyed for use with DDR2 SDRAMs. One socket is connected to FPGA B and the other is connected to FPGA C.

7.1 Clocking

SODIMM interfaces:

7.2 Serial presence detect.

The EEPROM on the SODIMM is accessible by PCI, USB, or configuration UART.
8 Headers

The daughter card interface includes a 400-pin MEG-Array connector, made by FCI. The daughter card header is arranged in “Banks”, correlating to the banks of IO on the Virtex 4 FPGA. Each 400-pin connector contains 3 full Virtex 4 IO banks of 62 signals each. (Virtex 4 banks are 64 IOs each, but two of these are reserved for DCI termination)

Other connections on the daughter card connector system include three dedicated, differential clock connections for inputting global clocks from an external source, power connections, bank VCCO power, a buffered power on reset signal.
8.1 Daughter card Physical

The connectors used in the expansion system are FCI MEG-Array 400-pin plug, 6mm, part #84520-102. This connector is capable of as much as 10Gbs transmission rates using differential signaling.

All daughter card expansion headers on the DN8000K10 are located on the bottom side of the PWB. This is done to eliminate the need for resolving board-to-board clearance issues, assuming the daughter card uses no large components on the backside.

The “Plug” of the system is located on the DN8000K10, and the “receptacle” is located on the expansion board. This selection was made to give a greater height selection to the daughter card designer.

8.1.1 Daughter Card Locations and Mounting

The 400-pin daughter card header is located on the bottom (solder) side near the right side of the board. Each MegArray header on a Dini Group product has four standard-position mountain holes. The drawing below shows the location of the daughter card header and it’s associated Mounting holes.
This view of the DN8000K10PSX daughter card locations is from the top of the PCB, looking through to the bottom side. The Dini Group standard daughtercard, DNMEG_OBS400 is compatible with the DN8000K10PSX

With this host-plate-daughtercard arrangement, there is a limited Z dimension clearance for backside components on the daughter card. This dimension is determined by the daughter card designer’s part selection for the MegArray receptacle.

Note that the components on the topside of the daughter card and DN8000K10PSX face in opposite directions.

8.1.2 Insertion and removal
Due to the small dimensions of the very high speed MegArray connector system, the pins on the plug and receptacle of the Meg Array connectors are very delicate.

When plugging in a daughter card, make sure to align the daughter card first before pressing on the connector. Be absolutely certain that both the small and the large keys at the narrow ends of the Meg Array line up BEFORE applying pressure to mate the connectors!
Place it down flat, then press down gently.

The following two excerpts are taken from the FCI application guide for the Meg Array series of connectors.

A part can be started from either end. Locate and match the connector’s A1 position marking (“?”) for both the Plug and Receptacle. (Markings are located on the long side of the housing.) Rough alignment is required prior to connector mating as misalignment of >0.8mm could damage connector contacts. Rough alignment of the connector is achieved through matching the Small alignment slot of the plug housing with the Small alignment key of the receptacle housing and the Large alignment slot with the Large alignment key. Both connector housings have generous lead-in around the perimeter and will allow the user to blind mate assemble the connectors. Align the two connectors by feel and when the receptacle keys start into the plug slots, push down on one end and then move force forward until the receptacle cover flange bottoms on the front face of the plug.

Dec 09, 2004
Like mating, a connector pair can be unmated by pulling them straight apart. However, it requires less effort to un-mate if the force is originated from one of the slot/key ends of the assembly. (Reverse procedure from mating) Mating or un-mating of the connector by rolling in a direction perpendicular to alignment slots/key may cause damage to the terminal contacts and is not recommended.

8.2 Daughter Card Electrical

The daughter card pin out and routing were designed to allow use of the Virtex 4's 1 Gbps general purpose IO. All signals on the DN8000K10PSX are all routed as differential, 50-Ohm transmission lines.

No length-matching is done on the PCB for daughter card signals, (except between two ends of a differential pair), because the Virtex 4 is capable of variable-delay input using the built-in IDelay module.

8.2.1 Pin assignments

The pin out of the DN8000K10 expansion system was designed to reduce cross talk to manageable levels while operating at full speed of the Virtex 4. The ground to signal ratio of the connector is 1:1. General purpose IO is arranged in a GSGS pattern to allow high speed single-ended or differential use. On the host, these signals are routed as loosely-coupled differential signals, meaning when used differentially, they benefit from the noise-resistant properties of a differential pair, but when used single-endedly, do not interfere with each other excessively.

All high-speed signals on the DN8000K10, including daughter card signals, are routed against a ground potential reference plane.

The RocketIO signals on daughter cards DC0 and DC3 are arranged in a GSSG. These signals can only be used in a differential configuration, and cross talk between the two signals is complementary and beneficial. On the host, these signals are routed as 110-Ohm differential signals. 110 Ohm signaling was chosen because the Meg Array connector system in the 14mm stack height configuration is slightly inductive. For the 35ps rise time of a Virtex 4 RocketIO CML signal, the Meg Array connector appear very much like a 110Ohm transmission line with a 70ps transmission delay. Daughter cards designed to work with RocketIO at the highest data rates should account for this during design.

You may want to read the following references for designing a daughter card using 110Ohm RocketIO signals:

Howard Johnson, High-Speed Signal Propagation, p. 315 Matching Pads
Xilinx Virtex 4 MGT Users Guide See: TXTERMITRIM

The central columns of the connector pin out use a closely coupled, differential pair pin arrangement, which is uniformly surrounded by ground pins. These differential pins are used for RocketIO connections on FX parts. All other signals use a “checkerboard” type of ground...
arrangement. This allows the signals to be used as high-speed, single-ended, or as loosely coupled differential pairs.

There are two types of connectors on the DN8000K10, 300 and 400 pins. The first 300 pins on both types of connectors are identical. This should allow a 300-pin connector to be installed on a 400-pin land pattern on a daughter card to allow limited functionality in 300-pin daughter card positions. The “Banks” of signals are segregated. On the 300-pin connector, there are extra signals in the checkerboard pattern that are left as NC.
Below is a graphic representation of the pin assignments for the 300- and 400-pin connectors. Note that this is a view from the backside of the connector. The green boxes represent ground connections.
Special purpose pins are described below.

### 8.2.2 CC, VREF, DCI

Some of the signals connected to the daughter card expansion headers are “clock-capable”; the inputs on the Virtex 4 FPGA can be used for source-synchronous clocking. In the schematic and customer netlist on the user CD, these pin contain a “\_CC” in the pin name. In the ucf file provided with the reference design, and in the schematic, these signals in the FPGA interconnect have “CLK” in the name.

Pins declared as “VREF” pins by Xilinx have a defined placement on the daughter card pin out to allow the daughter card to define a logic threshold as required by some standards.

DCI is used on all FPGA IO banks connected to a daughter card header. The reference resistance is 50 Ohms. A Virtex 4 bank has 64 pins. Of each bank connected to a daughter card header, 62 signals are connected to the header, and 2 are used as DCI reference pins.

### 8.2.3 Global clocks

The daughter card pin out defines 6 clock input pins. These clock inputs are intended to be used a 3 differential signals. Two clock signals GCA and GCB connect to the “GC” clock inputs in the FPGA. These clocks can be used as global clocks from within the FPGA code of the FPGA that connects to the daughter card, but not globally to the entire DN8000K10.

The GCC signal on every daughter card except DC4 and DC9 connects to the “Daughter card Global Clock” network. This clock input can be distributed to all 16 FPGAs on the DN8000K10. For more information on the daughtercard clock network, see Hardware: Clocks: Daughter card Clocks.

For distributing an FPGA-global clock to the entire board, the Dini Group standard daughter card DNMEGOBS-300 or DNMEGOBS-400 is capable of driving the global clock network from its GCC pin.

### 8.2.4 Power and Reset

The +3.3V, +5.0V and +12V power rails are supplied to the Daughter card headers. Each pin on the MegArray connector is rated to tolerate 1A of current without thermal overload. Most of the power available to daughter cards through the connector comes from the two 12V pins, for a total of 24W. Each power rail supplied to the Daughter card is fused with a reset-able switch. Daughter cards are required to provide their own power supply bypassing and onrush current limiting.
The RSTn signal to the daughter card is an open-drain, buffered copy of the SYS_RSTn signal. This signal causes the entire DN8000K10 to reset, losing all FPGA configuration data and resetting the configuration circuitry.

### 8.2.5 MGT Signals

Also see Hardware: MGT Serial Resources: The connections: Daughter cards.

### 8.2.6 VCCO Voltage

The signal voltage on the Daughter card interface is defined by the daughter card by setting the voltage on the VCCO0, VCCO1 and VCCO2 pins. Since the daughter card provides all the current necessary for the FPGA on the DN8000K10 to communicate over the daughter card interface, the daughter card designer will have to determine the current requirements of the interface. Typically, this current requirement of the host board can be an Amp.

The each VCCO_ net supplies power for the host board FPGA for one entire bank. Bank 0 (VCCO0) includes the signals B0L[0-31]. Bank 1 (VCCO1) includes B1L[0-31]. Bank 2 (VCCO2) includes signals B2L[0-31] (on the 400 pin headers only)

FPGA VCCO power is provided by the daughter card for each connected bank. This allows the daughter card to define the I/O standard to be used on the bank.

### 8.2.7 VCCO bias generation

Since a daughter card will not always be present on a daughter card connector, a VCCO bias generator is used on the motherboard for each daughter card bank to keep the VCCO pin on the FPGA within its recommended operating range. The VCCO bias generators supply +1.2V to the VCCO pins on the FPGAs, and are back-biased by the daughter card when it drives the VCCO rails.

Overdriving this power net from the daughtercard is perfectly safe. The bias regulator will simply shut off and not interfere with the VCCO power net.
The VCCO voltage impressed by the daughter card should be less than 3.75 to prevent destruction of the Virtex 4 IOs connected to that daughter card.

8.2.8 Types 2 Short (400pin Short)
The daughtercard mechanical provisions on the DN8000K10PSX are for a “type 2 short” daughtercard. The DNMEGOBS-400 is Type 2. This standard daughtercard provides headers and Mictor connectors for signal access to the DN8000k10PSX.
The mounting hole positions are standard, and the DN8000K10 has holes in its base plate to accommodate these holes.

9 LEDs

Figure 31 FPGA C LEDs

FPGA A is connected to 8 green LEDs. FPGA C is connected to 16 LEDs. These LEDs can be used for the user design. The brightness of these LEDs can be controlled by changing the output standard on the LED signals from 2, 4, 12, 16 or 24mA.

10 PCI interface

10.1 PCI edge connector

3.3v or 5.0V universal 64-bit card edge connector.
Do not attempt to plug a PCI card in backwards if it does not fit.

10.2 The Quicklogic 5064
In order to provide a highspeed easy-to-use interface for your design, the DN8000K10PSX comes equipped with a PCI bridge, a QuickLogic 5064.

To use PCI with your design, you should use the provided PCI interface module. A description of this module is in the file, QL5064_Interface_Module.doc on the user CD. The rest of this section describes to function of the PCI hardware.

10.3 Virtex 4 FPGA Communication

10.4 Spartan 2 Communication
The FIFO interface on the DN8000K10PSX side of the Quicklogic 5064 is shared between FPGA A, a Virtex 4 FPGA, and The configuration FPGA, a SpartanII. This allows the
configuration circuitry to configure the Virtex 4 FPGAs over the PCI bus.

10.5 PCI clocking
All communication to the Quicklogic 5064 chip is synchronized with a 75Mhz oscillator. The 75Mhz PCI UCLK is delivered to FPGA A, the Spartan 2 FPGA and the Quicklogic 5064.

10.6 JTAG
The PCI connector's JTAG signals are looped back to bypass the DN8000K10PSX when it is plugged into the PCI slot.
10.7 PCI Power

In most applications, the DN8000K10PSX can draw its power from the PCI slot. The PCI specification requires that the motherboard provide 25W of 5V power for the DN8000K10PSX to use (Most motherboards provide well in excess of this amount, supplying the power for PCI cards directly from the ATX power supply). In high power applications exceeding 25W, you may need to connect the Auxiliary power connector (P3).

The Aux. Power connector is a standard IDE hard drive power connector and should be supplied by the ATX power supply that is in your computer case. Aux power connector 5.0V and 3.3V are shorted to the PCI slot 5.0V and 3.3V. The power supply driving the PCI slot and IDE power cable must be driven from the same unit.

If you are operating the DN8000K10PSX in a server or other enclosure that does not have available IDE hard drive cables, and you intend to use the DN8000K10PSX in a high-power application, then an alternate setup will be required. There are eight 0Ω resistors shorting the PCI edge connector’s 5V pins and the power distribution plane on the DN8000K10PSX. This connection disallows safely connecting the Aux. IDE power connector to an external power supply. To allow the use of a separate power supply dedicated to the DN8000K10PSX while it is in a PCI slot, these resistors can be removed. (Please e-mail tech support if you think that you need these resistors removed). You will also need remove the 12V wire on the IDE hard drive cable to prevent contention with the PCI slot’s 12V supply. Note that in this configuration, the 5.0V power for the DN8000K10PSX (and most of the required power) will be supplied from the external supply over P3. The 3.3V power for the DN8000K10PSX will continue to be supplied from the eight 0Ω resistors are RN3, RN4, RN6, RN5, RN47, RN48, RN50, RN51.
Remember, Most users will use the same power supply to power the motherboard and Aux. IDE power connector (P3). In this configuration, these modifications are not required.

10.8 PCI Signaling

To allow universal (3.3V or 5.0V) PCI IO, the DN8000K10PSX uses the PCI bus's VIO pins to detect the IO levels used by your motherboard. Most motherboards use 5V signal levels on the PCI bus, but many servers, and PCI slots require 3.3V signaling. IO voltage for the DN8000K10PSX is provided by a jumper connecting the PCI VCCIO signal to the Quicklogic 5064. Be sure that a jumper is installed between pins 2 and 4 of the jumper block JP1. In the rare case that you have a PCI core loaded in A and want to operate the DN8000K10PSX in standalone mode, you may need to move power up and detect no PCI bus present. You should never install a jumper in both positions. This could short 5.0V to 3.3V when plugged into a 64-bit, 3.3V slot.

The DN8000K10PSX can be used in a PCI or PCI-X slot operating at 33Mhz or 66Mhz. It can also be used in 100Mhz and 133Mhz busses, although the DN8000K10PSX will cause the entire bus to operate at 66Mhz.
11 FPGA System monitor/ADC

The System Monitor and ADC functions of the Virtex 4 FPGA are no longer supported by Xilinx. The most important responsibility of the System Monitor, temperature sensing, has been moved to the configuration circuitry. The DN8000K10PSX will automatically monitor and prevent thermal overload in the three Virtex 4 FPGAs. No user action is required.

12 Mechanical

The topside clearance with the factory installed active heatsinks is 14mm.

The board should plug into any PCI or PCI-X slot with 5V or 3.3V keying, 32-bit or 64-bit slot widths. (33Mhz or 66Mhz (100 and 133Mhz will be brought down to 66Mhz automaticall
Chapter 5: The Reference Design

This chapter introduces the DN8000K10PSX Reference Design, including information on what the reference design does, how to build it from the source files, and how to modify it for another application.

1 Exploring the Reference Design

1.1 What is the Reference Design?
The reference design is a fully functional Virtex 4 FPGA design capable of demonstrating most of the features available on the DN8000K10PSX. Features exercised in the reference design include:

- Access to the DDR2 SDRAM Modules At 200Mhz
- UART Communication
- FPGA Interconnect
- Interaction with the Configuration FPGA and MCU
- Use of Embedded PowerPC Processors (eventually)
- Memory Mapped Access Between PPC And User Design (eventually)
- Access to external LEDs
- Communication via Rocket I/O Transceivers
- Instantiation of Daughter Card Test Headers
- USB memory map to DDR2 memory.
- PCI memory map to DDR2 memory.
Pin-multiplexed FPGA interconnect using LVDS at 700Mbs per signal pair

All source code for the reference design is included on the CD and may be used freely in customer development. Precompiled bit files for the most common stuffing options are also included and can be used to verify board functionality before beginning development. A build utility, described in the section Compiling The Reference Design, can be used to generate new bit files, or to generate bit files for less common configurations of the DN8000K10PSX.

The reference design was created using

Here are the default main.txt file lines.

```
verbose level: 2
sanity check: y

# clock frequency: A N 4 M 16 // 100 MHz – not used for PCI/MB test,
# header test uses this clk
clock frequency: B N 2 M 28 // 200 MHz

# clock frequency: D N 2 M 25 // 200 MHz

# clock frequency: 1 N 2 M 25 // 312 MHz

# clock frequency: 2 N 2 M 25 // 312 MHz
```

## 2 Reference Design Memory Map

The Dini Group reference design memory maps the main features of the DN8000K10PSX to the host interfaces: PCI, USB, and RS232.

The Main Bus interface is used to access the reference design memory map. Addresses are 32-bits. Each address contains a 32-bit word.

<table>
<thead>
<tr>
<th>FPGA A</th>
<th>Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0x08000002</td>
<td>IDCODE</td>
</tr>
<tr>
<td></td>
<td>0x08000004</td>
<td>INTERCONTYPE</td>
</tr>
<tr>
<td></td>
<td>0x08000006</td>
<td>RWREG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scratch Register for testing</td>
</tr>
<tr>
<td></td>
<td>0x08000010</td>
<td>LED_OE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Controls LED output enables</td>
</tr>
<tr>
<td></td>
<td>0x08000011</td>
<td>LED_OUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Controls LED outputs</td>
</tr>
<tr>
<td></td>
<td>0x08000021</td>
<td>CLK_COUNTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Contains contents of ACLK counter</td>
</tr>
<tr>
<td></td>
<td>0x08000022</td>
<td>CLK_COUNTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Contains contents of BCLK counter</td>
</tr>
<tr>
<td></td>
<td>0x08000023</td>
<td>CLK_COUNTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Contains contents of DCLK counter</td>
</tr>
<tr>
<td></td>
<td>0x08000024</td>
<td>CLK_COUNTER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Contains contents of SYSCLK counter</td>
</tr>
<tr>
<td></td>
<td>0x0C000000</td>
<td>ABP0 OUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>W; the output state of FPGA IOs</td>
</tr>
</tbody>
</table>
connected to the ABP0 interconnect bus

FPGA A 0x0C000004 ABP0 OE W; The output enable of each FPGA IO on the ABP0 interconnect bus.
FPGA A 0x0C000008 ABP0 IN The input state of each FPGA IO… …on the ABP0 interconnect bus
FPGA A 0x0C00000C ABP0 Name “ABP0” (ascii)
FPGA A 0x0C000010 ABP1 OUT W; ABP1 IO output values
FPGA A 0x0C000014 ABP1 OE W; Output enable of ABP1 bus
FPGA A 0x0C000018 ABP1 IN R; ABP1 input values
FPGA A 0x0C00001C ABP1 Name “ABP1” (ascii)

FPGA B 0x10000000 DDR2 B space… Mapped to DDR2 SODIMM…
FPGA B 0x17FFFFFF … …interface
FPGA B 0x18000002 IDCODE 0x05000121
FPGA B 0x18000004 INTERCONTYPE 0x34561111
FPGA B 0x18000006 RWREG Scratch Register for testing
FPGA B 0x18000010 LED_OE Controls LED output enables
FPGA B 0x18000011 LED_OUT Controls LED outputs
FPGA B 0x18000021 CLK_COUNTER Contains contents of ACLK counter
FPGA B 0x18000022 CLK_COUNTER Contains contents of BCLK counter
FPGA B 0x18000023 CLK_COUNTER Contains contents of DCLK counter
FPGA B 0x18000024 CLK_COUNTER Contains contents of SYSCLK counter
FPGA B 0x18000001 DDR2HIADDR upper address bits for DDR2 interface
FPGA B 0x18000003 HIADDRESIZE number of bits in DDR2HIADDR
FPGA B 0x18000005 DDR2SIZEHIADD The size of the DDR2 module.
FPGA B 0x18000007 DDR2TAPCNT0 Current IDELAY values of DDR2… …interface
FPGA B 0x18000008 DDR2TAPCNT1

FPGA B 0x1C000000 BUS XX OUT XX can be 0-21 hex. Output status of IOs on bus XX.
FPGA B 0x1C000004 BUS XX OE XX can be 0-21 hex. OE status of IOs
FPGA B 0x1C000008 BUS XX IN XX can be 0-21 hex. The input values
FPGA B 0x1C00000C BUS XX Name The name of the bus XX (schematic)
FPGA C 0x20000000 DDR2 C space… Mapped to DDR2 SODIMM…
FPGA C 0x27FFFFFF … … interface
### FPGA C

<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x28000000</td>
<td><strong>IDCODE</strong> 0x05000121</td>
</tr>
<tr>
<td>0x28000004</td>
<td><strong>INTERCONTYPE</strong> 0x34561111</td>
</tr>
<tr>
<td>0x28000006</td>
<td><strong>RWREG</strong> Scratch Register for testing</td>
</tr>
<tr>
<td>0x28000010</td>
<td><strong>LED_OE</strong> Controls LED output enables</td>
</tr>
<tr>
<td>0x28000011</td>
<td><strong>LED_OUT</strong> Controls LED outputs</td>
</tr>
<tr>
<td>0x28000021</td>
<td><strong>CLK_COUNTER</strong> Contains contents of ACLK counter</td>
</tr>
<tr>
<td>0x28000022</td>
<td><strong>CLK_COUNTER</strong> Contains contents of BCLK counter</td>
</tr>
<tr>
<td>0x28000023</td>
<td><strong>CLK_COUNTER</strong> Contains contents of DCLK counter</td>
</tr>
<tr>
<td>0x28000024</td>
<td><strong>CLK_COUNTER</strong> Contains contents of SYSCLK counte</td>
</tr>
<tr>
<td>0x28000001</td>
<td><strong>DDR2HIADDR</strong> upper address bits for DDR2 interface</td>
</tr>
<tr>
<td>0x28000003</td>
<td><strong>HIADDRSIZE</strong> number of bits in DDR2HIADDR</td>
</tr>
<tr>
<td>0x28000005</td>
<td><strong>DDR2SIZEHIADDR</strong> The size of the DDR2 module.</td>
</tr>
<tr>
<td>0x28000007</td>
<td><strong>DDR2TAPCNT0</strong> Current DELAY values of DDR2...</td>
</tr>
<tr>
<td>0x28000008</td>
<td><strong>DDR2TAPCNT1</strong> …interface</td>
</tr>
</tbody>
</table>
2.1 Using the Reference Design

3 Memory Mapped Data flow

All memory mapped transactions in the reference design occur over the MB bus. This 40-signal bus connects to all Virtex 4 FPGAs and to the Spartan II configuration FPGA. The Configuration circuit (Spartan 2) is the master of the bus. All access to the MB bus (reads and writes) is initiated by the Spartan II FPGA when the reference design is in use.

All transfers are synchronous to the USB_CLK (or SYS_CLK) signal. This clock is fixed at 48Mhz, and cannot be changed by the user. This clock is LVCMOS, single-ended. When the ALE signal is asserted by the configuration circuit, the slave device on the bus (the FPGA) is required to register the data on the AD bus. This is the “main bus address”. All future transfers over the main bus are said to be at this address, until a new address is latched. On a later clock cycle, the master may assert the “RD” signal. Some time after this, (within 256 clock cycles), the FPGA should assert DONE for one clock cycle. On this cycle, the master (Spartan) will register the data on the AD bus, and that will be the read data. If DONE is not asserted, then a timeout will be recorded and the transaction cancelled.

Here is a write transaction:
When the “WR” signal is asserted by the Spartan, the FPGA should register the data on the AD bus. (Note that by convention, FPGAs on the main bus are assigned the address range corresponding to one value of the highest nibble of the address. Hex addresses 0XXXXXXXX are FPGA A, 1XXXXXXX are FPGA B and 2XXXXXXX are fpga C.)

Some time after this, the FPGA should assert the DONE signal. This will allow the Spartan to begin more transactions. The FPGA may delay this for up to 256 clock cycles before a timeout is recorded and the transaction is cancelled.

Main bus can be controlled from the USB Controller program. (Read and write single addresses, or to/from files) It can also be written from the main.txt configuration method. The main.txt syntax is

MAIN BUS 0x<addr> 0x<data>

Where <addr> and <data> are 8-digit (32-bit) hexadecimal numbers.

### 3.1 Compiling the Reference Design

This section deals with the source code to the Reference Design, which can be found on the CD-ROM. All file references are with respect to the root directory of the Reference Design source code (/source/FPGA). Files that are specific to the DN8000K10PSX design are found in the DN8000K10PSX subdirectory, whereas general application code is found in the common subdirectory.

#### 3.1.1 Xilinx XST

The Dini Group uses XST software to for design synthesis. The XST projects for each of the 3 FPGAs on the DN8000K10PSX can be found at ‘buildxst/*.xst’. These projects have been created using XST Version 9.1
3.1.2 **Xilinx ISE**
Use HDL files as input. Modification of the ISE project may also require modification of the HDL, timing constraints are in tfiles ‘buildxst/*.xcf’.

3.1.3 **The Build Utility: Make.bat**
The Build Utility is found at

D:\FPGA_Reference_Designs\dn8000k10sxpci\MainRef\build\make.bat

This batch file calls XST and Xilinx ISE tools to create configuration files using the provided reference design source code. The same set of source files compile to become any of the three FPGAs on the DN8000K10PSX, using any FPGA type (SX55, LX40, LX80, LX160), so command-line parameters are used to tell the script how to configure the synthesis and place-and-route tools for the correct FPGA target.

Instructions for invoking the batch file can be found by viewing the batch file with a text editor. Additional information about using the batch file to build the reference design is found below.

The make.bat file will not work correctly if double-clicked from the windows environment. You should open a command prompt and run it using the correct command-line options. The directory structure that the batch file expects (with relation to the source files) is identical to the structure found on the user CD. You should copy the entire CD directory structure to your hard drive. When the batch file runs, it will create output directories: one for the synthesis output and one for the ISE output (ncd file and .bit file). If you ran the “make all” command, then the created “out” directory will contain three bit files for the three FPGAs on your board.

The command-line options of the batch file allow partial compilation, allowing you to only run the synthesis step, or only the place-and-route step (implement), or only run the process for an individual FPGA.

For the batch file to work correctly, the Xilinx tools must be in your PATH environment variable.

You may have to set the FPGA type before running make all. To change the FPGA type run “make SX55_X” to configure for SX55 FPGAs “make LX160_X” to configure for LX160 FPGAs

4 **Getting More Information**

4.1 **Printed Documentation**
The printed documentation, as mentioned previously, takes the form of a Virtex 4 datasheet and a DN8000K10PSX User Guide.

4.2 **Electronic Documentation**
Multiple documents and datasheets have been included on the CD.
4.3 Online Documentation
There is a public access site that can be found on the Dini Group web site at http://www.dinigroup.com/.

4.4 Support
Support can be reached at the email address support@dinigroup.com

Before contacting support please make the following common error checks on your design:

Make sure that the clock your design uses is running. Output the clock to an LED and probe it with an oscilloscope.

Check the pinout in your constraint file. Check the .PAR report file to make sure that 100% of your IOBs used have LOC constraints. There is never a reason not to constrain an IO.

Use the .PAD report to make sure your constraints were all applied. Some situations may cause constraints to be ignored.

Double-check that the connections match between your FPGA pins and the daughtercard pins using the schematic.

If “MainBus” interface is not working, make sure that none of the other FPGAs are driving those MB pins.

Make sure that the "Unused IOBs" option in bitgen is set to "Float". The “DRIVE DONE” option in bit gen should be set to “TRUE”
Chapter 6: Ordering Information

Part Number
DN8000K10PSX

1 FPGA Options

1.1 FPGA A:
Select an FPGA part to be supplied in the A position. This FPGA is connected to the PCI bus, an expansion header, and can source global clocks. The –12 speed grade is required for full speed operation (1Gbs/pair) of the interconnect between fpgas.

NONE
LX100 –10 –11 –12
LX160 –10 –11 –12
LX200 –10 –11

1.2 FPGA B:
Select an FPGA part to be supplied in the B position. This FPGA is connected to an expansion header, a memory module socket, and can source global clocks. The –12 speed grade is required for full speed operation (1Gbs/pair) of the interconnect between FPGAs.

NONE
LX100 –10 –11 –12
LX160 –10 –11 –12
LX200 –10 –11

1.3 FPGA C:
Select an FPGA part to be supplied in the C position. This fpga is connected to a memory module socket. This FPGA is required to provide Multi-Gigabit serial communication. In order to achieve 10 Gbs selectIO operation, the –12 speed grade is required.

NONE
FX40 –10 –11 -11x –12 (This option makes the 200-pin SODIMM memory socket, one SMA channel and one QSE cable channel unusable)

FX60 –10 – 11 -11x –12 (This option makes one channel of SMA and one channel of 5Gb QSE cable unusable)

FX100 –10 –11 -11x –12

### 2 Optional Equipment

The Dinigroup supplies standard daughtercards and memory modules that you can use with the DN8000K10PSX.

- SRAM module for use in the 200-pin SODIMM sockets of the DN8000K10PSX. QDRII, 300Mhz 64x2Mb
- DDR2 modules (128MB, 256 MB, 512 MB, 1GB, 2GB soon)
- SRAM module for use in the 200-pin SODIMM socket. 64x2Mb Standard SDR SRAM. Pipelined or Flowthrough, NoBL available
- RLDRAM module for use in the 200-pin SODIMM socket. 64x16Mb, 300Mhz DDRII
- Flash module for use in the 200-pin SODIMM header.
- Mictor module for use in the 200-pin SODIMM header. (2 Mictor 38 connectors for use with logic analyzer)
- Standard Daughtercard DNMEG_OBS