Table of Contents

Chapter 1    Getting Started

The DINI Group Technical Support .......................... 1-1
Relevant Information ................................. 1-1
Conventions ........................................ 1-3

Chapter 2    DN5000k10S Features, Overview and General Description

DN5000k10S Features ................................. 2-1
DN5000k10S Description .......................... 2-2
  Easy Configuration via SmartMedia .............. 2-3
FPGA — Stratix (U11, F) .......................... 2-3
  Flip-Flops and LUTs ............................... 2-4
  Embedded Memory ................................. 2-5
  Multipliers ........................................ 2-5
  I/O Issues ....................................... 2-8
  Bitstream Encryptions ........................... 2-9
µP and FPGA Configuration .......................... 2-9
  The µP: Some Details ............................ 2-10
P1: Unused µP Connections ........................ 2-10
ATmega128L JTAG Interface ........................ 2-10
Programming the ATmega128L (U8) ................. 2-13
  Detailed Instructions ............................ 2-13
CPLD—EPM3256A .................................... 2-15
  Some Miscellaneous Notes on the CPLD .......... 2-17
  Notes on Header P3 .............................. 2-17
Fast Passive Parallel Configuration Instructions . 2-18
  Creating RBF Files for Fast Passive Parallel . 2-18
  Setting up the Serial Port (P2 — RS232 Port) . 2-18
  Creating Main Configuration File main.txt ....... 2-20
  Starting Fast Passive Parallel Configuration ... 2-22
    Description of Main Menu Options ............... 2-23
SmartMedia ......................................... 2-24
Synthesis and Emulation Issues ....................... 2-25
Synthesis Notes .................................... 2-26
Chapter 3  PCI

Overview .................................................. 3-1

PCI Mechanical Specifications ......................... 3-1

Some Notes on the DN5000k10S and PCI/PCI-X .... 3-1


JP3: M66EN—66MHz Enable .......................... 3-5

TP13: PME—, Power Management Enable .......... 3-6

JP3: PCI/PCI-X Capability ............................. 3-6

JP3—PCIXCAP ........................................... 3-6

Chapter 4  Clocks and Clock Distribution

Functional Overview ...................................... 4-1

Clock Grid .................................................. 4-2

Orientation and Description ........................... 4-2

Jumper Control for the Most Common Applications .. 4-3

Ribbon Cable: Providing an Off-Board Clock to the
DN5000k10S ................................................ 4-4

Roboclock PLL Clock Buffers ........................... 4-5

Jumper Descriptions ...................................... 4-5

General Control ........................................... 4-9

Feedback and Clock Multiplication .................... 4-9

Clock Division ............................................. 4-9

Clock Skew .................................................. 4-10

Differential Clocks ........................................ 4-11

Useful Notes and Hints .................................. 4-12

Customizing the Oscillators ............................. 4-12

DN5000k10S PCI_CLK Operation ....................... 4-14

PCI_CLK Details .......................................... 4-14

GCLKOUT .................................................. 4-15

Header Clocks ............................................. 4-15

DCLK[7](R) .................................................. 4-15

Chapter 5  Memories

SSRAMs .................................................. 5-1

SSRAM Notes ............................................. 5-1

Pipeline, Flowthrough, ZBT ............................. 5-7

SDRAM ...................................................... 5-9

SDRAM On-Board Options .............................. 5-11

DDR SDRAM ............................................... 5-12
Chapter 6  Power Supplies and Power Distribution

+3.3 V Power ................................................. 6-2
+2.5 V Power ................................................. 6-2
+1.5 V Power ................................................. 6-3
Stand-Alone Operation ................................. 6-3

Chapter 7  Daughter Connections to DN3000k10SD—Observation Daughter Card for 200-pin Connectors

Purpose ......................................................... 7-1
Features ......................................................... 7-1
Daughter Card LEDs ........................................ 7-4
Power Supply ................................................... 7-4
Options ......................................................... 7-5
Power Rating .................................................... 7-5
Connector J8 ..................................................... 7-5
LVDS ............................................................. 7-6
Connector P5 ..................................................... 7-6
Unbuffered I/O .................................................. 7-6
Connectors P2, P4 ............................................. 7-6
Connector P7, P1, P3 ......................................... 7-6
Buffered I/O ..................................................... 7-7
Active .......................................................... 7-7
Passive ......................................................... 7-7
Test Interface ................................................... 7-7
Connector J1 ..................................................... 7-7
Daughter Card I/O Connections ...................... 7-8

Chapter 8  Reset Schemes, LEDs, Bus Bars and 200 Pin Connectors

Reset Schemes .................................................. 8-1
LEDs ............................................................. 8-3
Bus Bars ........................................................ 8-4
Chapter 9 Utilities

PCI Debug—General Pontificating .................. 9-1
PC-Based—AETEST.EXE. ............................. 9-1

AETEST Utility Installation Instructions ........... 9-2
  Installation Instructions for DOS .................. 9-2
  Installation Instructions for Windows NT ......... 9-2
  Installation Instructions for Windows 2000 ....... 9-2
  Installation Instructions for LINUX ............... 9-3
  Installation Instructions for Solaris ............... 9-3
  Installation Instructions for Windows 98/ME .... 9-4

AETEST Options: Description and Definitions ....... 9-4
  Startup ........................................ 9-4

AETEST Main Screen ............................... 9-6
  Options ...................................... 9-6

PCI Menu .......................................... 9-7
Memory Menu ...................................... 9-9

Appendix A Berg Connector Datasheets
List of Figures

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>DN5000k10S Block Diagram</td>
<td>2-2</td>
</tr>
<tr>
<td>2-2</td>
<td>General LE Diagram</td>
<td>2-4</td>
</tr>
<tr>
<td>2-3</td>
<td>Dual-Port Data Flows</td>
<td>2-5</td>
</tr>
<tr>
<td>2-4</td>
<td>DSP Block Diagram</td>
<td>2-6</td>
</tr>
<tr>
<td>2-5</td>
<td>Multiplier Sub-Component Block Diagram</td>
<td>2-7</td>
</tr>
<tr>
<td>2-6</td>
<td>DN5000k10SBlock Diagram of ATmega128L and DN5000k10S Interfaces</td>
<td>2-11</td>
</tr>
<tr>
<td>2-7</td>
<td>P1: Unused µP Connections</td>
<td>2-12</td>
</tr>
<tr>
<td>2-8</td>
<td>P7 JTAG Interface</td>
<td>2-12</td>
</tr>
<tr>
<td>2-9</td>
<td>P5 Schematic</td>
<td>2-13</td>
</tr>
<tr>
<td>2-10</td>
<td>Location of P4 on the DN5000k10S</td>
<td>2-16</td>
</tr>
<tr>
<td>2-11</td>
<td>P2 Serial Port Locations</td>
<td>2-19</td>
</tr>
<tr>
<td>2-12</td>
<td>Delkin 32 MB 3.3 V Smart Media Card</td>
<td>2-25</td>
</tr>
<tr>
<td>3-1</td>
<td>FPGA Pin Connections for PCI Signals</td>
<td>3-2</td>
</tr>
<tr>
<td>3-2</td>
<td>PCI/PCI-X Edge Connector</td>
<td>3-3</td>
</tr>
<tr>
<td>3-3</td>
<td>DN5000k10S Dimensions</td>
<td>3-4</td>
</tr>
<tr>
<td>3-4</td>
<td>JP2 PCI-X Present Header</td>
<td>3-5</td>
</tr>
<tr>
<td>3-5</td>
<td>PCI-X/M66EN Capability Header</td>
<td>3-6</td>
</tr>
<tr>
<td>4-1</td>
<td>Clock Distribution Block Diagram</td>
<td>4-1</td>
</tr>
<tr>
<td>4-2</td>
<td>Clock Grid</td>
<td>4-3</td>
</tr>
<tr>
<td>4-3</td>
<td>PECL Clock Input and Termination</td>
<td>4-4</td>
</tr>
<tr>
<td>4-4</td>
<td>External Ribbon Cable Connections</td>
<td>4-5</td>
</tr>
<tr>
<td>4-5</td>
<td>Functional Diagram of Roboclock 1 and Roboclock 2</td>
<td>4-6</td>
</tr>
<tr>
<td>4-6</td>
<td>Header Layout</td>
<td>4-7</td>
</tr>
<tr>
<td>4-7</td>
<td>Clock OE Pin Jumper Settings</td>
<td>4-13</td>
</tr>
<tr>
<td>4-8</td>
<td>PCI_CLK PLL Circuit</td>
<td>4-15</td>
</tr>
<tr>
<td>5-1</td>
<td>SSRAM 1 (U9) Bus Signals</td>
<td>5-2</td>
</tr>
<tr>
<td>5-2</td>
<td>SSRAM 2 (U10) Bus Signals</td>
<td>5-3</td>
</tr>
<tr>
<td>5-3</td>
<td>SSRAM 3 (U8) Bus Signals</td>
<td>5-4</td>
</tr>
<tr>
<td>5-4</td>
<td>SSRAM 4 (U13) Bus Signals</td>
<td>5-5</td>
</tr>
<tr>
<td>5-5</td>
<td>Syncburst FT</td>
<td>5-7</td>
</tr>
<tr>
<td>5-6</td>
<td>Syncburst PL</td>
<td>5-7</td>
</tr>
<tr>
<td>5-7</td>
<td>Syncburst ZBT FT</td>
<td>5-8</td>
</tr>
<tr>
<td>5-8</td>
<td>Syncburst ZBT PL</td>
<td>5-8</td>
</tr>
<tr>
<td>5-9</td>
<td>Syncburst and ZBT SSRAM Timing</td>
<td>5-8</td>
</tr>
<tr>
<td>5-10</td>
<td>SDRAM (J19) Bus Signals (Page 1 of 2)</td>
<td>5-10</td>
</tr>
<tr>
<td>5-11</td>
<td>SDRAM (J19) Bus Signals (Page 2 of 2)</td>
<td>5-11</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-12</td>
<td>DDR PLL Circuit Block Diagram</td>
<td>5-13</td>
</tr>
<tr>
<td>5-13</td>
<td>DDR Clock Select Jumper (JP4)</td>
<td>5-13</td>
</tr>
<tr>
<td>6-1</td>
<td>DN5000k10S Power Distribution</td>
<td>6-1</td>
</tr>
<tr>
<td>6-2</td>
<td>Molex Connector P1—Auxiliary Power</td>
<td>6-4</td>
</tr>
<tr>
<td>6-3</td>
<td>Example ATX Power Supply</td>
<td>6-4</td>
</tr>
<tr>
<td>7-1</td>
<td>DN3000k10SD Daughter Card Block Diagram</td>
<td>7-2</td>
</tr>
<tr>
<td>7-2</td>
<td>DN3000k10SD Daughter Card</td>
<td>7-3</td>
</tr>
<tr>
<td>7-3</td>
<td>DN3000k10SD Daughter Card Assembly Drawing</td>
<td>7-4</td>
</tr>
<tr>
<td>8-1</td>
<td>Reset Functionality</td>
<td>8-2</td>
</tr>
<tr>
<td>8-2</td>
<td>DN5000k10S LEDs</td>
<td>8-3</td>
</tr>
<tr>
<td>8-3</td>
<td>DN5000k10S LED Diagram</td>
<td>8-3</td>
</tr>
<tr>
<td>8-4</td>
<td>91294-003 Pin Numbering</td>
<td>8-5</td>
</tr>
<tr>
<td>8-5</td>
<td>200-Pin Connectors — Signal Connections</td>
<td>8-7</td>
</tr>
<tr>
<td>9-1</td>
<td>DN5000k10SAETEST Startup Screen, DN5000k10S</td>
<td>9-4</td>
</tr>
<tr>
<td>9-2</td>
<td>AETEST Startup Screen, No PCI Peripheral</td>
<td>9-5</td>
</tr>
<tr>
<td></td>
<td>Recognized</td>
<td></td>
</tr>
<tr>
<td>9-3</td>
<td>AETEST Main Screen</td>
<td>9-6</td>
</tr>
<tr>
<td>9-4</td>
<td>AETEST PCI Menu</td>
<td>9-7</td>
</tr>
<tr>
<td>9-5</td>
<td>AETEST Memory Menu</td>
<td>9-9</td>
</tr>
<tr>
<td>9-6</td>
<td>AETEST Write to Memory Test</td>
<td>9-10</td>
</tr>
<tr>
<td>9-7</td>
<td>AETEST Read Memory Test</td>
<td>9-10</td>
</tr>
<tr>
<td>9-8</td>
<td>AETEST Write/Read Test</td>
<td>9-11</td>
</tr>
<tr>
<td>9-9</td>
<td>AETEST Memory Fill</td>
<td>9-11</td>
</tr>
<tr>
<td>9-10</td>
<td>AETEST Memory Display</td>
<td>9-12</td>
</tr>
<tr>
<td>9-11</td>
<td>AETEST Write Memory Byte</td>
<td>9-12</td>
</tr>
<tr>
<td>9-12</td>
<td>AETEST Read Memory Byte</td>
<td>9-13</td>
</tr>
<tr>
<td>9-13</td>
<td>AETEST Write/Read Memory Byte</td>
<td>9-13</td>
</tr>
<tr>
<td>A-1</td>
<td>Berg 91403-003 Datasheet Page 1 of 2</td>
<td>A-2</td>
</tr>
<tr>
<td>A-2</td>
<td>Berg 91403-003 Datasheet Page 2 of 2</td>
<td>A-3</td>
</tr>
<tr>
<td>A-3</td>
<td>Berg 91294-003 Datasheet Page 1 of 3</td>
<td>A-4</td>
</tr>
<tr>
<td>A-5</td>
<td>Berg 91294-003 Datasheet Page 3 of 3</td>
<td>A-6</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>TABLE</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>DN5000k10S Stuffing Option Comparison</td>
<td>2-3</td>
</tr>
<tr>
<td>2-2</td>
<td>Signals and Connections to P4</td>
<td>2-16</td>
</tr>
<tr>
<td>2-3</td>
<td>FPGA Serial/JTAG Configuration Header</td>
<td>2-17</td>
</tr>
<tr>
<td>2-4</td>
<td>JP1 Configuration Jumper Settings</td>
<td>2-22</td>
</tr>
<tr>
<td>2-5</td>
<td>Stratix FPGA Approximate File Sizes</td>
<td>2-24</td>
</tr>
<tr>
<td>3-1</td>
<td>Present Signal Definitions</td>
<td>3-5</td>
</tr>
<tr>
<td>3-2</td>
<td>M66EN Jumper Descriptions</td>
<td>3-5</td>
</tr>
<tr>
<td>3-3</td>
<td>PCIXCAP Jumpers</td>
<td>3-6</td>
</tr>
<tr>
<td>3-4</td>
<td>M66EN and PCIXCAP Encoding</td>
<td>3-7</td>
</tr>
<tr>
<td>4-1</td>
<td>Clock Grid Signal Descriptions</td>
<td>4-2</td>
</tr>
<tr>
<td>4-2</td>
<td>Header Classification</td>
<td>4-7</td>
</tr>
<tr>
<td>4-3</td>
<td>Jumper Definitions</td>
<td>4-8</td>
</tr>
<tr>
<td>4-4</td>
<td>Frequency Range Settings</td>
<td>4-9</td>
</tr>
<tr>
<td>4-5</td>
<td>Output Divider Settings</td>
<td>4-10</td>
</tr>
<tr>
<td>4-6</td>
<td>Time Unit N-factor</td>
<td>4-10</td>
</tr>
<tr>
<td>4-7</td>
<td>Clock Skew Settings</td>
<td>4-11</td>
</tr>
<tr>
<td>4-8</td>
<td>LVPECL Input Specifications</td>
<td>4-11</td>
</tr>
<tr>
<td>4-9</td>
<td>Clock OE Pin Jumper Settings</td>
<td>4-13</td>
</tr>
<tr>
<td>5-1</td>
<td>Requirements for Non-Standard SSRAMs</td>
<td>5-6</td>
</tr>
<tr>
<td>5-2</td>
<td>Syncburst and ZBT SSRAM Timing</td>
<td>5-9</td>
</tr>
<tr>
<td>6-1</td>
<td>Specification for +3.3 V Power</td>
<td>6-2</td>
</tr>
<tr>
<td>6-2</td>
<td>Specification for +2.5 V Power and +1.25 V</td>
<td>6-3</td>
</tr>
<tr>
<td></td>
<td>Reference</td>
<td></td>
</tr>
<tr>
<td>6-3</td>
<td>Specification for +1.5 V Power</td>
<td>6-3</td>
</tr>
<tr>
<td>7-1</td>
<td>Connector J8 Pins External Power</td>
<td>7-5</td>
</tr>
<tr>
<td>7-2</td>
<td>Daughter Board-Header-FPGA Pin Map</td>
<td>7-8</td>
</tr>
</tbody>
</table>
Getting Started

The DN5000k10S is sensitive to static electricity, so treat the PWB accordingly. The target market for this product is engineers that are familiar with FPGAs and circuit boards, so a lecture in ESD really isn’t appropriate (and wouldn’t be read anyway). However, we have sold some of these units to people who are not as familiar with this issue. The following web page has an excellent tutorial on the Fundamentals of ESD for those of you who are new to ESD-sensitive products:


The DINI Group Technical Support

The following means of technical support are available:

1. The DN5000k10S User’s Manual. This is the main source of technical information. We strive to produce excellent documentation, and this manual should contain most of the answers to your questions.


3. E-Mail to support@dinigroup.com. You may direct questions and feedback to The DINI Group using this e-mail address.

4. Phone Support. We are happy to help. Call us at (858) 454-3419 during the hours of 8:00 A.M. to 5:00 P.M. Pacific Time. Some of us get in early and stay late, so you might try us outside of these hours also.

5. Frequently Asked Questions. In the downloads section of our web page you can find a document called DN5000k10/S Frequently Asked Questions (FAQ). We will update this document occasionally with information that may not be in the User’s Manual.

Relevant Information

Information about PCI can be obtained from the following sources:

The PCI Special Interest Group has a web page that has lots of good stuff. Copies of the latest PCI specification may be ordered here.

http://www.pcisig.com/
PCI Special Interest Group
2575 NE Kathryn St. #17
Hillsboro, OR 97124
FAX: (503) 693-8344
As of June 2003, the most current versions of the PCI Specifications are:

- **PCI Local Bus Specification, Revision 3.0**
- **PCI Hot-Plug Specification, Revision 2.0**
- **PCI Power Management Interface Specification, Revision 1.1**
- **PCI-X Addendum to the PCI Local Bus Specification, Revision 1.0a**

Other recommended specifications include:

- **PCIMG 2.0 Compact PCI Specification, Revision 2.1 (or greater)**
- PCI Industrial Computer Manufacturers Group (PICMG)
  401 Edgewater Place, Suite 500
  Wakefield, MA 01880, USA
  TEL: 781-224-1100
  FAX: 781-224-1239
  [http://www.picmg.org](http://www.picmg.org)

The best book to get if you need an introduction to PCI is:

- **PCI System Architecture**
  Fourth Edition
  MindShare, Inc.
  Tom Shanley and Don Anderson

Ignore some of the ignorant statements made in the Customer Review section at [http://www.amazon.com/](http://www.amazon.com/). This is an excellent book for PCI and well worth the money.

The best book to get if you need an introduction to PCI-X is:

- **PCI-X System Architecture**
  MindShare, Inc.
  Tom Shanley and Karen Gettman

You are going to need to know Verilog or VHDL to use the Stratix FPGA. If you need a reference, we recommend the following book for Verilog:

- **Verilog HDL: A Guide to Digital Design and Synthesis**
  Samir Palnitkar

If you are one of those people that actually like VHDL, we feel sorry for you. The following books may be helpful:

- **Essential VHDL: RTL Synthesis Done Right**
  Sundar Rajan

- **The IQ Booster: Improve Your IQ Performance Dramatically**
  Edwin Breecher
Conventions

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

- The term PCI-X will be used generically unless there is a specific instance where PCI applies.

- This design guide generically refers to PCI-X protocol. When the PCI-X HalfBridge core is in PCI mode, PCI protocol will be followed.

- **Courier font** denotes the following items:
  - Signals on PCI Bus side of the PCI-X Interface
    
    `FRAME_IO` (PCI-X Interface signal name)
    
    `FRAME#` (PCI-X Bus signal name)
  - Signals within the user application
    
    `BACK_UP, START`
  - Command line input and output
    
    `setenv XIL_MAP_LOC_CLOSED`
  - HDL pseudocode
    
    `assign question = to_be | !to_be;`
    
    `assign cannot = have_cake & eat_it;`
  - Design file names
    
    `pcim_top.v, pcim_top.vhd`

- **Courier bold** denotes the following items:
  - Signals on the user side of the LogiCORE PCI-X Interface
    
    `ADDR_VLD`
  - Menu selections or button presses
    
    `FILE -> OPEN`

- Italic font denotes the following items:
  - Variables in statements which require user-supplied values
    
    `ngdbuild design_name`
  - References to other manuals

See the *Libraries Guide* for more information.
Getting Started

- Emphasis in text
  
  It is not a bug, it is a feature.
- Dark shading indicates items that are not supported or reserved:

<table>
<thead>
<tr>
<th>SDONE_I</th>
<th>in/out</th>
<th>Snoop Done signal. Not Supported.</th>
</tr>
</thead>
</table>

- Square brackets “[ ]” indicate an optional entry or a bus index:

  ngdbuild [option_name] design_name

  DATA[31:0]
- A vertical or horizontal ellipsis indicates repetitive material that has been omitted.

  A B C... X Y Z
- The use of “fn(SIG1. . . SIGn)” in an HDL pseudocode fragment should be interpreted as “combinational function of signals SIG1 through SIGn.

  SUM = fn(A, B, Cin);
- The prefix “0x” or the suffix “h” indicate hexadecimal notation.

  A read of address 0x00110373 returned 45524943h.
- A “#” an “_n”, an “n” or a “–” means the signal is active low

  INT# is active low.

  fpga_inta_n is active low.

  SRAMCS– is active low.

  FPGA_GRSTn is active low.
Chapter 2

DN5000k10S Features, Overview and General Description

DN5000k10S Features

The DN5000k10S features include:

- 32/64-bit, +3.3V, PCI/PCI-X-based PWB with a single Altera Stratix™ FPGA (FBGA1508).
  - Device availability: EP1S80, EP1S60 and EP1S40
  - ~450,000 ASIC gates (with EP1S80 — LSI standard)

- Fast/Easy FPGA configuration via standard SmartMedia FLASH card
  - Microprocessor controlled (ATmega128L)
  - RS232 port for configuration/operation status and control
  - Fastest possible configuration speed (via Passive Parallel method)

- 10A on-board linear regulator for +3.3V and +1.5V
  - Standalone operation via separate power connector
  - +3.3V not needed on backplane

- 6 low skew clocks distributed to the FPGA and test connectors:
  - 2 CY7B993/4 RoboclockII PLLs
  - 2 socketed oscillators
  - PCI Clock
  - 1 dividable clock via CPLD

- Robust observation/debug with 242 connections for logic analyzer observability or for pattern generator stimulus.

- Status LEDs.

- User-designed daughter PWB for custom circuitry and interfaces.

<table>
<thead>
<tr>
<th>Device</th>
<th>I/O</th>
<th>Flip-Flops</th>
<th>18 x 18 Multipliers</th>
<th>Embedded Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>M512 RAM</td>
</tr>
<tr>
<td>EP1S40</td>
<td>822</td>
<td>41,250</td>
<td>56</td>
<td>384</td>
</tr>
<tr>
<td>EP1S60</td>
<td>1022</td>
<td>57,120</td>
<td>72</td>
<td>574</td>
</tr>
<tr>
<td>EP1S80</td>
<td>1203</td>
<td>79,040</td>
<td>88</td>
<td>767</td>
</tr>
</tbody>
</table>
- SignalTap and Identify (from Synplicity) fully supported via JTAG interface.

Figure 2-1 shows a block diagram of the DN5000k10S.

**DN5000k10S Description**

The DN5000k10S is a complete logic emulation system that enables ASIC or IP designers a vehicle to prototype logic and memory designs for a fraction of the cost of existing solutions. The DN5000k10S can be hosted in a 32/64-bit PCI/PCI-X slot, or can be used as a stand-alone device. A single DN5000k10S stuffed with a single EP1580 can emulate up to 450,000 gates of logic as measured by LSI. A high I/O-count, 1508-pin, flip-chip BGA package is employed. The F1508 package has 1203 I/Os, which allows for abundant connections to daughter connectors and external memories. A total of 242 test pins are provided on the top of the PWB via high-density connectors for logic analyzer-based debugging, or for pattern generator stimulus. Custom daughter cards such as the DN3000k10SD can be mounted to these connectors as a means of interfacing the DN5000k10S to application-specific circuits. A reference 32-bit PCI target design and test bench is provided in Verilog and VHDL at no additional cost.
The configuration bit files for the FPGA are copied onto a 32-megabyte SmartMedia FLASH card (provided) and an on-board microprocessor controls the FPGA configuration process. Visibility into the configuration process is enhanced with an RS232 port. FPGA configuration runs quickly at 48 MHz. Eight LEDs provide instant status and operational feedback. Four of these LEDs are connected to the CPLD and can be user-configured.

FPGA — Stratix (U11, F)

The DN5000k10S contains one Stratix™ FPGA. The package is a flip chip fine-pitch BGA with 1508 pins (F1508). The pitch on the pins is 1 mm. This isn’t important, but this pin density makes the PWB a bitch to layout. Keep that in mind if you try to make one of these at home. Most of the 1203 I/O pins are utilized on the F1508 package. The standard speed grade we stuff is –7. We can use the –6 speed grade, but don’t fall out of your chair when you get the price. Note that Altera seems to have cancelled plans for the EP1S120. Although this part appears in some Altera literature, we haven’t seen any scheduled release date or other documentation for it. Don’t expect to see anything larger than the EP1S80 until at least the 2004 time frame. Table 2-1 shows the stuffing options for the DN5000k10S.

**Table 2-1  DN5000k10S Stuffing Option Comparison**

<table>
<thead>
<tr>
<th>Stuffed FPGA</th>
<th>SSRAMs</th>
<th>SDRAM</th>
<th>DDR SDRAM</th>
<th>Total Header Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>EP1S40_1508</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>77</td>
</tr>
<tr>
<td>EP1S60_1508</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>180</td>
</tr>
<tr>
<td>EP1S80_1508</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>242</td>
</tr>
</tbody>
</table>

The following is a very brief overview of the Stratix family. More information can be gleaned from the Stratix Datasheet (ds_stx.pdf). This file is on the CD-ROM supplied with the DN5000k10S, but you are better off getting the latest version from the Altera Web page (http://www.altera.com/). Make sure to get the latest errata sheet also.
Flip-Flops and LUTs

Figure 2-2 shows what Altera calls a Logic Element, or LE. Each LE contains a flip-flop and a 4x1 look-up table (LUT). LEs are arranged in groups of 10, called Logic Array Blocks (LAB). The EP1S80 is an array of LABs with 91 rows and 101 columns, but there are 9 RAM blocks which appear in place of 13-row by 11-column sections of the grid, leaving a total of 7904 LABs and 19040 LEs. (Other blocks, such as DSP/multiplier blocks and smaller RAM, are arranged in entire columns squeezed between two LAB columns.)

Each LUT can implement any Boolean function of four inputs. An LUT can also be configured as a two-input adder/subtractor with a carry chain coming from the adjacent LE and going to the next LE. In order to reduce delays caused by long carry chains, each set of 5 LEs computes two adder results simultaneously, then uses the carry result from the previous set of 5 to select which result is correct.

The flip-flop in each LE includes a clock enable input, an asynchronous preset and reset, synchronous set and reset logic, and an asynchronous load function. Data input can come from the LUT in the same LE to register addition or boolean outputs, or the LUT and flip-flop can be used independently of each other. For more information, check www.altera.com for the Stratix datasheet.
Embedded Memory

Stratix has boatloads of embedded memory. The EP1S80 contains 767 blocks of 576 bits, 364 blocks of 4.5 Kbits, and 9 blocks of 576 Kbits. The smallest memory blocks (called M512 RAM) can be configured for data widths ranging from 32 x 18 bits to 512 x 1 bit; medium-sized blocks (M4K RAM) can be configured ranging from 128 x 36 bits to 4K x 1 bit; and the largest blocks (M-RAM) can be configured anywhere from 4K x 144 bits to 64K x 9 bits. The embedded memory is dual-ported, and can be used to construct almost any type of memory - FIFOs, dual-port RAMs, single-port RAMs, etc.

The two largest blocks, M-RAM and M4K RAM, are fully dual-ported memory, with read and write functions available on two separately clocked ports. M512 RAM is a “simple dual-port” memory, meaning that one port is write-only and the other is read-only. Any of the memory blocks can be configured as simple dual-port or single-port memory. See Figure 2-3 for a diagram of the memory.

Multipliers

Stratix devices feature a large number of multipliers grouped into what Altera calls DSP blocks (see Figure 2-4). The EP1S80 contains 22 DSP blocks, each of which can provide one 36x36 bit multiplier, four 18x18 bit multipliers, or eight 9x9 bit multipliers. Each block also contains adder/subtractor/accumulator registers which can be configured to provide many common DSP functions, such as FIR or IIR filters, FFT, or DCT, without the use of LAB resources. The Stratix datasheet (available at www.altera.com) has

![Figure 2-3 Dual-Port Data Flows](image-url)
Figure 2-4  DSP Block Diagram
more detailed information on how the multipliers and adders are configured for some common functions.

Figure 2-4 shows a DSP block configured for four 18x18 bit multipliers. A DSP block can be configured as two parallel systems of 9x9 bit multipliers, each of which is also described by Figure 2-4. The adder blocks can be used to add or subtract two or four multipliers, such as in complex multiplication, or to add a new result each clock cycle to an accumulated sum. They are also used to configure the DSP block as a 36x36 bit multiplier, with or without an accumulator.

All registers in Figure 2-4 are optional, as shown by Figure 2-5, which is a detailed view of a single 18x18 bit or 9x9 bit multiplier. Any or all of the registers may be used to pipeline the multiplier logic and improve the clock speed, or the alternate path may be used to bypass the register. Figure 2-5 also shows more detail about the optional shift register path, which makes FIR or IIR filters easy to implement.

Most synthesis tools will accept Verilog or VHDL descriptions of multipliers and infer a DSP block with the appropriate configuration. For those that don’t, Altera provides a megafunction generator to help with direct instantiation of the hardware resources. See “Synthesis and Emulation Issues” on page 2-25 for more detail.

![Multiplier Sub-Component Block Diagram](image)

**Figure 2-5 Multiplier Sub-Component Block Diagram**
I/O Issues

Terminator technology is supported on all pins. The resistors used for RDN and RUP should be 250 ohms for series termination or impedance matching I/O standards. Parallel termination requires 1000 ohm resistors for RDN and RUP. Terminator technology is a very nice feature and we recommend you use it on all I/O signals. The default IO_STANDARD attribute for the .csf file is LVTTL.

All VCCO pins are connected to either +3.3 V or +2.5 V. The VREF pins are connected to +1.5 V or +2.5 V, so the DN5000k10S does not support I/O standards that require other values of VREF. So the I/O standards supported are:

**LVTTL — Low-Voltage TTL**
The low-voltage TTL, or LVTTL, standard is a general-purpose EIA/JESD58 standard for 3.3 V applications that use the LVTTL input buffer and a Push-Pull output buffer. The standard requires a 3.3 V input and output source voltage (\(V_{CCO}\)) but does not require the use of a reference voltage (\(V_{REF}\)) or a termination voltage (\(V_{TT}\)).

**LVCMOS33 — 3.3 Volt Low-Voltage CMOS**
This standard is an extension of the LVCMOS standard (JESD8. –5). It is used in general-purpose 3.3 V applications. The standard requires a 3.3 V input/output source voltage (\(V_{CCO}\)) but does not require the use of a reference voltage (\(V_{REF}\)) or a termination voltage (\(V_{TT}\)).

**PCI-X — Peripheral Component Interface**
The PCI standard specifies support for 33 MHz, 66 MHz and 133 MHz PCI bus applications. It uses a LVTTL input buffer and a Push-Pull output buffer. This standard does not require the use of a reference voltage (\(V_{REF}\)) or a board termination voltage (\(V_{TT}\)); however, it does require 3.3 V input output source voltage (\(V_{CCO}\)).

**SSTL-3 class I and II**
SSTL-3 uses a series termination resistor on output signals and a parallel termination resistor on input signals. Stratix devices use a VREF of +1.5V to enable the appropriate resistors internally. Because SSTL-3 requires parallel termination, it is only available on banks 3, 4, 7 and 8, and on clock output signals.

**CTT**
CTT uses a parallel termination resistor on input signals, with no termination resistors on output signals. Stratix devices use a VREF of +1.5V to enable the appropriate resistors internally. Because CTT requires parallel termination, it is only available on I/O banks 3, 4, 7 and 8, and on clock output signals.

**SSTL-2 Class I & II**
The SSTL-2 I/O standard is a 2.5-V memory bus standard used for applications such as high-speed DDR SDRAM interfaces. This standard defines the input and output specifications for devices that operate in the SSTL-2 logic switching range of 0.0 to 2.5 V. This standard improves operation in conditions where a bus must be isolated from large stubs.
Differential SSTL-2

The differential SSTL-2 I/O standard is a 2.5-V standard used for applications such as high-speed DDR SDRAM clock interfaces. This standard supports differential signals in systems using the SSTL-2 standard and supplements the SSTL-2 standard for differential clocks. The differential SSTL-2 standard does not require an input reference voltage differential.

Bitstream Encryptions

Stratix devices have no special bitstream encryption function. The DINI Group may be able to assist with scrambling bitfiles to protect IP on the SmartMedia card, which would then be descrambled in the programming CPLD. Users should be aware, however, that the bitstream would be unprotected between the CPLD and FPGA, so they could still be examined and reverse-engineered. Our DN3000k10 products use Xilinx FPGAs which can be used to decrypt the bitstream inside the FPGA, providing complete design protection. If you are interested in this feature, please be aware that there are some issues with the Xilinx encryption feature, described in the DN3000k10 FAQ on our website.

µP and FPGA Configuration

The DN5000k10S has an ATmega128L microprocessor (µP) that is used to control the configuration process (U8). The amount of internal SRAM (4 Kbytes) was not large enough to hold the FAT needed for SmartMedia, so an external 32 k x 8 SRAM was added. The address latching function is done via an LVT373 (U3).

The microprocessor has the following responsibilities:

- Reading the SmartMedia card
- Configuring the Stratix FPGA
- Executing DN5000k10S self tests.

Other than FPGA configuration, the µP has no responsibilities. Less than 25% of the 128 Kbytes of FLASH is used for FPGA configuration and utilities, so you are welcome to use the rest of the resources of the µP for your own purposes. Instructions for customizing the µP are contained in the file Custom_ATmega128L.pdf. This file is on the CD-ROM, or it can be downloaded from the DINI Group web page.

REMEMBER: You can use the microprocessor for your own purposes!

We ship a programming cable for the ATmega128L with the DN5000k10S. Updates to the code will be posted on our web site. If you wish to do your own development you will need the compiler, which we do not ship with the product. The compiler is available from IAR (http://www.iar.com/). The part number is EWA90PCUBLV150.

Note that if you are willing to program the FPGA with the JTAG or serial cable, the CPLD and the µP have no function. In this case you can use all of the resources of the µP for your own purposes.
**The µP: Some Details**

The ATmega128L is gross overkill for the FPGA configuration function. The datasheet and user's manual are on the CD-ROM that was shipped with the DN5000k10S. The file names are `ATmega128_UM.pdf` and `ATmega128_DS.pdf`. But if you intend to use the µP for your own purposes, you should check the Atmel web page to get a copy of the latest user's manual, datasheet, and erratas. The Atmel web page is [http://www.eu.atmel.com/atmel/](http://www.eu.atmel.com/atmel/). The ATmega128L is under the section called “Flash Microcontroller, AVR 8-Bit RISC.” Most of the features are unused. A variety of test headers allow for possible use of these features. Each header and the various possible functions are described in the sections that follow. Figure 2-6 is a block diagram of the ATmega128L and its various interfaces on the DN5000k10S.

**P1: Unused µP Connections**

P1 contains connections to the ATmega128L that were not used elsewhere. These ten connections can be used for external TTL connections to the µP, externally generated interrupts, or any other function that the ATmega128L supports on these pins. Remember that the ATmega128L is not +5 V tolerant, so if you attach external TTL signals to these pins, the voltage level of these signals must not exceed +3.3 V.

The P1 schematic is shown in Figure 2-7.

**ATmega128L JTAG Interface**

The ATmega128L processor has a JTAG interface that can be used for on-chip debugging, real-time emulation, and programming of FLASH, EEPROM, fuses, and Lock Bits. In order to take advantage of the JTAG interface, you must have the Atmel AVR JTAG ICE kit (part number ATAVRJTAGICE) and AVR studio software that Atmel provides free at [www.atmel.com](http://www.atmel.com). The JTAG interface for the ATmega128L can be accessed through header P7 of the DN5000k10S (see Figure 2-8).
**Figure 2-6** DN5000k10S Block Diagram of ATmega128L and DN5000k10S Interfaces
**μP GPIO**

```
P1

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_D2</td>
<td>1</td>
</tr>
<tr>
<td>P_D4</td>
<td>3</td>
</tr>
<tr>
<td>P_D6</td>
<td>5</td>
</tr>
<tr>
<td>MISO</td>
<td>7</td>
</tr>
<tr>
<td>MOSI</td>
<td>9</td>
</tr>
</tbody>
</table>

```

**Figure 2-7  P1: Unused μP Connections**

**μP JTAG**

```
+3.3V

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF4</td>
<td>1</td>
</tr>
<tr>
<td>PF6</td>
<td>3</td>
</tr>
<tr>
<td>PF5</td>
<td>5</td>
</tr>
<tr>
<td>PF7</td>
<td>9</td>
</tr>
</tbody>
</table>
```

**Figure 2-8  P7 JTAG Interface**
Programming the ATmega128L (U8)

A cable used to reprogram the ATmega128L is shipped with the DN5000k10S. You will need to reprogram the ATmega128L if we update the code or you intend to use the processor for your own application. P5 is used for this purpose.

Figure 2-9 illustrates P5.

Figure 2-9   P5 Schematic

Detailed Instructions

1. Download the latest update for the processor and CPLD at www.dinigroup.com (file uP_CPLD.zip).

2. You will first need to reprogram the CPLD. Please see “CPLD—EPM3256A” on page 2-15 for instructions (use the file *.jed that can be found in the downloaded zip file.

3. Next, you will program the processor (ATmega128L). Connect the AVR cable that was shipped with the DN5000k10S to header P5 with the red/purple wire on the cable connected to pin 1 and connect the other end to the serial port of your PC.

4. In order to program the processor, you will need to install AVR Studio that is included on the CD that was shipped with the DN5000k10S. This software can also be downloaded at www.atmel.com.

5. From the Windows START menu, choose PROGRAMS->Atmel AVR Studio x.xx (where x.xx is the version number).

6. Once AVR Studio is open, select TOOLS->STK500/AVRISP/JTAG ICE and a new window should appear with the title STK500. At the bottom of the STK500 window, if you see:

   Detecting...FAILED!

that means either there is no power on the DN5000k10S, there is another program open that is using the serial port, or the serial cable connecting the AVR tool is not connected properly. If this happens, you should close down the window titled STK500, correct the situation, and then select TOOLS->STK500/AVRISP/JTAG ICE again. You
will not be able to continue unless you see something very similar to the following at the bottom of the **STK500** window:

```
Detecting...AVRISP found on COM1:
Getting revisions...HW: 0x01, SW Major: 0x01, SW
Minor: 0x07...OK
```

7. On the **PROGRAM** tab, select the **ATmega128** under the **DEVICE** drop down menu, and in the **FLASH** section where it says **INPUT HEX FILE**, browse and select the file **DN5000k10S_128.a90** that can be found in the downloaded zip file (**uP_CPLD.zip**) from the Dini Group website. To program the device all you need to do is hit the **PROGRAM** button in the **FLASH** section. When the programming is complete (it takes about 45 seconds) you should see a message at the bottom of the window that looks something like this:

```
Detecting...AVRISP found on COM1:
Getting revisions...HW: 0x01, SW Major 0x01, SW
Minor: 0x07...OK
Reading FLASH input file...OK
Setting device parameters, serial programming
mode...OK
Entering programming mode...OK
Erasing device...OK
Programming FLASH using block mode...100% OK
Leaving programming mode...OK
```

8. After programming the processor, close all AVR Studio windows and setup the serial port according to the section titled “Setting up the Serial Port (P2 — RS232 Port)” on page 2-18. Please note that in this situation, connecting the serial port is mandatory and the FPGA cannot be configured via the SmartMedia card until you have completed all the instructions in this section.

9. Reset the DN5000k10S by pressing **S1**. After about 5 seconds, you should see the following in the HyperTerminal window:

```
Please select the FPGA on the board:
```

Enter one of the FPGA locations on your board that contains an FPGA, and you should see the following menu:

```
1) Virtex II 1000 (FG456)
2) Virtex II 6000 (FF1152)
3) Virtex II 4000 (FF1152)
4) Virtex II 3000 (FG676)
5) Virtex II 8000 (FF1152)
6) Altera Apex II (2A40)
7) Altera Apex II (2A70)
8) Altera Stratix (EP1S80F1508C7)
```

Please enter selection (1-6): for FPGA D:

Enter option 8 for Stratix FPGAs.
10. The processor and the CPLD are now ready to configure the FPGA(s). Please see the section titled “Starting Fast Passive Parallel Configuration” on page 2-22 for further instructions.

**CPLD—EPM3256A**

Some non-volatile logic is needed to handle the counters and state machines associated with the high-speed interface to the SmartMedia card. We used an EPM3256A CPLD from Altera for this function. The datasheet is on the CD-ROM and is titled `epm3256a.pdf`. Approximately 90% of the resources of this device are utilized, so 10% are available for your own purposes. The Verilog source for the CPLD is provided on the CD-ROM. The file name is `CPLD.V`.

The CPLD performs the following functions:

- **Interface to ATmega128L µP and SRAM**
  - Clock Output to µP: BUP_CLK
  - Data/Lower Address: UPAD[7:0]
  - Upper Address: UPPADDR[15:8]
  - Control Signals: UP_ALE, UP_RDn, UP_WRn
  - SRAM Select: SRAM_CS

- **Data Retrieval from SmartMedia Card**
  - Data Bus: SM_D[7:0]
  - Control: SM_CLE, SM_ALE, SM_WE, SM_WP, SM_CE, SM_RE, SM_RDYN

- **Configuration and Clock Status Reporting:**
  - CPLD_LED[3:0], ROBO_LOCK1, ROBO_LOCK2

- **Control of FPGA Parallel Configuration**
  - Clock: FPGA_DCLK
  - Chip Select: FPGA_CSnF, FPGA_CEnF
  - Control: FPGA_nCONFF, FPGA_CDONEF, FPGA_IODONEF, FPGA_RDYNBUSYF
  - Data Bus: {FPGA_D[7:1], FPGA_D0F}
  - Mode Selector Switches: FPGA_MSEL[2:0], DIP1_0

- **Pass-Through of Serial/JTAG Cable Signals**
  - Cable: DCLK/TCK, CONF_DONE/TDO, nCONFIG/TMS, nSTATUS, DATA0/TDI
  - FPGA Chain: CPLD_TMS, CPLD_TDO, CPLD_TDI, CPLD_TCK, CPLD_TRST

- **Support for Clocking Schemes**
  - CPLD Clock Input: CLK[48]
  - Inputs from Clock Buffers: CPLD_CLK[1:0]
  - Output to Clock Grid: PCPLD_CLKOUT

- **Interface to Reset Schemes**
  - FPGA_GRSTn, PWR_RSTn

We may periodically update the CPLD. The CPLD can be reprogrammed using the Altera JTAG cable supplied with the DN5000k10S. The connections are on header P4. The relevant signals and the connections to P4 are listed in Table 2-2. Figure 2-9 shows the location of P4.
### Table 2-2  Signals and Connections to P4

<table>
<thead>
<tr>
<th>JTAG Cable</th>
<th>P4 Signal Name</th>
<th>P4 Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC</td>
<td>+3.3 V</td>
<td>4, 6</td>
</tr>
<tr>
<td>GND</td>
<td>GND</td>
<td>2, 10</td>
</tr>
<tr>
<td>TCK</td>
<td>JTAG_CPLD_TCK</td>
<td>1</td>
</tr>
<tr>
<td>TDO</td>
<td>JTAG_CPLD_TDO</td>
<td>3</td>
</tr>
<tr>
<td>TDI</td>
<td>JTAG_CPLD_TDI</td>
<td>9</td>
</tr>
<tr>
<td>TMS</td>
<td>JTAG_CPLD_TMS</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 2-10  Location of P4 on the DN5000k10S
Some Miscellaneous Notes on the CPLD

X1 is a 48 MHz oscillator. This part is soldered down to the PWB and is not intended to be user-configurable. The 48 MHz is divided down to 8 MHz in the CPLD to provide the clock for the ATmega128L µP. The processor clock signal is labeled CPuCLK (and BCpuCLK) on the schematic.

Serial and JTAG configuration of the Stratix FPGA are back off positions only—that is why those signals are connected to the CPLD. Fast Passive Parallel is the quickest configuration method, but we wanted to provide the user as many options as possible.

If you want to use 100% of the CPLD and µP for your own purposes, you can configure the FPGA using the JTAG cable.

The 48 MHz clock can be divided down in the CPLD and used to drive the PWB clock network. See Chapter 4 for a more detailed description of this option.

Notes on Header P3

Fast Passive Parallel using the SmartMedia card is the best way to configure the FPGA. Two other options exists if, for some reason, the SmartMedia card method is not applicable.

1. Serial Programming Using the Cable. Header P3 has the 5 serial connections that are used to configure the FPGA using the serial method. Table 2-3 has the pinouts. Note that this is a back-off position to SmartMedia and JTAG and should only be used in dire circumstances. Note also that the switches on P5 will need to change to reflect “slave-serial” configuration.

2. JTAG Programming.
   The JTAG connection can be used to configure the FPGA and can also be used to connect the SignalTap Logic Analyzer (See Application Note 175 at www.altera.com/literature/lit-qts.html) or other solutions such as the Bridges2Silicon system, which was recently acquired by Synplicity (see www.bridges2silicon.com). The JTAG method of configuration should be used if the SmartMedia method isn’t working. Remember that programming a Stratix part through JTAG uses a .sof file, not a .rbf file. Table 2-3 has the pinouts.

Table 2-3 FPGA Serial/JTAG Configuration Header

<table>
<thead>
<tr>
<th>Name on Schematic</th>
<th>Name on Cable</th>
<th>Header Pin (P3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCLK/TCK</td>
<td>DCLK</td>
<td>1</td>
</tr>
<tr>
<td>CONFDONE/TDO</td>
<td>CONF_DONE</td>
<td>3</td>
</tr>
<tr>
<td>DATA0/TDI</td>
<td>DATA0</td>
<td>9</td>
</tr>
<tr>
<td>nCONFIG/TMS</td>
<td>nCONFIG</td>
<td>5</td>
</tr>
<tr>
<td>nSTATUS</td>
<td>nSTATUS</td>
<td>7</td>
</tr>
</tbody>
</table>
Fast Passive Parallel Configuration Instructions

The FPGA on the DN5000k10S can be configured in Fast Passive Parallel mode using a Smart Media card. Fast Passive Parallel configuration is the easiest and quickest way to configure the FPGA. The DN5000k10S is shipped with two 32 MB Smart Media cards. One of these Smart Media cards contains reference design bit files produced for Fast Passive Parallel configuration, and files `main.txt` and `iotst.txt` that sets options for the configuration process (for description of options, see "Creating Main Configuration File main.txt" on page 2-20). This Smart Media card has been labeled with a sticker marked “reference design.” The other Smart Media card is empty and is for use with your own designs. To configure the FPGA with the reference design, please skip to “Starting Fast Passive Parallel Configuration” on page 2-22.

Creating RBF Files for Fast Passive Parallel

To create an RBF file with QuartusII software:

Go to Assignments menu and drag down to Settings. Click on Device under Compiler Settings on the left, then click the Device & Pin Options button on the right. Go to the Configuration tab, select Configuration Scheme = Fast Passive Parallel, and disable the option to Use Configuration Device. Go to the Programming Files tab, turn on Raw Binary File (.rbf), and turn off all other options. (Note: the .sof file for JTAG programming will also be created.)

The easy way to assign pins is to create your project, then open the .csf file created in a text editor. If your pinlist is formatted correctly, you can copy it and paste it into the .csf file in the section labeled “CHIP (design_name)”. Sample files on the software CD provided, found in the folder labeled Verilog, show how to format the information and provide the correct pinlist for the signal names used on the board.

Setting up the Serial Port (P2 — RS232 Port)

P2 is for an RS232 connection to a terminal. An ICL3221 (U2) provides voltage translation to RS232 levels. A cable that converts the 10-pin header to a DB9 is shipped with the DN5000k10S. This cable comes packaged with a bracket attached. Remove the bracket to eliminate the possibility of it falling on the DN5000k10S, which could short signals and damage the board. After you have removed the bracket, plug the cable into P2. P2 is not keyed—so make sure you get the orientation correct. Pin 1 is identified with the number 1 and a dot. Figure 2-11 is a cutout from the assembly drawing, and shows the location of P2 and Pin 1.

<table>
<thead>
<tr>
<th>Name on Schematic</th>
<th>Name on Cable</th>
<th>Header Pin (P3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name on Cable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial Mode</td>
<td>JTAG Mode</td>
<td></td>
</tr>
<tr>
<td>GND</td>
<td>GND</td>
<td>2, 10</td>
</tr>
<tr>
<td>VCC</td>
<td>VCC</td>
<td>4, 6</td>
</tr>
</tbody>
</table>

Table 2-3 FPGA Serial/JTAG Configuration Header
A female-to-female RS232 cable is provided with the DN5000k10S. This cable will attach directly to the RS232 port of a PC. We get our cables from Jameco (http://www.jameco.com). The part number is 132345. Male-to-female extension cables are part number 25700.

The RS232 port is configured with the following parameters:

- Bits per second: 9600
- Data bits: 8
- Parity: None
- Stop Bits: 1
- Flow control: None
- Terminal Emulation: VT100

We use the Windows-based program HyperTerminal (Hypertrm.exe). The configuration file DN5000k10S.ht is supplied on the CD-ROM or can be downloaded from our web page.

Users have the option of connecting the serial port if they wish to see any messages during the configuration process.
NOTE: It is NOT mandatory to have the serial port connection in order to configure the FPGA in SelectMAP mode. However, if an error occurs during the configuration, then without a serial port connection the user will not be able to see any error messages. In addition, without a serial port connection, a user cannot select any Main Menu options after the configuration process is complete.

Creating Main Configuration File main.txt

To control which bit file on the Smart Media card is used to configure the FPGA in SelectMAP mode a file named main.txt must be created and copied to the root directory of the Smart Media card. The configuration process cannot be performed without this file. Below is a description of the options that can be set in the file, a description of the format this file needs to follow, and an example of a main.txt file.

Options:

Verbose Level — During the configuration process, there are three different verbose levels that can be selected for the serial port messages:

- Level 0:
  - Fatal error messages
  - Sanity Check errors (e.g., RBF file was created for the wrong part, RBF file was created with wrong version of Altera tools, or Quartus options are set incorrectly)
  - Initializing message will appear before configuration
  - A single message will appear once the FPGA is configured

- Level 1:
  - All messages that Level 0 displays
  - Displays configuration type (should be Fast Passive Parallel
  - Displays current FPGA being configured if the configuration type is set to Fast Passive Parallel
  - Displays a message at the completion of configuration for each FPGA configured.

- Level 2:
  - All messages that Level 1 displays
  - Options that are found in main.txt
  - RBF file names for each FPGA as entered in main.txt
  - Maker ID, Device ID, and size of Smart Media card
  - All files found on Smart Media card
  - If sanity check is chosen, the RBF file attributes will be displayed (part, package, date, and time of the RBF file)
  - During configuration, a “.” will be printed out after each block (16 KB) has successfully been transferred from the Smart Media to the current FPGA.

Sanity Check — The Sanity Check if enabled, verifies that the RBF file was created for the right part, the right version of Altera was used, and the Quartus options were set correctly. If any of the settings found in the RBF file are not compatible with the FPGA, a message will appear from the serial port, and the user will be asked whether or not they want to continue with the RBF file. Please see the section “Creating RBF Files for
Fast Passive Parallel” on page 2-18 for details on which Quartus options need to be changed from the default settings.

**Format:**

The format of the `main.txt` file is as follows:

- The first nonempty/uncommented line in `main.txt` should be:

  ```
  Verbose level: X
  ```

  where “X” can be 0, 1 or 2. If this line is missing or X is an invalid level, then the default verbose level will be 2.

- The second nonempty/uncommented line in `main.txt` tells whether or not to perform a sanity check on the bit files before configuring an FPGA:

  ```
  Sanity check: y
  ```

  where “y” stands for yes, “n” for no. If the line is missing or the character after the “:” is not “y” or “n” then the sanity check will be enabled.

- For each FPGA that the user wants to configure, there should be exactly one entry in the `main.txt` file with the following format:

  ```
  FPGA F: example.rbf
  ```

  In the above format, the “F” following FPGA is to signal that this entry is for FPGA F, and FPGA F would then be configured with the bit file `example.rbf`. The DN5000k10S has one to five FPGAs, which are FPGA A, B, D, E and F. The example has only one FPGA, which is FPGA F. There can be any number of spaces between the “:” and the configuration file name, but they need to be on the same line.

- Comments are allowed with the following rules:
  1. All comments must start at the beginning of the line.
  2. All comments must begin with `//`
  3. If a comment spans multiple lines, then each line must start with `//`

  Commented lines will be ignored during configuration, and are only for the user’s purpose.

- The file `main.txt` is **NOT** case sensitive.

**IMPORTANT:** All configuration file names have a maximum length of eight (8) characters, with an additional three (3) for the extension. Do not name your configuration files with long file names. In addition, all file names should be located in the root directory of the Smart Media card—no subdirectories or folders are allowed. Since the `main.txt` file controls which file is used to configure the FPGA, the Smart Media card can contain other files.

**Example of `main.txt`:**
Given the above example file:

- Verbose level is set to 2
- A sanity check on the bit files will be performed
- FPGA F will be configured with file `fpgaF.rbf`.

### Starting Fast Passive Parallel Configuration

If using the reference design SmartMedia card that came with the DN5000k10S then no files need to be copied to the card. Otherwise, copy your RBF file and `main.txt` to the root directory of the SmartMedia card using the FlashPath floppy adapter. Make sure the jumpers on JP1 are set for Fast Passive Parallel as shown in Table 2-4.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>off</td>
<td>off</td>
<td>off</td>
<td>Fast Passive Parallel</td>
</tr>
<tr>
<td>off</td>
<td>on</td>
<td>off</td>
<td>Passive Serial</td>
</tr>
</tbody>
</table>

Set up the serial port connection as described above in “Setting up the Serial Port (P2 — RS232 Port)” on page 2-18. Next, place the SmartMedia card in the SmartMedia socket on the DN5000k10S and turn on the power (NOTE: the card can only go in one way). The SmartMedia card is hot-swappable and can be taken out or put into the socket even when the power is on. Once the power has been turned on, the configuration process will begin as long as there is a valid SmartMedia card inserted properly in the socket. If there is not a valid SmartMedia card in the socket, then UP_LED[3:0] will flash (see Figure 8-2 on page 8-3 for LED descriptions) and the Main Menu will appear from the serial port. A SmartMedia card is determined to be invalid if either the format of the card does not follow the SSFDC specifications, or if it does not contain a file named `main.txt` in the root directory. If the configuration was successful, a message stating so will appear and the **Main Menu** will come up. Otherwise, an error message will appear.

The LEDs on DS1 and DS2 give feedback during and after the configuration process; see “LEDs” on page 8-3 for further details.

After the FPGA has been configured, the following **Main Menu** will appear on the serial port:

1. Configure FPGA(s) using `main.txt`
2. Interactive FPGA configuration menu
3. Check Configuration status

4. Select file to use in place of main.txt

5. List files on SmartMedia

6. Select FPGA to program via JTAG

Description of Main Menu Options.

1. Configure FPGAs Using "main.txt" as the Configuration File — By selecting this option, the FPGA will configure in Fast Passive Parallel mode. You can also press the reset button (S1) to reconfigure the FPGA in Fast Passive Parallel mode.

2. Interactive FPGA configuration menu — This option takes you to a menu titled "Interactive Configuration Menu" and allows the FPGA to be configured through a set of menu options instead of using the main.txt file. The menu options are described below.

   Description of Interactive Configuration Menu options:
   
   1. Select a bit file to configure FPGA(s) — This menu option allows the user to select a file from a list of files found on the SmartMedia card to use to configure the FPGA.

   2. Set verbose level (current level = 2) — This menu option allows the user to change the verbose level from the current setting. Please note: if the user goes back to the main menu and configures the FPGA(s) using main.txt, the verbose level will be set to whatever setting is specified in main.txt.

   3. Disable/Enable sanity check for bit files — This menu option either allows the user to disable or enable the sanity check, depending on what the current setting is. Please note: if the user goes back to the main menu and configures the FPGA(s) using main.txt, the sanity check will be set to whatever setting is specified in main.txt.

   M) Main menu — This menu option takes the user back to the Main Menu described above.

3. Check Configuration status — This option checks the status of the DONE pin and prints out whether or not the FPGA(s) have been configured along with the file name that was used for configuration.

4. Select file to use in place of main.txt — By default, the processor uses the file main.txt to get the names of the files to be used for configuration as well as options for the configuration process. However, a user can put several files that follow the format for main.txt on the SmartMedia card that contain different options for the configuration process. By selecting the main menu option 4, the user can select a .txt file from a list of files that should be used in place of main.txt. After selecting a new file to use in place of main.txt, the user should select Main Menu option 1 to configure the FPGA(s) according to this new file. If the power is turned off or the reset button (S1) is pressed, the configuration file is changed back to the default, main.txt.
5. **List files on SmartMedia** — This option prints out a list of all the files found on the SmartMedia card.

6. **Select FPGA to Program with JTAG** — This option must be set to enable an FPGA before it can be programmed through JTAG.

## SmartMedia

The configuration file for the FPGA is copied to a SmartMedia card using the SmartDisk FlashPath Floppy Disk Adapter. The approximate file size for each possible Stratix FPGA is shown below in Table 2-5. Note that several files can be put on a 32-megabyte card. We supply two 32-megabyte SmartMedia cards with the DN5000k10S. SmartMedia is a standard, so you can get more SmartMedia cards if you want. The DN5000k10S requires a +3.3 V card. Card sizes of 16, 32, 64, and 128 megabytes have been tested on the DN5000k10S. We have not seen 256 MB or larger cards for sale yet, but when we do there will probably be an update to the CPLD and processor on our website to support them.

**Table 2-5  Stratix FPGA Approximate File Sizes**

<table>
<thead>
<tr>
<th>Stratix FPGA</th>
<th>Number of Configuration Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOF for EP1S80</td>
<td>2,954,672</td>
</tr>
<tr>
<td>RBF for EP1S80</td>
<td>2,992,071</td>
</tr>
</tbody>
</table>

We get our SmartMedia cards from [http://www.computers4sure.com/](http://www.computers4sure.com/). A Delkin Devices 16-megabyte card (part number DDSMFLS2-16) sells for about $15. A 32-megabyte card (part number DDSMFLS2-32) will set you back about $20 (see Figure 2-12). New SmartMedia cards do not require formatting before use.

**NOTE:** SmartMedia cards do not need to be formatted before they are used. The Windows format command does not work—it is necessary to use the FlashPath utility to format a SmartMedia card.

Do not press down on the top of the SmartMedia Connector J1 if a SmartMedia card is not installed. The metal case shorts to the +3.3 V power supply and the case gets hot enough to burn your finger. We suggest that you leave a SmartMedia card in the connector to prevent this from occurring. A polyswitch fuse (F1) has been added so that the PWB and the SmartMedia connector are protected if you do accidently press on the top of the connector.

**NOTE:** Do NOT press on the SmartMedia Connector P58 if a card is not installed!
Synthesis and Emulation Issues

The QuartusII™ software from Altera is able to synthesize directly from Verilog or VHDL code. However, third-party synthesis tools provide an advantage: to create a memory block or multiplier, all you need to do is describe them functionally, and the tool will infer the appropriate DSP and RAM megafunctions for Quartus to place and route. On the other hand, if you are using Quartus to synthesize, and you try to infer an M-RAM block using a functional description, Quartus will attempt to route 200,000 LEs as a memory array. So, if you don’t have any other synthesis tool, you will need to become familiar with Quartus megafunctions.

We have tried the following tools for synthesis:

- Synplicity Synplify (http://www.synplicity.com/)
- Synopsys FPGA Express (http://www.synopsys.com/)
- Synopsys FPGA Compiler II

Of the four listed here, we find that Synplicity offers the best performance, followed by Exemplar. The Synopsys products are not the easiest products to use, and probably should be avoided until Synopsys decides that they

WARNING:

Do NOT format a SmartMedia card using the default Windows format program. All Smart Media cards come preformatted from the factory, and files can be deleted from the card when they are no longer needed. If for some reason you absolutely need to format a SmartMedia card, you must use the format program that is included in the FlashPath (SmartMedia floppy adapter) software.

Figure 2-12 Delkin 32 MB 3.3 V Smart Media Card
want to be in this market. It is generally not worth your time to preserve
your Synopsys ASIC compiler directives and scripts by using the FPGA
synthesis products from Synopsys. The time you save using Snopsys prod-
ucts is offset by other hassles.

**Synthesis Notes**

1. The FPGA used on your DN5000k10S is an EP1S80s in an F1508 pack-
age (EP1S60s available on request). Unless you paid for a faster speed
grade, the –6 is what you will be getting.

2. Assuming you have a synthesis tool other than QuartusII, memories
are best implemented by describing them behaviorally in your RTL. All
four synthesis products are sophisticated enough to map your behav-
ioral descriptions into the memory blocks. It is **NOT** necessary to
instantiate memories manually, unless you are synthesizing with
Quartus. Make sure, however, to check the report files to make sure
that your memories were implemented in memory blocks (if this is
possible). If input and output registers in your RTL don’t match the
behavior of the embedded memory blocks, the synthesis program
may not recognize what you intended, and give you arrays of LEs
instead.

3. Much to our surprise, the synthesis programs recognized RTL multi-
plier code and used the embedded multipliers without any trouble.
So, like the memories, RTL description of your multipliers is all that is
necessary unless you are synthesizing with Quartus. Make sure to
check the report files—multipliers that are implemented using logic
blocks (as opposed to the embedded memory blocks) take huge
amounts of FPGA resources.

4. Clocks are the biggest problem when converting ASIC code to FPGA
code. FPGAs only have a limited number of clock arrays. This is far too
complicated to describe here, so get the *Stratix Data Sheet* and read
about the clocks.
Chapter 3

PCI

Overview

The DN5000k10S can be hosted in a 32-bit, or 64-bit PCI slot. PCI-X is also supported. Stand-alone operation is described in “Stand-Alone Operation” on page 6-3. An EP1580-7, with care, should be able to support a 64-bit, 66 MHz PCI or PCI-X controller. We have not tested the PWB at PCI-X speeds of 100 MHz and 133 MHz. We suspect, but won’t guarantee, that the DN5000k10S can support these high frequencies, provided the speed grade of the FPGA is adequate. Figure 3-1 shows the FPGA pin connections for the PCI signals. This data is provided on the CD-ROM in a .csf file titled pins_F.csf, for your convenience.

The PCI/PCI-X edge connector is shown in Figure 3-2.

Stratix parts cannot tolerate +5 V TTL signaling, so the DN5000k10S must be plugged into a +3.3 V PCI slot. PCI-X, by definition, is +3.3 V signaling. The PWB is keyed so that it is not possible to mistakenly plug the board into a +5 V PCI slot. Do NOT grind out the key in the PCI host slot, and Do NOT modify the DN5000k10S to get it to fit into the slot. If you need a +3.3 V PCI slot, the DNPCIEXT-S3 Extender card can do this function. The link is http://www.dinigroup.com/products/pciextender.html. This extender also has the capability to slow the clock frequency of the PCI bus by a factor of two—a function that is very useful when prototyping ASICs.

Some Notes on the DN5000k10S and PCI/PCI-X

+3.3 V power is not needed on the host PCI connector. +3.3 V power is derived from +5 V using an on-board 10 A switching regulator. Power distribution for the DN5000k10S is described in “Power Supplies and Power Distribution” on page 6-1.

LOCK# has a pull-up. This is technically a violation of the PCI specification, but we have seen systems (from SUN!) that have the LOCK# pin floating. Remember that the function of this pin was deleted in the 2.2 version of

PCI Mechanical Specifications

The DN5000k10S is not a standard sized PCI card—it is too tall and slightly too long. This is sometimes an issue in servers that have a bracket installed over the top of the PCI cards. If you need to close the case on a DN5000k10S, some tower configurations may work. Figure 3-3 shows the exact dimensions of the DN5000k10S.

NOTE: +5 V Signaling on Stratix parts causes them to smoke! This is quite BAD! Do NOT Modify the DN5000k10S board to fit into your PCI slot.
<table>
<thead>
<tr>
<th>FPGA F (U11)</th>
<th>PCI Pin (P2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D24 → PCI_AD[0]</td>
<td>A58</td>
</tr>
<tr>
<td>B24 → PCI_AD[1]</td>
<td>B58</td>
</tr>
<tr>
<td>A24 → PCI_AD[3]</td>
<td>B56</td>
</tr>
<tr>
<td>D26 → PCI_AD[6]</td>
<td>A54</td>
</tr>
<tr>
<td>B26 → PCI_AD[7]</td>
<td>B54</td>
</tr>
<tr>
<td>C27 → PCI_AD[8]</td>
<td>A53</td>
</tr>
<tr>
<td>A27 → PCI_AD[9]</td>
<td>B52</td>
</tr>
<tr>
<td>A28 → PCI_AD[10]</td>
<td>B48</td>
</tr>
<tr>
<td>C29 → PCI_AD[12]</td>
<td>B47</td>
</tr>
<tr>
<td>D30 → PCI_AD[14]</td>
<td>B45</td>
</tr>
<tr>
<td>A30 → PCI_AD[15]</td>
<td>A44</td>
</tr>
<tr>
<td>A31 → PCI_AD[16]</td>
<td>A32</td>
</tr>
<tr>
<td>B31 → PCI_AD[17]</td>
<td>B32</td>
</tr>
<tr>
<td>C32 → PCI_AD[18]</td>
<td>A31</td>
</tr>
<tr>
<td>A32 → PCI_AD[19]</td>
<td>A29</td>
</tr>
<tr>
<td>B32 → PCI_AD[20]</td>
<td>B29</td>
</tr>
<tr>
<td>A33 → PCI_AD[21]</td>
<td>A28</td>
</tr>
<tr>
<td>B33 → PCI_AD[22]</td>
<td>B28</td>
</tr>
<tr>
<td>C34 → PCI_AD[23]</td>
<td>A27</td>
</tr>
<tr>
<td>A34 → PCI_AD[24]</td>
<td>A25</td>
</tr>
<tr>
<td>B34 → PCI_AD[25]</td>
<td>B25</td>
</tr>
<tr>
<td>A35 → PCI_AD[26]</td>
<td>A24</td>
</tr>
<tr>
<td>B35 → PCI_AD[27]</td>
<td>A23</td>
</tr>
<tr>
<td>C36 → PCI_AD[28]</td>
<td>A22</td>
</tr>
<tr>
<td>A36 → PCI_AD[29]</td>
<td>A21</td>
</tr>
<tr>
<td>B36 → PCI_AD[30]</td>
<td>A20</td>
</tr>
<tr>
<td>B37 → PCI_AD[31]</td>
<td>B20</td>
</tr>
<tr>
<td>C38 → C/BE[0]#</td>
<td>A52</td>
</tr>
<tr>
<td>A39 → C/BE[1]#</td>
<td>B44</td>
</tr>
<tr>
<td>A40 → C/BE[2]#</td>
<td>B33</td>
</tr>
<tr>
<td>B41 → C/BE[3]#</td>
<td>B26</td>
</tr>
<tr>
<td>D42 → FRAME#</td>
<td>A34</td>
</tr>
<tr>
<td>B43 → IRDY#</td>
<td>B35</td>
</tr>
<tr>
<td>C44 → PCI_CLK</td>
<td>B16</td>
</tr>
<tr>
<td>D45 → TRDY#</td>
<td>A36</td>
</tr>
<tr>
<td>B46 → DEVSEL#</td>
<td>B37</td>
</tr>
<tr>
<td>C47 → STOP#</td>
<td>A38</td>
</tr>
<tr>
<td>A48 → IDSEL</td>
<td>B26</td>
</tr>
<tr>
<td>B49 → REQ#</td>
<td>B18</td>
</tr>
<tr>
<td>C50 → REQ64#</td>
<td>A60</td>
</tr>
<tr>
<td>A51 → ACK64#</td>
<td>B60</td>
</tr>
<tr>
<td>B52 → PAR#</td>
<td>A67</td>
</tr>
<tr>
<td>C53 → PAR64</td>
<td>A6</td>
</tr>
<tr>
<td>A54 → ERR#</td>
<td>B40</td>
</tr>
<tr>
<td>B55 → SERR#</td>
<td>B42</td>
</tr>
<tr>
<td>C56 → LOCK#</td>
<td>A15</td>
</tr>
<tr>
<td>B57 → PCI_RST#</td>
<td>B39</td>
</tr>
<tr>
<td>D58 → GNT#</td>
<td>A17</td>
</tr>
<tr>
<td>A59 → PCI_AD[32]</td>
<td>A91</td>
</tr>
<tr>
<td>B60 → PCI_AD[33]</td>
<td>B90</td>
</tr>
<tr>
<td>C61 → PCI_AD[34]</td>
<td>A89</td>
</tr>
<tr>
<td>B62 → PCI_AD[35]</td>
<td>B89</td>
</tr>
<tr>
<td>C63 → PCI_AD[36]</td>
<td>A88</td>
</tr>
<tr>
<td>B64 → PCI_AD[37]</td>
<td>B87</td>
</tr>
<tr>
<td>D65 → PCI_AD[38]</td>
<td>B86</td>
</tr>
<tr>
<td>A66 → PCI_AD[39]</td>
<td>A85</td>
</tr>
<tr>
<td>B67 → PCI_AD[40]</td>
<td>B84</td>
</tr>
<tr>
<td>C68 → PCI_AD[41]</td>
<td>A83</td>
</tr>
<tr>
<td>B69 → PCI_AD[42]</td>
<td>B83</td>
</tr>
<tr>
<td>D70 → PCI_AD[43]</td>
<td>A82</td>
</tr>
<tr>
<td>A71 → PCI_AD[44]</td>
<td>B81</td>
</tr>
<tr>
<td>B72 → PCI_AD[45]</td>
<td>B80</td>
</tr>
<tr>
<td>C73 → PCI_AD[46]</td>
<td>A79</td>
</tr>
<tr>
<td>B74 → PCI_AD[47]</td>
<td>B78</td>
</tr>
<tr>
<td>D75 → PCI_AD[48]</td>
<td>A77</td>
</tr>
<tr>
<td>A76 → PCI_AD[49]</td>
<td>B77</td>
</tr>
<tr>
<td>B77 → PCI_AD[50]</td>
<td>A76</td>
</tr>
<tr>
<td>C78 → PCI_AD[51]</td>
<td>B75</td>
</tr>
<tr>
<td>B79 → PCI_AD[52]</td>
<td>A74</td>
</tr>
<tr>
<td>D80 → PCI_AD[53]</td>
<td>B74</td>
</tr>
<tr>
<td>A81 → PCI_AD[54]</td>
<td>A73</td>
</tr>
<tr>
<td>B82 → PCI_AD[55]</td>
<td>B72</td>
</tr>
<tr>
<td>C83 → PCI_AD[56]</td>
<td>A71</td>
</tr>
<tr>
<td>B84 → PCI_AD[57]</td>
<td>B71</td>
</tr>
<tr>
<td>D85 → PCI_AD[58]</td>
<td>A70</td>
</tr>
<tr>
<td>A86 → PCI_AD[59]</td>
<td>B69</td>
</tr>
<tr>
<td>B87 → PCI_AD[60]</td>
<td>A68</td>
</tr>
<tr>
<td>C88 → PCI_AD[61]</td>
<td>B68</td>
</tr>
<tr>
<td>B89 → PCI_AD[62]</td>
<td>A66</td>
</tr>
<tr>
<td>D90 → PCI_AD[63]</td>
<td>A65</td>
</tr>
<tr>
<td>A91 → C/BE[4]#</td>
<td>B64</td>
</tr>
<tr>
<td>B92 → C/BE[5]#</td>
<td>B63</td>
</tr>
<tr>
<td>C93 → C/BE[6]#</td>
<td>B62</td>
</tr>
<tr>
<td>B94 → C/BE[7]#</td>
<td>B61</td>
</tr>
</tbody>
</table>

**Figure 3-1** FPGA Pin Connections for PCI Signals
Figure 3-2  PCI/PCI-X Edge Connector
The PCI Specification. The pull-up is 1M, which should not adversely impact PCI functionality in any way.

The PCI JTAG signals TDI, TDO, TCK, TMS, TRST#, are not used. TDI and TDO are connected together per the PCI Specification to maintain JTAG chain integrity on the motherboard. The signals TMS, TCK, and TRST# are left unconnected.

The FPGA is volatile, meaning it loses its brains when power is off. The SmartMedia method takes about 1 second to configure an EP1580 after power is stable. It is likely that FPGA F will finish the configuration process before RST# is deasserted. If your system has an unusually fast RST#, it is possible that the FPGA will not be configured when RST# deasserts. A RST# that deasserts before the FPGA has finished cannot properly configure the PCI/PCI-X mode latch.

The signal 3.3Vaux is not connected.

The signals INTB#, INTC#, and INTD# are not connected.
JP2: Present Signals for PCI/PCI-X

The present signals indicate to the system board whether an add-in card is physically present in the slot and, if one is present, the total power requirements of the add-in card.

The JP2 PCI-X Present Header is shown in Figure 3-4.

![JP2 PCI-X Present Header](image)

Table 3-1 shows the Present Signal Definitions for PCI/PCI-X.

Table 3-1 Present Signal Definitions

<table>
<thead>
<tr>
<th>PRSNT1#</th>
<th>PRSNT2#</th>
<th>Expansion Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>Open</td>
<td>No expansion board present</td>
</tr>
<tr>
<td>Ground</td>
<td>Open</td>
<td>Expansion board present, 25W maximum</td>
</tr>
<tr>
<td>Open</td>
<td>Ground</td>
<td>Expansion board present, 15W maximum</td>
</tr>
<tr>
<td>Ground</td>
<td>Ground</td>
<td>Expansion board present, 7.5W maximum.</td>
</tr>
</tbody>
</table>

We have never seen the present signals used anywhere, but we have heard of systems that will not PNP (Plug-and-Play) configure a PCI board if both the present pins are left open. We recommend installing a jumper in location 1-2 (for PRSNT1-), or 3–4(for PRSNT2-), or both.

JP3: M66EN—66MHz Enable

The 66MHz_ENABLE pin (M66EN) indicates to the host whether the device can operate at 66 MHz or 33 MHz. Section 7.5.1 in the PCI Specification 2.2 provides the gory details. For 33 MHz only FPGA designs, install a jumper between pins 9 and 10 of JP3. For 66 MHz capable designs, install a jumper between pins 7 and 8 instead. Table 3-2 shows the jumper descriptions for M66EN.

Table 3-2 M66EN Jumper Descriptions

<table>
<thead>
<tr>
<th>Jumper JP3</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M66EN</td>
<td>Pins 9-10</td>
</tr>
<tr>
<td></td>
<td>Pins 7-8</td>
</tr>
</tbody>
</table>
TP13: PME–, Power Management Enable

This board does not have built-in support for PME– (power management enable). Connecting PME– to an FPGA that is not powered is a bad idea—the system powers up as the board is installed. PME– is connected to TP13. This test pin allows the user to connect external circuitry to PME– if this functionality is desired.

JP3: PCI/PCI-X Capability

Figure 3-5 shows the PCI-X/M66EN Capabilities Header. Add in PCI-X boards tell the system what speed they are capable of running by the correct setting of this header.

![Figure 3-5 PCI-X/M66EN Capability Header](image)

Add-in cards indicate at which frequency they support PCI-X, using a pin called PCIXCAP. If the card's maximum frequency is 133 MHz, this pin is left unconnected (except for a decoupling capacitor C206). If the card's maximum frequency is 66 MHz, it connects PCIXCAP to ground through a resistor R93 (and decoupling capacitor C206). Conventional PCI cards connect this pin to ground.

**JP3—PCIXCAP**

For PCI only (not PCI-X capable), jumper between pins 5 and 6.
For PCI-X 133 MHz capable, jumper between pins 3 and 4.
For PCI-X 66 MHz capable, jumper between pins 1 and 2 and pins 3 and 4.
The PCIXCAP jumpers are detailed in Table 3-3.

**Table 3-3 PCIXCAP Jumpers**

<table>
<thead>
<tr>
<th>PCIXCAP</th>
<th>Jumper(s) Installed</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCI Only</td>
<td>5–6</td>
</tr>
<tr>
<td>PCI-X 133 MHz</td>
<td>3–4</td>
</tr>
<tr>
<td>PCI-X 66 MHz</td>
<td>1-2, 3–4</td>
</tr>
</tbody>
</table>
The M66EN and PCIXCAP Encodings are shown in Table 3-4.

**Table 3-4  M66EN and PCIXCAP Encoding**

<table>
<thead>
<tr>
<th>M66EN</th>
<th>PCIXCAP</th>
<th>Conventional Device Frequency Capability</th>
<th>PCI-X Device Frequency Capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground</td>
<td>Ground</td>
<td>33 MHz</td>
<td>Not Capable</td>
</tr>
<tr>
<td>Not Connected</td>
<td>Ground</td>
<td>66 MHz</td>
<td>Not Capable</td>
</tr>
<tr>
<td>Ground</td>
<td>Pull-down</td>
<td>33 MHz</td>
<td>PCI-X 66 MHz</td>
</tr>
<tr>
<td>Not Connected</td>
<td>Pull-down</td>
<td>66 MHz</td>
<td>PCI-X 66 MHz</td>
</tr>
<tr>
<td>Ground</td>
<td>Not Connected</td>
<td>33 MHz</td>
<td>PCI-X 133 MHz</td>
</tr>
<tr>
<td>Not Connected</td>
<td>Not Connected</td>
<td>66 MHz</td>
<td>PCI-X 133 MHz</td>
</tr>
</tbody>
</table>
Chapter 4

Clocks and Clock Distribution

Functional Overview

The DN5000k10S ASIC emulation board has a flexible and configurable clock scheme. Figure 4-1 is a block diagram showing the clocking resources and connections.

The clocking structures for the DN5000k10S include the following features:

- 2 user-selectable socketed oscillators (X2, X3)
- 1 48 MHz oscillator (X1)
- 2 CY7B993 (or CY7B994) RoboclockII™ Multi-Phase PLL Clock Buffers
- 2 FCT3807 Low-Skew Clock Buffers

![Figure 4-1 Clock Distribution Block Diagram](image-url)
The Clock Grid, JP6: a 5X3 0.1 in. header distributes clock signals to two FCT3807 clock buffers and two RoboclockII™ PLL clock buffers (CY7B993 or CY7B994). The clock outputs from the buffers are dispersed throughout the board.

Two 3.3 V half-can oscillator sockets (X2 and X3) and the signal CLKOUT from the CPLD provide on-board input clock solutions. The DN5000k10S is shipped with both a 14.318 MHz (X2) and a 33 MHz (X3) oscillator. Neither X2 nor X3 are used by the configuration circuitry, so the user is free to stuff any standard 3.3 V half-can oscillator in the X2 and X3 positions (more detail later in “Customizing the Oscillators” on page 4-12). The Clock Grid can also accept a 5X2 ribbon cable. This cable can provide input clocks to both of the RoboclockII’s and one of the 3807 buffers.

The FCT3807 clock buffer provides a high speed 1-to-10 buffer with low skew (0.35 ns) allowing clocks A (ACLK[9:0]) and B (BCLK[9:0]) to be distributed point-to-point. The two RoboclockII PLL clock buffers (U14 and U15) offer functional control of clock frequency and skew, among other things. They are configured via header arrays (J8, JP10, JP9 and JP11). The DN5000k10S comes from the factory stuffed with CY7B994V, which can operate at frequencies from 24 MHz to 200 MHz. They can also be stuffed with CY7B993V which operate from 12 MHz to 100 MHz. (Note: Output frequency can be as low as 1 MHz, depending on the operating frequency—see below for details.) Each chip has 16 output clocks along with 2 feedback output clocks. Two sets of eight output clocks are jumper selectable for each chip. The feedback clocks are controlled separately.

The PLL clock buffers can accept either 3.3 V LVTTL or LV Differential (LVPECL) reference inputs. The devices can operate at up to 12x the input frequency while the output clocks can be divided up to 12x the operating frequency. Phase adjustments can be made in 625 ps or 1300 ps steps up to ±10.4 ns. All adjustments are jumper selectable.

**Clock Grid**

**Orientation and Description**

The clock grid, JP6, gives the user the ability to customize the clock scheme on the DN5000k10S. A brief description of each pin is given in Table 4-1. The physical orientation of the pins is diagrammed in Figure 4-2.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLKOUT</td>
<td>Clock signal from CPLD. Typically 12 MHz.</td>
</tr>
<tr>
<td>PLL1A</td>
<td>Input to RoboclockII #1</td>
</tr>
<tr>
<td>CLOCKA</td>
<td>Clock signal of oscillator #1 (X1)</td>
</tr>
<tr>
<td>BUFINB</td>
<td>Clock input to 3807 #2</td>
</tr>
<tr>
<td>CLOCKB</td>
<td>Clock signal of oscillator #2 (X2)</td>
</tr>
</tbody>
</table>
Clocks and Clock Distribution

Three main configurations are the most common.

First, the grid may be jumpered as follows:

**Configuration #1:**  
CLKOUT $\leftrightarrow$ PLL1A, CLOCKA $\leftrightarrow$ BUFINA and CLOCKB $\leftrightarrow$ BUFINB

Both 3807s receive their inputs from the oscillators. RoboclockII #1 receives a clock input from the CPLD. Also, RoboclockII #2 can use DCLK[7] from RoboclockII #1 as an input. This is explained in “Roboclock PLL Clock Buffers” on page 4-5. Second, the input clock distribution can be configured as:

**Configuration #2:**  
CLKOUT $\leftrightarrow$ PLL2BN_PRE, CLOCKA $\leftrightarrow$ PLL1A and CLOCKB $\leftrightarrow$ BUFINB

<table>
<thead>
<tr>
<th>Signal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLL2B_PRE</td>
<td>Secondary clock input to RoboclockII #2. Differential pair with PLL2BN_PRE.</td>
</tr>
<tr>
<td>PLL2BN_PRE</td>
<td>Secondary clock input to RoboclockII #2. Differential pair with PLL2B_PRE.</td>
</tr>
<tr>
<td>BUFINA</td>
<td>Clock input to 3807 #1</td>
</tr>
<tr>
<td>PLL1BN_PRE</td>
<td>Secondary clock input to RoboclockII #1. Differential pair with PLL1B_PRE.</td>
</tr>
<tr>
<td>PLL1B_PRE</td>
<td>Secondary clock input to RoboclockII #1. Differential pair with PLL1BN_PRE.</td>
</tr>
<tr>
<td>GND</td>
<td>Ground signals to provide signal integrity for ribbon cables.</td>
</tr>
</tbody>
</table>
In this configuration, a 3807 #2 receives an oscillator input. RoboclockII #1 receives an oscillator input while RoboclockII #2 receives the CPLD output clock signal. 3807 #1 is unused.

Finally, the grid may be configured as:

**Configuration #3**: \( \text{CLKOUT} \leftrightarrow \text{BUFINB}, \text{CLOCKA} \leftrightarrow \text{BUFINA}, \) and \( \text{CLOCKB} \leftrightarrow \text{PLL1BN_PRE} \)

The 3807 #1 receives an oscillator input, and the 3807 #2 receives a CPLD input. Meanwhile, RoboclockII #1 receives the other oscillator input. The user can wire-wrap a clock to the unused driver(s) as needed. This enables full use of the timing devices on the DN5000k10S.

Also, the destination of the output clocks might dictate some other configuration. This manual and other documentation should provide more than enough information to satisfy the user’s needs (See Figure 4-3).

**Figure 4-3** PECL Clock Input and Termination

![Figure 4-3 PECL Clock Input and Termination](image)

**NOTE**: C380, C381, C382 and C383 are stuffed with 0-ohm resistors!

Note that the schematic shows capacitors in positions C380, C381, C382 and C383. The DN5000k10S has 0-ohm resistors in these capacitor positions. The termination resistors R206-R211, R207–R212, R209–R198 and R214–R210 are not stuffed.

**Ribbon Cable: Providing an Off-Board Clock to the DN5000k10S**

The DN5000k10S gives the user a simple means to bring off-board clocks onto the board. The user can attach 10-pin ribbon cable to rows B and C of the Clock Grid. JP6-B consists of an input to 3807 #1 and differential pair inputs to both RoboclockII’s. JP6-C consists of ground pins for signal integrity. These signals are described in Table 4-1 on page 4-2. BUFINA is a standard 3.3 V TTL input.
Both differential pairs provide some flexibility. The user can provide a single 3.3 V TTL input. It can be attached to either input. However, the other input must be left open. The user can provide a differential clock input to the pair. The differential clock inputs must obey the electrical specifications listed in Table 4-8 on page 4-11.

While attaching a ribbon cable, the user can jumper oscillator signal **CLOCKB** to **BUFINB** (3807 #2) on P54. This results in full use of all of the timing devices on the DN5000k10S (See Figure 4-4).

**Roboclock PLL Clock Buffers**

Figure 4-5 is a functional diagram of Roboclock 1 and Roboclock 2.

**Jumper Descriptions**

Headers J8, JP10, JP9 and JP11 are used to control the PLLs. Each header consists of GND pins in row A, various PLL inputs in row B, and +3.3 V pins in row C. The layout of the headers is shown in Figure 4-6.
Figure 4-5   Functional Diagram of Roboclock 1 and Roboclock 2
The input pins are either LVTTL or 3-level input pins. The LVTTL pins need to be jumpered HIGH or LOW, which is achieved by connecting the input pin to the neighboring +3.3 V or GND pin, using a jumper. The 3-level input pins can be in a HIGH, MID, or LOW state. The HIGH and LOW states are achieved in the same way as the LVTTL pins. The MID state is reached by leaving the input pin unjumpered. The RoboclockII’s have internal circuitry to bring the pin to 1.5 V when left open. The Jumper Definitions are shown in Table 4-3.
### Table 4-3  Jumper Definitions

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLLSEL2[1]</td>
<td>LVTTL</td>
<td>LOW</td>
<td>Input Clock Select: If LOW, U15:DCLK[7] or FCLKOUT (U14:PLL1A or HDR_CLKOUT) is selected as the input clock. If HIGH, the U15:PLL2BN (U14:PLL1BN) pair is selected as the input clock.</td>
</tr>
<tr>
<td>MODE[2:1]</td>
<td>3-Level</td>
<td>HIGH</td>
<td>Output Mode: If HIGH, clock outputs disable to high-Z state. If LOW, clock outputs disable to “HOLD-OFF” mode. If MID, clock outputs disable to factory test mode.</td>
</tr>
<tr>
<td>INV2[1]</td>
<td>3-Level</td>
<td>MID</td>
<td>Invert Mode: When HIGH, clocks CCLK[3:0] (ECLK[3:0]) are inverted. When MID, these clock outputs are non-inverting. When LOW, the pairs CCLK[1:0] and CCLK[3:2] (ECLK[1:0] and ECLK[3:2]) will be complementary.</td>
</tr>
<tr>
<td>FBDIS[2:1]</td>
<td>LVTTL</td>
<td>LOW</td>
<td>Feedback Disable: When HIGH, feedback is disabled. When LOW, feedback is enabled.</td>
</tr>
<tr>
<td>RB[C-F]F[1:0]</td>
<td>3-Level</td>
<td>MID</td>
<td>Output Phase Function: Each pair controls the phase function of the respective group of outputs. See “Clock Skew” on page 4-10 for more information.</td>
</tr>
<tr>
<td>[C-F]DS[1:0]</td>
<td>3-Level</td>
<td>LOW</td>
<td>Output Divider Function: Each pair controls the divider function of the respective group of outputs. See “Clock Division” on page 4-9 for more information.</td>
</tr>
<tr>
<td>FS[2:1]</td>
<td>3-Level</td>
<td>LOW</td>
<td>Frequency Select: The input specifies the operating range of the nominal frequency (f_NOM). See “General Control” on page 4-9 for more information.</td>
</tr>
<tr>
<td>FBF0[2:1]</td>
<td>3-Level</td>
<td>LOW</td>
<td>Feedback Output Phase Function: The input controls the phase function of the feedback outputs. See “Feedback and Clock Multiplication” on page 4-9 for more information.</td>
</tr>
<tr>
<td>FBDS[1:0][2:1]</td>
<td>3-Level</td>
<td>MID</td>
<td>Feedback Output Divider Function: Each pair controls the divider function of the feedback outputs. See “Feedback and Clock Multiplication” on page 4-9 for more information.</td>
</tr>
</tbody>
</table>
**General Control**

$FS[2:1]$ is a 3-Level input which determines the allowable range for the operating frequency $f_{NOM}$ of the device. Depending on the chip grade, the PLL can operate between 12–100 MHz or 24–200 MHz. The actual $f_{NOM}$ frequency can be determined by setting all jumpers to their defaults. Thus, $f_{NOM}$ will be seen on all of the “divide-by-one” clock outputs. The user can set $FS$ accordingly. The Frequency Range Settings are shown in Table 4-4.

**Table 4-4  Frequency Range Settings**

<table>
<thead>
<tr>
<th>FS[2:1]</th>
<th>CY7B993V</th>
<th>CY7B994V</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f_{NOM}$ (MHz)</td>
<td>$f_{NOM}$ (MHz)</td>
</tr>
<tr>
<td></td>
<td>MIN</td>
<td>MAX</td>
</tr>
<tr>
<td>LOW</td>
<td>12</td>
<td>26</td>
</tr>
<tr>
<td>MID</td>
<td>24</td>
<td>52</td>
</tr>
<tr>
<td>HIGH</td>
<td>48</td>
<td>100</td>
</tr>
</tbody>
</table>

**Feedback and Clock Multiplication**

First of all, $FBDIS[2:1]$ must be set LOW, enabling feedback. The feedback output is looped back to the feedback input. When a divided output is applied to the feedback input, the VCO (voltage controlled oscillator) of the PLL aligns the feedback input with the original input clock. Thus, with a 10 MHz input clock and the feedback outputs set to divide by 2, $f_{NOM}$ must be 20 MHz. Consequently, 10 MHz is seen on the feedback output clocks and can be aligned with the input clocks. The feedback clock divider function actually serves as a clock multiplication mechanism for the operating frequency $f_{NOM}$. The divider function and the clock skew function are set in the same manner for the feedback and the normal clock outputs. See “Clock Division” on page 4-9 and “Clock Skew” on page 4-10, respectively.

**Clock Division**

The three pairs of DS inputs per chip are used to control the two groups of clock outputs and the feedback outputs of each PLL. The user can simply follow the Divider Function Table to acquire the desired output frequency. There are two things to remember. First, $FS[2:1]$ must be set properly according to $f_{NOM}$. Second, the $FBDS$ feedback inputs act as operating clock frequency multipliers. The Output Divider Settings are shown in Table 4-5.
Clocks and Clock Distribution

Table 4-5  Output Divider Settings

<table>
<thead>
<tr>
<th>Input Signals</th>
<th>Output Divider Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>[C-fF]DS1 and [C-fF]DS0 and FBDS1[2:1]</td>
<td>[C-fF]DS0 and FBDS0[2:1]</td>
</tr>
<tr>
<td>LOW</td>
<td>LOW</td>
</tr>
<tr>
<td>LOW</td>
<td>MID</td>
</tr>
<tr>
<td>LOW</td>
<td>HIGH</td>
</tr>
<tr>
<td>MID</td>
<td>LOW</td>
</tr>
<tr>
<td>MID</td>
<td>MID</td>
</tr>
<tr>
<td>MID</td>
<td>HIGH</td>
</tr>
<tr>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>HIGH</td>
<td>MID</td>
</tr>
<tr>
<td>HIGH</td>
<td>HIGH</td>
</tr>
</tbody>
</table>

Table 4-6  Time Unit N-factor

<table>
<thead>
<tr>
<th>FS</th>
<th>CY7B993V</th>
<th>CY7B994V</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N</td>
<td>f_NOM (MHz) at which t_U = 1 ns</td>
</tr>
<tr>
<td>LOW</td>
<td>64</td>
<td>15.625</td>
</tr>
<tr>
<td>MID</td>
<td>32</td>
<td>31.25</td>
</tr>
<tr>
<td>HIGH</td>
<td>16</td>
<td>62.5</td>
</tr>
</tbody>
</table>

**Clock Skew**

Clock skew is controlled by the “F” inputs. The clock skew may be any integer value from 0 to ±8 times the RoboclockII time unit t_U. The time unit value is derived from the operating frequency f_NOM and the FS[2:1] setting. The following equation yields the time unit t_U.

\[ t_U = \frac{1}{f_{\text{NOM}} \cdot N} \]

The possible values for N are given in Table 4-6. The available skew for each RoboclockII derived clock is given in Table 4-7. Based on the following information, the user will be able to adjust the skew for any of the RoboclockII outputs.
Differential Clocks

In addition to LVTTL clock signals, the RoboclockII clock buffers can handle LV Differential (LVPECL) clocks. The user can cable in an acceptable differential signal to PLL1B and PLL1BN, or PLL2B and PLL2BN through the clock grid JP6. The signals must obey the specifications given in Table 4-8. Onboard circuitry is available to center the signals about the proper voltage, if needed.

Table 4-8 LVPECL Input Specifications

<table>
<thead>
<tr>
<th>Description</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Differential Voltage</td>
<td>0.4</td>
<td>3.3</td>
</tr>
<tr>
<td>Highest HIGH Voltage</td>
<td>1.0</td>
<td>3.3</td>
</tr>
<tr>
<td>Lowest LOW Voltage</td>
<td>GND</td>
<td>2.9</td>
</tr>
<tr>
<td>Common Mode range (crossing voltage)</td>
<td>0.8</td>
<td>3.3</td>
</tr>
</tbody>
</table>

The clock input of the RoboclockII can accept a superset of PECL. PECL involves a 1 V swing about Vcc/2. The RoboclockII clock input can accept a swing of up to 3.3 V about Vcc/2, which gives the user another dimension of flexibility.
The CY7B993V/4V can output LVTTL complementary (differential) signals, too. Setting \texttt{INV1 (INV2)} LOW will result in clocks \texttt{CCLK[1:0]} and \texttt{CCLK[3:2]} (\texttt{ECLK[1:0]} and \texttt{ECLK[3:2]}) becoming complementary pairs. A network of series and parallel resistors could be used to reduce the nominal swing of the clock signals.

**Useful Notes and Hints**

The CYB993V consistently outputs \(\sim 32.5\) MHz signals in cases of improper settings or unacceptable clock inputs. This was observed when:

- The CY7B993V part was operating at a nominal frequency \(f_{\text{nom}}\) of 36.4 MHz with \(\texttt{FS}\) set LOW.
- Identical clocks were sent to \texttt{PLL2B} and \texttt{PLL2BN}.

For the CY7B994V part, the operating frequency can reach up to 200 MHz. However, the maximum output frequency is 185 MHz. This means when \(185\) MHz \(\leq f_{\text{nom}} \leq 200\) MHz, the output divider must be set to at least 2. Otherwise, the RoboclockII’s will output garbage.

**Customizing the Oscillators**

The user can customize the frequency of the clock networks by stuffing oscillators in \texttt{X2} and \texttt{X3}. The DN5000k10S is shipped with a 14.318 MHz oscillator in location \texttt{X2} and a 100 MHz oscillator in \texttt{X2}. The RoboclockII’s are not \(+5\) V tolerant, so \(+3.3\) V oscillators are necessary.

We get our oscillators from Digi-Key (http://www.digikey.com/). Of note is an Epson line of oscillators called the \textbf{SG-8002 Programmable Oscillators}. Any frequency between 1.00 MHz–106.25 MHz can be procured in the normal Digi-Key shipping time of 24 hours. A half-can, \(+3.3\) V CMOS version is needed with a tolerance of 50 ppm. The part number for an acceptable oscillator from this family would be:

\begin{itemize}
  \item \textbf{SG-8002DC-PCB-ND}
  \item package \texttt{SG-531}
  \item output enable
  \item \(+3.3\) V CMOS
  \item 50 ppm
\end{itemize}

If the order is placed via the web page, the requested frequency to two decimal places is placed in the Web Order Notes. The datasheet is on the CD-ROM for this oscillator. The file name is SG8002DC.pdf.
Any polarity of output enable for each oscillator (on pin 1) is acceptable. Make sure that you have the proper jumper settings at positions 9 and 10 of JP9A, JP9B and JP9C. See Figure 4-7 and Table 4-9 for a description.

**Figure 4-7  Clock OE Pin Jumper Settings**

**Table 4-9  Clock OE Pin Jumper Settings**

<table>
<thead>
<tr>
<th>Clock OE</th>
<th>Jumper Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active High OE for X2</td>
<td>Jumper JP9.26 to JP9.27</td>
</tr>
<tr>
<td>Active Low OE for X2</td>
<td>Jumper JP9.26 to JP9.25</td>
</tr>
<tr>
<td>Active High OE for X3</td>
<td>Jumper JP9.29 to JP9.30</td>
</tr>
<tr>
<td>Active Low OE for X3</td>
<td>Jumper JP9.29 to JP9.28</td>
</tr>
</tbody>
</table>
DN5000k10S PCI_CLK Operation

The DN5000k10S ASIC emulation board has the ability to run the FPGA, all SSRAMs, SDRAM, and DDR SDRAM off of PCI_CLK. PCI_CLK is a single destination clock which is routed to FPGA F(U11) from the PCI connector. The user can input PCI_CLK to the Stratix Enhanced PLL5. The resulting PLL output can be sent to RoboclockII (U14) via the signal GCLKOUT. All of the memories on the DN5000k10S run off one of the ECLK clock outputs from RoboclockII.

**PCI_CLK Details**

PCI_CLK is connected to the Stratix Enhanced PLL5 input (pin B22). To run all memories off of PCI_CLK, a PLL must be instantiated in the FPGA code. The PLL will require a minimum of three connections (input clk, output clk, and external feedback input). For further information on Stratix PLL operation, see the Altera website at [http://www.altera.com](http://www.altera.com). The Stratix Datasheet (ds_stx.pdf) which can be found on the DN5000k10S CD-ROM, also provides useful information on PLLs.

The GCLKOUT signal is connected to one of the four input pins on RoboclockII. GCLKOUT’s complementary input is DCLK[7](R). FCLKOUT and DCLK[7](R) are both single ended TTL inputs. When either of them is being used, the other one must be left open. For complete PCI_CLK operation, jumper JP5 must not be stuffed leaving DCLK[7](R) open.

If set to the default configuration, RoboclockII will drive a one-to-one PCI_CLK derived clock on its outputs. See “Roboclock PLL Clock Buffers” on page 4-5 for more information. RoboclockII has 12 clock outputs (ECLK[12:0]). The FPGA (ECLK[9], ECLK[12]), SSRAMs (ECLK[7:4]), DDR SDRAM (ECLK[3:0]), and DDR SDRAM (PLL outputs – JP4 must have jumper connecting pins 9 & 10 to send ECLK[8] to DDR SDRAM) receive ECLK signals.

To complete this setup, a feedback signal must be connected to the PLL in FPGA. RoboclockII sends ECLK[12] to the feedback input of the FPGA. ECLK[12] needs to be connected to the “fbin” input signal of the PLL. Using ECLK[12] as feedback allows the PLL to properly synchronize the DN5000k10S PCI_CLK network which completes the setup (see Figure 4-8 for a diagram of the PCI_CLK PLL circuit).

The DN5000k10S can be run off any single-ended TTL clock signal which is sent to the Roboclocks. The ECK distribution provides the DN5000k10S this flexibility. PCI_CLK has special implications for the DN5000k10 PCI operation.
Clocks and Clock Distribution

GCLKOUT

GCLKOUT is assigned to a dedicated clock output pin in the Stratix architecture, and can be used to drive Roboclock 2 (see the section “DN5000k10S PCI_CLK Operation” on page 4-14 for details). The DN5000k10S differs from previous Dini Group emulator boards because the Stratix architecture assigns each PLL to specific clock pins. In the case of GCLKOUT, the only possible source is from PLL5, which has only one possible input, PCI_CLK. So, the sole purpose of GCLKOUT is to provide the means to run the whole board with PCI_CLK.

Header Clocks

Each of the two 200-pin header (P8 and P9) receives a clock signal from each of the five clock groups (ACLK, BCLK, CCLK, DCLK, and ECLK).

DCLK[7](R)

The signal DCLK[7](R) is routed from one of the Roboclock I outputs to one of the RoboclockII inputs. Jumper JP5 lies along this connection route. JP5 must be installed in order to utilize DCLK[7](R). DCLK[7](R) and GCLKOUT are complementary input on Roboclock 2, and are both single-ended TTL inputs. When either of them is being used, the other one must be left open. Thus, GCLKOUT must be undriven on FPGA F for DCLK[7](R) to operate.

DCLK[7](R) provides two useful results. First, any clock signal or some derivation sent to Roboclock 1 can be driven onto RoboclockII for full distribution.

Second, running a clock through Roboclock I to RoboclockII gives the user more divide and multiply options for the clock frequencies. Here is an example. If you have a 40MHz input clock, the user cannot output a 30MHz clock with a single RoboclockII’s multiply and divide options. However, the user can input a 40MHz to RoboclockII #1 and divide it by 4. By installing the JP5 jumper, a 10MHz clock will be driven onto RoboclockII #2. Setting RoboclockII #2’s feedback outputs to divide by 3, the operating frequency...
Clocks and Clock Distribution

will become 30MHz. Thus, a 30MHz could be driven onto the
RoboclockII #2 output signals.

NOTE: The signal GCLKOUT must be left open in order to utilize
DCLK[7] (R)
The DN5000k10S has six external memories: four 36-bit SSRAMs, one 72-bit SDRAM DIMM, and one 72-bit DDR SDRAM DIMM. The four SSRAMs are referred to as SSRAM 1 (U9), SSRAM 2 (U6), SSRAM 3 (U8), and SSRAM 4 (U13).

SSRAMs

The SSRAMs can be stuffed with ZBT, non-ZBT, pipeline, or flowthrough parts. We believe we have anticipated the additional address lines for the 1 M x 36 and 2 M x 36 parts when they are available. The DN5000k10S is stuffed at the factory with 512 K x 36-bit Synchronous Pipeline Burst SRAM. Samsung K7A163600M-QC1400 are probably the parts you will have stuffed into your DN5000k10S. The datasheet is on the CD-ROM in the file DS_K7A1636(18)00M.pdf. The SSRAMs are tested at 133 MHz.

SSRAM Notes

All SSRAMs use ECLK for their clock.

The signal connections for SSRAM 1 are shown in Figure 5-1.
The signal connections for SSRAM 2 are shown in Figure 5-2.
The signal connections for SSRAM 3 are shown in Figure 5-3.
The signal connections for SSRAM 4 are shown in Figure 5-4.

Flowthrough SSRAMs are functionally the closest to ASIC-style memories. Pipeline SSRAMs can be clocked at faster frequencies. ZBT SSRAMs are typically one generation behind in density. The subtle differences between the styles of memories are described in the next section.
Figure 5-2   SSRAM 2 (U10) Bus Signals
Figure 5-3  SSRAM 3 (U8) Bus Signals
## Figure 5-4  SSRAM 4 (U13) Bus Signals
Pin 14 of each SSRAM may be pulled high, pulled low, or left unconnected. Table 5-1 describes which 0-ohm resistors must be used for each type of SSRAM to function correctly.

**Table 5-1 Requirements for Non-Standard SSRAMs**

<table>
<thead>
<tr>
<th></th>
<th>FB</th>
<th>AD</th>
<th>AB</th>
<th>ED</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ZBT Pipeline</strong></td>
<td>Install R129 R2</td>
<td>Install R130 R3</td>
<td>Install R128 R1</td>
<td>Install R215 R70</td>
</tr>
<tr>
<td><strong>ZBT Flowthrough</strong></td>
<td>Install R129 R140</td>
<td>Install R130 R142</td>
<td>Install R128 R138</td>
<td>Install R215 R216</td>
</tr>
<tr>
<td><strong>Syncburst Flowthrough or Pipeline</strong></td>
<td>No Extra Resistors</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pipeline, Flowthrough, ZBT

Syncburst FT (Flowthrough) (Figure 5-5) is the most straightforward type of SSRAM available for the DN5000k10S. Write data may be accepted on the same clock cycle as the activation signal and address, and read data is returned one clock cycle after it is requested. Syncburst is designed to allow two controllers to access the same SSRAM, using two activation signals, ADSC# and ADSP#; an activation with ADSP# requires data and byte enables one clock cycle after the address and activation. Syncburst PL (Pipelined) (Figure 5-6) is identical except for registered outputs, which delay read data an additional clock cycle but may be necessary for high-speed designs.

Zero-Bus-Turnaround (ZBT) SSRAMs are designed to eliminate wait states between reads and writes by synchronizing data. Thus, ZBT FT SSRAMs...
(Figure 5-7) accept and return data one clock cycle after the address phase, and ZBT PL SSRAMs (Figure 5-8) accept and return data two clock cycles after the address phase. This allows the user to begin a write burst immediately after the last word of a read burst, because read data will be returned before the first write data is required. The timing is illustrated in Figure 5-9 and Table 5-2.

**Figure 5-7  Syncburst ZBT FT**

**Figure 5-8  Syncburst ZBT PL**

**Figure 5-9  Syncburst and ZBT SSRAM Timing**
The DN5000k10S has a socket for a +3.3 V 168-pin SDRAM DIMM. Either registered or unbuffered modules fit in the socket (J3). The same PC100/PC133 SDRAM modules that you put into your PC are used here. Your DN5000k10S will be stuffed and tested with a 1 Gbyte PC133 SDRAM DIMM, unless otherwise requested.

All DIMM pins are connected to the FPGA and the pins are shown in Figure 5-10 and Figure 5-11. We aren’t quite sure what the largest size SDRAM DIMM is that will work in the DN5000k10S, but here is the math as best we understand it:

14 Address lines $A_{[13:0]}$  
(multiplexed between $RAS^*$ and $CAS^*$, address 10 not used for $CAS^*$)  
2 bank address $BA_{[1:0]}$  
4 chip selects ($S_{[3:0]}^*$) used in pairs

So, we think that there are 29 address bits ($27 + 2$) and 2 possible chip selects, which add one more address bit. This totals 30 address bits — 1 G of 72-bit long words, which is 8 Gbytes. Please tell us if this math is wrong.

SDRAM modules require 4 clocks — $CK_{[3:0]}$. These clocks are driven by the RoboclockII 2 and the signal names are $ECLK$.

The CD-ROM has a datasheet of an acceptable 1 Gbyte SDRAM module from Micron. The file name is SDF36C64_127x72G_B.pdf.
<table>
<thead>
<tr>
<th>FPGA F (U11)</th>
<th>SDRAM (J3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M5</td>
<td>SDRAM_CKE[0]</td>
</tr>
<tr>
<td>K7</td>
<td>SDRAM_CKE[1]</td>
</tr>
<tr>
<td>W7</td>
<td>SDRAM_CSN[0]</td>
</tr>
<tr>
<td>U5</td>
<td>SDRAM_CSN[1]</td>
</tr>
<tr>
<td>R7</td>
<td>SDRAM_CSN[2]</td>
</tr>
<tr>
<td>L6</td>
<td>SDRAM_CSN[3]</td>
</tr>
<tr>
<td>N6</td>
<td>SDRAM_BA[0]</td>
</tr>
<tr>
<td>R8</td>
<td>SDRAM_BA[1]</td>
</tr>
<tr>
<td>AB8</td>
<td>SDRAM_WEn</td>
</tr>
<tr>
<td>V6</td>
<td>SDRAM_CASn</td>
</tr>
<tr>
<td>E5</td>
<td>SDRAM_RASn</td>
</tr>
<tr>
<td>V8</td>
<td>SDRAM_ADDR[0]</td>
</tr>
<tr>
<td>L3</td>
<td>SDRAM_ADDR[1]</td>
</tr>
<tr>
<td>V7</td>
<td>SDRAM_ADDR[2]</td>
</tr>
<tr>
<td>U8</td>
<td>SDRAM_ADDR[4]</td>
</tr>
<tr>
<td>T8</td>
<td>SDRAM_ADDR[8]</td>
</tr>
<tr>
<td>P5</td>
<td>SDRAM_ADDR[9]</td>
</tr>
<tr>
<td>T7</td>
<td>SDRAM_ADDR[10]</td>
</tr>
<tr>
<td>M6</td>
<td>SDRAM_ADDR[12]</td>
</tr>
<tr>
<td>AA8</td>
<td>SDRAM_DQBM[0]</td>
</tr>
<tr>
<td>Y7</td>
<td>SDRAM_DQBM[1]</td>
</tr>
<tr>
<td>P8</td>
<td>SDRAM_DQBM[2]</td>
</tr>
<tr>
<td>P7</td>
<td>SDRAM_DQBM[3]</td>
</tr>
<tr>
<td>V5</td>
<td>SDRAM_DQBM[4]</td>
</tr>
<tr>
<td>U6</td>
<td>SDRAM_DQBM[5]</td>
</tr>
<tr>
<td>L5</td>
<td>SDRAM_DQBM[6]</td>
</tr>
<tr>
<td>K6</td>
<td>SDRAM_DQBM[7]</td>
</tr>
<tr>
<td>M9</td>
<td>SDRAM_SCL</td>
</tr>
<tr>
<td>T9</td>
<td>SDRAM_SDA</td>
</tr>
<tr>
<td>J4</td>
<td>SDRAM_SA[0]</td>
</tr>
<tr>
<td>J3</td>
<td>SDRAM_SA[1]</td>
</tr>
<tr>
<td>AC8</td>
<td>SDRAM_CB[0]</td>
</tr>
<tr>
<td>AC7</td>
<td>SDRAM_CB[1]</td>
</tr>
<tr>
<td>N8</td>
<td>SDRAM_CB[2]</td>
</tr>
<tr>
<td>N7</td>
<td>SDRAM_CB[3]</td>
</tr>
<tr>
<td>AB6</td>
<td>SDRAM_CB[4]</td>
</tr>
<tr>
<td>AB5</td>
<td>SDRAM_CB[5]</td>
</tr>
<tr>
<td>AL4</td>
<td>SDRAM_CB[6]</td>
</tr>
<tr>
<td>AL4</td>
<td>SDRAM_CB[7]</td>
</tr>
<tr>
<td>AL8</td>
<td>SDRAM_DATA[0]</td>
</tr>
<tr>
<td>AL7</td>
<td>SDRAM_DATA[1]</td>
</tr>
<tr>
<td>AK8</td>
<td>SDRAM_DATA[2]</td>
</tr>
<tr>
<td>AK7</td>
<td>SDRAM_DATA[3]</td>
</tr>
<tr>
<td>AJ7</td>
<td>SDRAM_DATA[5]</td>
</tr>
<tr>
<td>AH8</td>
<td>SDRAM_DATA[6]</td>
</tr>
<tr>
<td>AH7</td>
<td>SDRAM_DATA[7]</td>
</tr>
<tr>
<td>AG8</td>
<td>SDRAM_DATA[8]</td>
</tr>
<tr>
<td>AG7</td>
<td>SDRAM_DATA[9]</td>
</tr>
<tr>
<td>AF8</td>
<td>SDRAM_DATA[10]</td>
</tr>
<tr>
<td>AE8</td>
<td>SDRAM_DATA[12]</td>
</tr>
<tr>
<td>AD8</td>
<td>SDRAM_DATA[14]</td>
</tr>
<tr>
<td>AD7</td>
<td>SDRAM_DATA[15]</td>
</tr>
<tr>
<td>MB8</td>
<td>SDRAM_DATA[16]</td>
</tr>
</tbody>
</table>

Figure 5-10  SDRAM (J19) Bus Signals (Page 1 of 2)
SDRAM On-Board Options

**R218** and **R217** are connected to the WP (Write Protect) input of the SDRAM EEPROM. Stuffing a 0-ohm resistor in **R217** will keep the WP signal high, whereas stuffing it in **R218** drives the signal low. The default configuration is **R217** stuffed. **NEVER** stuff both resistors at the same time.

The EEPROM holds data describing the size, configuration, and timing characteristics of the SDRAM. The data is write-protected when the WP signal is high. There should be little or no reason to want to overwrite the EEPROM data. Some SDRAM manufacturers simply connect the WP pin of the EEPROM chip to the power supply of the SDRAM, in which case the WP resistors have no effect whatsoever.

---

**Figure 5-11** SDRAM (J19) Bus Signals (Page 2 of 2)
Header **JP7** is connected to the **REGE** (Register Enable) input of the SDRAM and to ground. A pull-up resistor keeps the **REGE** signal high when the header is unconnected; adding a jumper between the two pins drives the signal low. The default configuration is no jumper. **REGE** is also connected to the FPGA, intended as an input so that the design can check the status of **REGE**. Do **NOT** drive this signal high when **JP7** is jumpered.

On some SDRAMs, the **REGE** input may be used to select Registered or Non-Registered behavior. If **REGE** is high, the control signals will go through registers before being sent to the individual DRAMs, delaying access by one clock cycle but improving fanout; if it is low, the signals will be passed directly to the DRAMs.

**DDR SDRAM**

The DN5000k10S has a socket for a 184-pin DDR SDRAM DIMM. Either a registered or unbuffered module fits in the socket (**J2**). The same PC266/PC2100 modules that you put into your PC are used here. Your DN5000k10S will be stuffed and tested with a 512 MB PC2100 DDR SDRAM DIMM unless otherwise specified.

All DIMM pins are connected to the FPGA and the pins are shown in Figure 5-10 on page 5-10 and Figure 5-11 on page 5-11. The largest DDR SDRAM that the DN5000k10S can be stuffed with is 1 GB x 72 (8 GB).

DDR SDRAM modules require three (3) differential clocks: **CK[2:0]** and **CK#[2:0]**. These clocks are driven by the FPGA’s enhanced PLL6 outputs and the signal names are **DDR_CLK[2:0]** and **DDR_CLKn[2:0]**. For further information on Stratix PLL operation, see the Altera website at [www.altera.com](http://www.altera.com). The Stratix Datasheet (**ds_stx.pdf**), which can be found on the DN5000k10S CD-ROM, also provides useful information on PLLs.

**DDR SDRAM On-Board Options**

**R59** is connected to the **WP** (Write Protect) input of the DDR SDRAM EEPROM. Stuffing a 10 k-Ohm resistor in **R59** will keep the **WP** signal low (inactive). The default configuration is **R59** stuffed.

The EEPROM holds data describing size, configuration, and timing characteristics of the DDR SDRAM. The data is write-protected when the **WP** signal is high. There should be little or no reason to want to overwrite the EEPROM data. Some manufacturers simply connect the **WP** pin of the EEPROM chip to the power supply, in which case the **WP** resistor has no effect whatsoever.

Header **JP4** allows the user to select which clock the DDR SDRAM runs off of. Figure 5-12 shows the DDR Clock Select Jumper **JP4**. The signal **DDR_PLL6**, which is connected to pin 2 of **JP4** connects to the input of the FPGA’s Enhance PLL6. A PLL must be instantiated in the FPGA HDL code to setup the DDR clock signal. This PLL will need to have three (3) output positive differential clocks: one board level clock output used for feedback, one clock input and one feedback input (see Figure 5-12 for a diagram of the DDR PLL circuit).
To make the three positive differential output clocks differential, the \texttt{IO_STANDARD} for all three clock signals needs to be set to \texttt{DIFFERENTIAL SSTL-2}. The Board level PLL clock output is used for the feedback, and needs to have \texttt{IO_STANDARD} set to \texttt{SSTL-2 CLASS I}. The input clock to the PLL must have its \texttt{IO_STANDARD} set to \texttt{SSTL-2 CLASS II}. For information on how to set the \texttt{IO_STANDARD} property, please see the Quantus help files. If the \texttt{IO_STANDARD} properties are set up correctly, then the three negative differential clock signals (\texttt{DDR_CLKn[2:0]}) will automatically be output from the FPGA. You do not need to include these negative clock signals in your FPGA HDL code or pin assignment file.

To change which clock goes to the DDR, the user just needs to change the position of the jumper on \texttt{JP4}. Please note there should only be one jumper on \texttt{JP4}. The DDR Clock Select Jumper is illustrated in Figure 5-13.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{Figure5-12.png}
\caption{DDR PLL Circuit Block Diagram}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{Figure5-13.png}
\caption{DDR Clock Select Jumper (JP4)}
\end{figure}
The DN5000k10S can be hosted in a +3.3 V PCI slot, or it can be used stand-alone. Figure 6-1 shows the various supplies used on the DN5000k10S and the connections of these supplies on the circuit board. The supply, +5 V, from the PCI connector (or P1) supplies the basic power to the DN5000k10S. The +3.3 V power from the PCI connector is not used, nor is it connected to any circuitry on the DN5000k10S.

Figure 6-1  DN5000k10S Power Distribution

The DN5000k10S, when plugged into a PCI slot, has the following different power rails:

- +5 V
- +3.3 V
• +2.5 V
• +1.25 V (tracks to +2.5 V)
• +1.5 V
• –12 V
• +12 V

The power rails +3.3 V, +2.5 V, and +1.25 V and +1.5 V are created using switching regulators with +5 V as the input, while +1.5 V is created with a linear regulator. +3.3 V from the PCI fingers is not used. U17 is for +3.3 V, U16 is for +2.5 V and +1.25 V, and U18 is for +1.5 V. Heat is not an issue with this style of switching regulator. Each regulator should be able to supply the minimum 10 A of current without strain. The most demanding application of the DN5000k10S should fit within the 10 A budget on these two power rails. A heat sink is used to keep the linear regulator within its specification.

### +3.3 V Power

The specification for the +3.3 V power is shown in Table 6-1. The +3.3 V supply is used by the following components on the DN5000k10S:

- Stratix FPGA I/O (U11, banks 1–6)
- Roboclocks U14, U15
- Clock buffer (U12)
- CPLD (U6)
- Microprocessor (U4)
- Microprocessor SRAM (U7)
- 4 SSRAMs (U8, U9, U10, U13)
- DDR SDRAM DIMM (J3)
- 3 Oscillators X1, X2, X3

We do run +3.3 V a little hot. At worst case for all components, the +3.3 V power supply should never fall below +3.30 V.

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+3.35 V</td>
<td>+3.39 V</td>
<td>+3.44 V</td>
</tr>
</tbody>
</table>

### +2.5 V Power

The specification for the +2.5 V power is shown in Table 6-2. The +2.5 V supply is used by the following components on the DN5000k10S:

- Stratix FPGA I/O (U11, banks 7–8)
- DDR SDRAM DIMM (J2)

In addition, the +2.5 V supply outputs a +1.25 V power rail, used by the Stratix FPGA as a reference voltage. The reference voltage tracks to the supply voltage, within 1%.
+1.5 V Power

The specification for the +1.5 V power is shown in Table 6-3. The +1.5 V supply is used by the following component on the DN5000k10S:

- Stratix FPGA $V_{cc\text{INT}}$ (U11)

We also run +1.5 V a little hot. At worst case for all components, the +1.5 V power supply should never fall below +1.50 V.

### Table 6-3 Specification for +1.5 V Power

<table>
<thead>
<tr>
<th></th>
<th>Minimum</th>
<th>Typical</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>+1.55 V</td>
<td>+1.56 V</td>
<td>+1.58 V</td>
</tr>
<tr>
<td>Current</td>
<td>N/A</td>
<td>N/A</td>
<td>12 A</td>
</tr>
</tbody>
</table>

If you use the DN5000k10S in a lab environment, the Stratix FPGA will never see worst case power and temperature—so you can use typical, commercial timing.

NOTE: In a lab environment, the FPGA never sees the worst-case temperature and power. You can use typical, commercial timing!

Stand-Alone Operation

The DN5000k10S can be used stand-alone, meaning it doesn’t have to be plugged into a PCI slot. Connector P1 is used to provide power to the DN5000k10S in this configuration. P1 is a Molex drive power connector and will connect to any standard ATX power supply (see Figure 6-2). The power supply that we used is shown in Figure 6-3, but any ATX or AT style power supply will work. We use a 250-watt ATX supply. Since the DN5000k10S does not draw enough current to meet the minimums required by the supply, we plug an old disk drive into another one of the Molex connectors. The current drawn by the disk drive sinks enough current to make the switchers in the power supply happy.
The P1 connector is rated to 13 A, far more current than the DN5000k10S can use.

The DN5000k10S, when used stand-alone, has the following different power rails:

- +5 V
- +3.3 V
- +2.5 V
- +1.25 V (tracks to 2.5 V)
- +1.5 V
- +12 V

NOTE: If you use the DN5000k10S stand-alone with an ATX power supply, the DN5000k10S may not draw enough current to meet the minimum current required by the switchers in the supply. Connecting a disk drive to another connector will solve this problem!

By specification, a PCI board may consume a maximum of 25 watts from the fingers of the PCI connector. This power limit is below that the DN5000k10S is capable of consuming, even if daughter cards and/or large SDRAM banks are installed. The P1 connector can be used to augment the power obtained from the PCI fingers. P1 can be used provided that the +5 V and +12 V power rails on the connector are supplied by the same power source as the PCI fingers.

NOTE: P1 and PCI may provide power at the same time, but ONLY if the same power source used to supply P1 is also supplying power to PCI.
Daughter Connections to DN3000k10SD—Observation Daughter Card for 200-pin Connectors

Chapter 7

The traditional approach to experiment with new devices involving wiring together some ICs on a breadboard is fast becoming impractical and ineffective. Instead, designers using new high-density devices need custom PC boards representing a substantial investment of time and money.

Prototype boards from manufacturers can meet this demand for experimentation while eliminating the expense and time involved with custom PC boards. Additionally, such prototype boards facilitate the understanding and advantages of new device features.

Purpose

The DN3000k10SD daughter card allows external connection to the signals present on the DN5000k10S series ASIC prototyping boards. The DN5000k10S allows logic emulation with Stratix™ devices prior to committing to using them for specific applications. It allows designers to try Stratix features such as BlockRAM, DLLs, and SelectI/O™ resource, with an off-the-shelf resource.

Features

The DN3000k10SD Daughter Card has the following features:

• Buffered I/O, Passive and Active Bus Drivers
• Unbuffered I/O
• Differential LVDS pairs (Note: Not available on DN5000k10S ASIC prototyping board)
• Headers for Test Points

The daughter card contains headers that may be useful with certain types of oscilloscope probes, or when wiring pins to prototype areas.

Figure 7-1 is a block diagram of the DN3000k10SD Daughter Card.

The DN3000k10SD Daughter Card is pictured in Figure 7-2.

Figure 7-3 shows the assembly drawing of the DN3000k10SD Daughter Card.

The DN3000k10SD Daughter Card provides 16 differential pairs, 48 buffered (passive/active) I/O, and 66 unbuffered I/O signals. The
Figure 7-1 DN3000k10SD Daughter Card Block Diagram
Figure 7-2  DN3000k10SD Daughter Card
IDT74FST163245 chips are used as bus switches in the passive mode, and the IDT74LVC16245A chips are used as bus transceivers in the active mode. The DN3000k10SD has separate enable/direction signals for each driver.

NOTE: Availability of these I/O signals depends on the location of the daughter card with respect to the development board.

**Daughter Card**

**LEDs**

The LEDs act as visual indicators, representing the active power sources.

- **D1** — LED indicating +3.3 V present
- **D2** — LED indicating +5.0 V present
- **D3** — LED indicating +12 V present

Under normal operating conditions, all LEDs should be on.

**Power Supply**

A linear power supply (U4) is present to provide level shift/translation functions when the board is populated with bus switches.
Options

Resistors R10 and R11 can be used to select different voltage sources, +5 V or +3.3 V, respectively. When used, U4 must be removed in order to prevent contention.

NOTE: Never populate R10/R11 simultaneously: this will result in a shorted power supply.

Power Rating

- +5 V power supply is rated for 1 A.
- +3.3 V power supply is rated for 1 A.
- +1.5 V power supply is rated for 1 A.
- +12 V power supply is rated for 0.5 A.
- –12 V power supply is rated for 0.5 A.

Connector J8

Table 7-1 shows the connections of J8.

Table 7-1  Connector J8 Pins External Power

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GND</td>
<td>11</td>
<td>GND</td>
</tr>
<tr>
<td>2</td>
<td>+5 V</td>
<td>12</td>
<td>+1.5 V</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>13</td>
<td>GND</td>
</tr>
<tr>
<td>4</td>
<td>+5 V</td>
<td>14</td>
<td>+12 V</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>15</td>
<td>GND</td>
</tr>
<tr>
<td>6</td>
<td>+3.3 V</td>
<td>16</td>
<td>+12 V</td>
</tr>
<tr>
<td>7</td>
<td>GND</td>
<td>17</td>
<td>GND</td>
</tr>
<tr>
<td>8</td>
<td>+3.3 V</td>
<td>18</td>
<td>–12 V</td>
</tr>
<tr>
<td>9</td>
<td>GND</td>
<td>19</td>
<td>GND</td>
</tr>
<tr>
<td>10</td>
<td>+1.5 V</td>
<td>20</td>
<td>–12 V</td>
</tr>
</tbody>
</table>
LVDS

Low-voltage differential signaling (LVDS) is a signaling method used for high-speed transmission of binary data over copper. It is well recognized that the benefits of balanced data transmission begin to outweigh the costs over single-ended techniques when the signal transmission times approach 10 ns. This represents signaling rates of about 30 Mbps or clock rates of 60 MHz (in single-edge clocking systems) and above. LVDS is defined in the TIA/EIA-644 standards.

NOTE: Not available on the DN5000k10S ASIC prototyping board.

Connector P5

This is a Mini D Ribbon (MDR) connector (50 pin) manufactured by 3M, used specifically for high speed LVDS signaling. The connector mates with a standard off-the-shelf 3M-cable assembly:

P/N 14150-EZBB-XXX-0LC

where XXX is:
- 050 = 0.5 m
- 150 = 1.5 m
- 300 = 3.0 m
- 500 = 5.0 m

Please contact 3M for further details: http://www1.3m.com/.

Unbuffered I/O

The DN3000k10SD Daughter Card provides 66 unbuffered I/O signals, including 5 single ended clock signals. The function of these signals is position dependent.

NOTE: Signals P4NX7 and P4NX6 are also used for direction select and output enable on U2 and U3 respectively.

Connectors P2, P4

P2, P4—Buffered Interface header
IDC headers (50 pin) providing 48 buffered I/O signals.
See Table 7-2 on page 7-8.

Connector P7, P1, P3

P7, P1, P3—Unbuffered Interface Header
IDC headers (50 pin) providing 66 buffered I/O signals.
See Table 7-2 on page 7-8.
Buffered I/O

The DN3000k10SD Daughter Card provides 48 buffered I/O signals. The function of these signals is position dependent. U1, U2, and U3 allow for different populating options, and devices can be active or passive.

Active

The LCV162245A is used for asynchronous communication between data buses. It allows data transmission from the A to the B or from the B to the A bus, depending on the logic level at the direction-control (DIR) input. The output-enable (OE#) input can be used to disable the device so that the busses are effectively isolated.

Passive

The FST163245 bus switches are used to connect or isolate two ports without providing any current sink or source capabilities. Thus, they generate little or no noise of their own while providing a low resistance path for an external driver. The output-enable (OE#) input can be used to disable the device so that the busses are effectively isolated.

Test Interface

The DN3000k10SD Daughter Card provides a 200-pin connector to interface to one of three test connectors on the DN5000k10S ASIC prototyping board: J9, J10 and J16.

Connector J1

J1—Test Interface Connector

Micropax connector (200 pin) used as a standard interface to all the DINI Group development boards. This connector has a specified current rating of 0.5 amps per contact. See Table 7-2 on page 7-8.
Daughter Card I/O Connections

Table 7-2 shows the DN3000k10SD Daughter Card I/O Interconnects to connectors P55, P56 and P58.

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+12 V</td>
<td>J1.184</td>
<td>P8.001</td>
<td>+12 V</td>
<td>P9.001</td>
<td>+12 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>+5 V</td>
<td>J1.006</td>
<td>P8.004</td>
<td>+5 V</td>
<td>P9.004</td>
<td>+5 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>+5 V</td>
<td></td>
<td>P8.006</td>
<td>+5 V</td>
<td>P9.006</td>
<td>+5 V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>GND</td>
<td>J1.204</td>
<td>P8.008</td>
<td>GND</td>
<td>K20</td>
<td>P9.008</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>9</td>
<td>+3.3 V</td>
<td></td>
<td>P8.009</td>
<td>+3.3 V</td>
<td>H20</td>
<td>P9.009</td>
<td>+3.3 V</td>
<td>H20</td>
</tr>
<tr>
<td>22</td>
<td>GND</td>
<td>J1.101</td>
<td>P8.022</td>
<td>GND</td>
<td>K20</td>
<td>P9.022</td>
<td>GND</td>
<td>K20</td>
</tr>
</tbody>
</table>
### Table 7-2 Daughter Board-Header-FPGA Pin Map

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>GND</td>
<td>J5.20 J6.22</td>
<td>P8.033</td>
<td>GND</td>
<td>K20</td>
<td>P9.033</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>44</td>
<td>GND</td>
<td>J3.36</td>
<td>P8.044</td>
<td>GND</td>
<td>K20</td>
<td>P9.044</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>55</td>
<td>GND</td>
<td>J1.044</td>
<td>P8.055</td>
<td>GND</td>
<td>K20</td>
<td>P9.055</td>
<td>GND</td>
<td>K20</td>
</tr>
</tbody>
</table>
Table 7-2  Daughter Board-Header-FPGA Pin Map

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>56</td>
<td>P3N[38]</td>
<td>U2.44 J4.11</td>
<td>P8.056</td>
<td>TST_HDRA[40]</td>
<td>AP6</td>
<td>P9.056</td>
<td>TST_HDRB[40]</td>
<td>H29</td>
</tr>
<tr>
<td>66</td>
<td>GND</td>
<td>J1.044</td>
<td>P8.066</td>
<td>GND</td>
<td>K20</td>
<td>P9.066</td>
<td>GND</td>
<td>K20</td>
</tr>
</tbody>
</table>
### Table 7-2 Daughter Board-Header-FPGA Pin Map

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>88</td>
<td>GND</td>
<td>J1.184</td>
<td>P8.088</td>
<td>GND</td>
<td>K20</td>
<td>P9.088</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>93</td>
<td>+1.5 V</td>
<td>J1.103</td>
<td>P8.093</td>
<td>+1.5 V</td>
<td>AE20</td>
<td>P9.093</td>
<td>+1.5 V</td>
<td>AE20</td>
</tr>
<tr>
<td>100</td>
<td>-12 V</td>
<td>J5.26 J6.24 J7.12</td>
<td>P8.101</td>
<td>-12 V</td>
<td>P9.100</td>
<td>-12 V</td>
<td>P9.100</td>
<td>G20</td>
</tr>
<tr>
<td>103</td>
<td>+1.5 V</td>
<td>J2.27</td>
<td>P8.103</td>
<td>+1.5 V</td>
<td>AE20</td>
<td>P9.103</td>
<td>+1.5 V</td>
<td>AE20</td>
</tr>
<tr>
<td>104</td>
<td>MBCK[0]</td>
<td>J.2.28</td>
<td>P8.104</td>
<td>GND</td>
<td>K20</td>
<td>P9.104</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>105</td>
<td>+3.3 V</td>
<td>J5.9</td>
<td>P8.105</td>
<td>+3.3 V</td>
<td>H20</td>
<td>P9.105</td>
<td>+3.3 V</td>
<td>H20</td>
</tr>
<tr>
<td>J1</td>
<td>Signal</td>
<td>Daughter Board Header</td>
<td>Test Header P8</td>
<td>P8 Signal</td>
<td>FPGA Pin U11</td>
<td>Test Header P9</td>
<td>P9 Signal</td>
<td>FPGA Pin U11</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>-----------------------</td>
<td>----------------</td>
<td>-----------</td>
<td>---------------</td>
<td>----------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>118</td>
<td>GND</td>
<td>J1.033</td>
<td>P8.118</td>
<td>GND</td>
<td>K20</td>
<td>P9.118</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>133</td>
<td>P3N[79]</td>
<td>J2.3</td>
<td>P8.133</td>
<td>TST_HDRA[99]</td>
<td>K11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>137</td>
<td>P3N[71]</td>
<td>J2.33</td>
<td>P8.137</td>
<td>TST_HDRA[103]</td>
<td>F11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>138</td>
<td>P3N[70]</td>
<td>J2.34</td>
<td>P8.138</td>
<td>TST_HDRA[104]</td>
<td>AG12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>139</td>
<td>P3N[65]</td>
<td>J5.43</td>
<td>P8.139</td>
<td>TST_HDRA[105]</td>
<td>AF12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>140</td>
<td>GND</td>
<td>J1.203</td>
<td>P8.140</td>
<td>GND</td>
<td>K20</td>
<td>P9.140</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>142</td>
<td>P3N[61]</td>
<td>J5.47</td>
<td>P8.142</td>
<td>TST_HDRA[107]</td>
<td>AD12</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 7-2  Daughter Board-Header-FPGA Pin Map

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>147</td>
<td>P3N[52]</td>
<td>J6.7</td>
<td>P8.147</td>
<td>TST_HDRA[112]</td>
<td>U12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>149</td>
<td>P3N[50]</td>
<td>J2.18</td>
<td>P8.149</td>
<td>TST_HDRA[114]</td>
<td>R12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>151</td>
<td>GND</td>
<td>J1.011</td>
<td>P8.151</td>
<td>GND</td>
<td>K20</td>
<td>P9.151</td>
<td>GND</td>
<td>K20</td>
</tr>
</tbody>
</table>
Table 7-2  Daughter Board-Header-FPGA Pin Map

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>184</td>
<td>GND</td>
<td>J5.16 J7.2</td>
<td>P8.184</td>
<td>GND</td>
<td>K2</td>
<td>P9.184</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>188</td>
<td>P4N[5]</td>
<td>J7.21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>190</td>
<td>P4N[1]</td>
<td>J7.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>191</td>
<td>P4N[0]</td>
<td>J7.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>P4NX[12]</td>
<td>J7.31</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>194</td>
<td>P4NX[9]</td>
<td>J7.33</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>196</td>
<td>P4NX[8]</td>
<td>J7.35</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>199</td>
<td>P4NX[1]</td>
<td>J7.41</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>PRNX[0]</td>
<td>J7.43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 7-2 Daughter Board-Header-FPGA Pin Map

<table>
<thead>
<tr>
<th>J1</th>
<th>Signal</th>
<th>Daughter Board Header</th>
<th>Test Header P8</th>
<th>P8 Signal</th>
<th>FPGA Pin U11</th>
<th>Test Header P9</th>
<th>P9 Signal</th>
<th>FPGA Pin U11</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>GND</td>
<td>J1.204</td>
<td>P8.201</td>
<td>GND</td>
<td>K20</td>
<td>P9.201</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>203</td>
<td>GND</td>
<td>J5.4</td>
<td>P8.203</td>
<td>GND</td>
<td>K20</td>
<td>P9.203</td>
<td>GND</td>
<td>K20</td>
</tr>
<tr>
<td>204</td>
<td>GND</td>
<td>J5.2</td>
<td>P8.204</td>
<td>GND</td>
<td>K20</td>
<td>P9.204</td>
<td>GND</td>
<td>K20</td>
</tr>
</tbody>
</table>
Chapter 8

Reset Schemes, LEDs, Bus Bars and 200 Pin Connectors

Reset Schemes

A LTC1326 chip from Linear Technology controls reset functionality for the DN5000k10S. Figure 8-1 shows the distribution of the reset signal PWRRST-. In addition to controlling the reset, the power supplies rails +5 V, +3.3 V, +2.5 V and +1.5 V are threshold detected by the LTC1326. Undervoltage conditions will cause the assertion of the reset signal.

The LTC1326 has a push-button. Momentarily depressing this button causes a 200 ms reset pulse on the signal PWRRST-. If the push-button is depressed for 2 seconds and held, PWRRST- is asserted continuously. LED5, when lit, means that reset is asserted, so if you press and hold S1, you should see LED5 illuminate after a few seconds. If LED5 illuminates for any reason during normal operation, this indicates that PWRRST- is active and that something is wrong. Note that if you press S1 and release it quickly, you probably won’t see LED5 since the 200 ms reset pulse is not long enough for the eye to observe.

Depressing the push-button S1 causes the following sequence of events:

1. Reset of the CPLD and μP
2. FPGA configuration is cleared
3. If the switches on S2 are set for Fast Passive Parallel and there is a valid SmartMedia card inserted into the socket, then the FPGA will be configured. A SmartMedia card is valid if it complies with the SSFDC specification and contains a file named main.txt in the root directory. If the card is invalid or there is no card present, then the FPGA will not be configured.
4. The Main Menu will appear.

The identical sequence of events occurs at power-up.
Reset Schemes, LEDs, Bus Bars and 200 Pin Connectors

Figure 8-1  Reset Functionality

Note: RS232 Transceiver must be disabled during µP programming phase in order to avoid contention on the BTXD signal pin.
LEDs

The DN5000k10S has eight LEDs that are used to visually communicate the status of circuitry (Figure 8-2).

From left to right, the LEDs are labeled: CPLD_LED0, CPLD_LED1, CPLD_LED2, CPLD_LED3, UP_LED0, UP_LED1, UP_LED2, UP_LED3 (see Figure 8-3).

The LEDs have the following functions:

- UP_LED3: Lights when the configuration process from the SmartMedia was successful.
- UP_LED[2:0]: These three LEDs have multiple meanings:
  - When all 3 LEDs are blinking, then the µP has been reprogrammed and is waiting for the user to enter FPGA stuffing.
information via serial port or there is not a valid SmartMedia card present and the FPGA has been configured.

- During configuration, the combination of LEDs lit tells the user which FPGA is currently being configured (UP_LED2, UP_LED1, UP_LED0):
  - off, off, on = FPGA F
  - off, on, off = FPGA A
  - off, on, on = FPGA E
  - on, off, off = FPGA B
  - on, off, on = FPGA D

CPLD_LED3 lights when the FPGA is NOT configured
CPLD_LED2 lights when reset is asserted (PWRST−)
CPLD_LED1 lights when the PLL in Roboclock I is LOCKED
CPLD_LED0 lights when the PLL in Roboclock II is LOCKED

You are free to reprogram the CPLD and/or microprocessor to use any or all of the LEDs for your own purposes.

**Bus Bars**

The two bus bars, B1 and B2, are installed to prevent flexing of the PWB and serve no other purpose. They are connected quite solidly into the ground plane of the DN5000k10S at every hole, and you can use the metal bars to ground-test equipment such as oscilloscopes and pattern generators. Be careful not to short any power rails or signals to these metal bars—they can carry a lot of current. The PCI bracket, BRK1, is also connected to the ground plane at each of the screw mounts.

**The 200 Pin Connectors: P8 and P9**

The DN5000k10S contains two 200-pin connectors, P8 and P9. Daughter cards of any sort may be plugged into these connectors. The relative pin location of the powers, grounds, and signals is identical for each of the connectors. A hole that can be used to attach a standoff is located at the same relative position from each connector (see Figure 8-4). This hole is grounded on the DN5000k10S, so connect this mounting hole to digital ground on your daughter card.

The mechanical position of the 200-pin connectors on the DN5000k10S is shown in Figure 8-4. The 200-pin connector used on the DN5000k10S is a Berg Electronics 91294-003 in the Micropax™ family. This link will take you to the Berg website: [http://www.berg.com/](http://www.berg.com/). This Berg connector was chosen because of its high pin density, performance, and availability. The part number for the mating connector is 91403-003. We stock the mating connector at our offices in La Jolla, CA, so if you are designing a daughter card and are having trouble getting this part, call us. We would be happy to send you a few at our cost. Appendix A contains a mechanical datasheet for both the Berg 91403-003 and 91294-003 connectors.

This style of connector has four mounting holes—two screw holes at each end and two alignment holes between pins 50–51 and after pin 100 (see Figure 8-4). These mounting holes are part of the metal shell of the connector and make an important connection to the mating connector.
All four of these mounting holes are connected to digital ground on the DN5000k10S—therefore, the shell of the connector is grounded.

We used the pin numbering shown in Figure 8-4 for the 200-pin, 91294-003 connectors.

**The Signals**

Each of the two 200-pin connectors has the following:

- **P8** has 147 signals connected to the FPGA.
- **P9** has 96 signals connected to the FPGA.
- 5clocks
- The following power rails:
  - +12V (1 pin)
  - -12V (1 pin)
  - +5V (2 pins)
Reset Schemes, LEDs, Bus Bars and 200 Pin Connectors

- +3.3V (2 pins)
- +1.5V (2 pins)
- GND (23 pins + case)

Regarding the amount of current that the power pins can carry, the following text is lifted directly from the specification for the Micropax™ family of connectors:

6.1 Current Rating. Current rating shall be evaluated in still air at 25°C ambient temperature. Under the following conditions, the temperature rise shall be no greater than 30°C:

- All contacts powered at 0.5 amp
- One contact powered at 3.0 amps

Most of the signals are TTL (or some low current variation such as LVDS), so you can reasonably expect to get up to 3 amps per power pin through this connector. Remember that the +3.3V and +1.5V power supplies are limited to 5 amps total—the memories, the FPGA and the clock circuitry on the DN5000k10S consume +3.3V. The FPGA only consumes +1.5V.

If you use the DN5000k10S stand-alone (meaning that it is not plugged into a PCI slot), the auxiliary power connector has +5V and +12V, but does not have −12V. So unless you provide −12V to the DN5000k10S via another connection, −12V will not be available for use by a daughter card.

The 200-pin connectors are shown in Figure 8-5.

NOTE: −12V is not required by the DN5000k10S. The DN5000k10S will operate normally without a −12V power supply.
Figure 8-5  200-Pin Connectors — Signal Connections
Reset Schemes, LEDs, Bus Bars and 200 Pin Connectors
Chapter 9

Utilities

PCI Debug—General Pontificating

Debugging of PCI-based hardware can be troublesome, so it is best to do so with a tiered approach. The following sequence of events needs to occur for a PCI-based peripheral to start working:

1. The hardware must boot itself at power-up — in the case of the DN5000k10S:
   a. The µP must boot.
   b. Recognize the SmartMedia card.
   c. Configure the FPGA. (Hopefully, all this occurs before RST# on the PCI bus is deasserted.)
2. The PCI BIOS executes the PNP routines and configures the BARs on all PCI peripherals.
3. The operating system driver initializes the card.
4. The application initiates communication with the driver and the application executes.

The steps are dependant. Each of the steps must start and execute flawlessly before the next step occurs. When you get a PCI card for the first time, it is necessary to debug each step before attempting to go to the next. We provide utilities to help with each step.

Steps 1 and 2 are best done without an operating system in place. Windows NT-based systems take minutes to reboot after a crash (the BLUE-screen-of-death) and an NT driver won’t work unless the hardware is debugged. Since crashing is a regular occurrence in a PCI hardware debug environment, we find it easiest to do our debug and manufacturing test in the old DOS environment. Virtually all PCI peripherals get configured with addresses beyond the IM boundary. On a PC, C programs cannot access memory locations beyond IM unless special programs called DOS extenders are used. Several freeware DOS extenders are available. We use a free DOS-extender called DJGPP. More information can be found at http://www.delorie.com/DJGPP.

PC-Based—AETEST.EXE

A utility program called AETEST is provided with the DN5000k10S. AETEST can be run under DOS, Windows 98/ME, Windows NT/2000, or LINUX. When used under DOS, you must boot your PC with a DOS disk. We ship one with the DN5000k10S in case you don’t know how to make one on your own. All features work in the native mode of AETEST, which is DOS.
All source code for AETEST is provided, so you are welcome to customize the program to your own applications.

AETEST is not a stable program. We add and subtract features when we need to for debug and verification purposes, so don’t be concerned if the screens that you see aren’t exactly replicated here.

In a nutshell, AETEST lets you do the following:

- Determine if PCI recognizes the DN5000k10S
- Read/write/loop any memory location
- Read/write/loop configuration space
- Display all configured PCI devices
- Display memory setting from any locations
- Fill memory with various patterns
- Run various tests on the DN5000k10S
  - SSRAM Test
  - Multiplier Test
  - SDRAM Test
  - Interconnect Test
  - Daughter Card Test

### Installation Instructions for DOS

1. The files `aetestdj.exe` and `cwsdpmi.exe` (the DOS extender) need to be in the same directory.
2. Run `aetestdj.exe`.

### Installation Instructions for Windows NT

1. Install the device driver: `install.exe` and `qldriver.sys` must be in the same directory.
2. Type “install”
3. After the driver is installed, start the driver by selecting `Control Panel`–>`Devices`–>`find “QLDriver”`–>`click “Start”`
4. Run `aetestnt.exe`.

### Installation Instructions for Windows 2000

1. Install the device driver: `qldriver2000.inf` and the driver file (`qldriver.sys`) should be in the same directory.
2. Open `Control Panel`, click on “Add/Remove Hardware” and then go to “Next–>.”
3. Choose `Add/Troubleshoot a device` (the default option) and click on “Next–>.”
4. Wait until it finishes new hardware device searching; choose “Add a new device,” and click on “Next–>.”
5. Choose “No, I want to select the hardware from a list,” and press “Next–>.”
6. Choose “Other Devices” from `Hardware Types` list and press “Next–>”
7. Click on “Have Disk…”

---

*Utilities*

---
8. In “Copy Manufacturer’s Files From:” window, find the directory where qldriver.sys is located, then press “OK.”

9. You should see “dn2000k10 driver” under Models; click on “Next–>.”

10. Press “Next–>” and then “Finish.”

11. Run aetestnt.exe.

**Installation Instructions for LINUX**

This has been tested on Red Hat Linux 7.2 (kernel version 2.4.x).

Note that all the text files, including the scripts, are DOS text format (with an extra carriage return character after every new line), so you need to convert them.

1. You must be root to start the driver and the program. “dndev_load” and “dndev_unload” are scripts that load and unload the driver; “dndev.o” is the driver file.

2. Load the driver; type “sh dndev_load”

3. Unload the driver; type “sh dndev_unload”

4. After driver is loaded, run the utility aetest_linux.

5. Note: You might need to run chmod on aetest_linux to make it executable: type “chmod u+x aetest_linux.”

**Installation Instructions for Solaris**

The utility and driver are tested on Solaris 7.0/Sparc, with the 32-bit kernel.

Note that all the text files, including the scripts, are DOS text format (with an extra carriage return character after every new line), so you need to convert them.

1. To install the driver, go to the driver directory, make sure the driver file “dndev” is in the sparc sub-directory, and run “sh dndev_uninstall.sh”

2. To uninstall the driver, run “sh dndev_uninstall.sh”

3. To run the test utility, run “aetest_solaris” as root after the driver is loaded.

The driver is compiled with the gcc compiler.

*aetest_solaris* is compiled with “gmake.” You can download it from the GNU website. The “make” from the Solaris installation does not work with our makefile format.

You may need to make aetest_solaris executable; run “chmod u+x aetest_solaris.”
Utilities

Installation Instructions for Windows 98/ME

There are two ways to run AETEST: You can run the DOS version “aetestdj.exe” directly, or you can run AETEST with a device driver.

To run AETEST with a device driver follow the steps below.

1. Choose a default PCI driver for the device. When Windows first starts with the device plugged in, it should ask for a device driver. Select “Specify the location of the driver.”
2. Select “Display a list of the drivers in a specific location…”
3. Select “Other devices.”
4. Under “ Manufacturers” tab, select “unknown device.”
5. Under “Models” select “unsupported device.”
6. The driver file (pcifg.vxd) and aetest98.exe must be in the same directory. Run aetest98.exe.

NOTE: To re-compile the driver file pcifg.vxd, you need the VtoolsD compiler from www.numega.com.

AETEST Options:
Description and Definitions

<table>
<thead>
<tr>
<th>Description</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Startup</td>
<td>When AETEST is first started, it tries to find a device that it recognizes. We have arbitrarily defined the DN5000k10S with a DEVICE_ID of 0x1505 and a VENDOR_ID of 0x17DF. You should see the following screen if AETEST recognizes a DN5000k10S (Figure 9-1):</td>
</tr>
</tbody>
</table>

```
searching for "DN2000K10 Asic Emulator (1000R)" VENDOR_ID--abcd, DEVICE_ID--1236
searching for "DN2000K10 Asic Emulator (1600R)" VENDOR_ID--abcd, DEVICE_ID--1237
searching for "DN3000K10S Asic Emulator (6000)" VENDOR_ID--abcd, DEVICE_ID--1240
```

found device ---- vabcd, d1240 name="DN3000K10S Asic Emulator (6000)"

Configuration space:


BAR0: base: 0xfd800000, size: 0x00800000
BAR1: base: 0xe0000000, size: 0x10000000
BAR2: base: 0x00000000, size: 0x00000000
BAR3: base: 0x00000000, size: 0x00000000
BAR4: base: 0x00000000, size: 0x00000000
BAR5: base: 0x00000000, size: 0x00000000

press any key

Figure 9-1  DN5000k10SAETEST Startup Screen, DN5000k10S Recognized

Most of this initial display is debug information. The program is looking for a Vendor and Device ID that it recognizes, and finds vendor=0x17DF and device=0x1505, which is a DN5000k10S. The lines after Configura-
tion space: show what is in the configuration space and how the BARs are configured.

If AETEST does not see a PCI peripheral it recognizes, you will see the following (Figure 9-2):

```
searching for "Quad Sharc" VENDOR_ID==5045, DEVICE_ID==1
searching for "DN2000K10 Asic Emulator" VENDOR_ID==abcd, DEVICE_ID==1234
searching for "DN2000K10 Asic Emulator (2000E)" VENDOR_ID==abcd, DEVICE_ID==1235
searching for "DN2000K10 Asic Emulator (1000E)" VENDOR_ID==abcd, DEVICE_ID==1236
searching for "DN2000K10 Asic Emulator (1600E)" VENDOR_ID==abcd, DEVICE_ID==1237
searching for "DN3000K10S Asic Emulator (6000)" VENDOR_ID==abcd, DEVICE_ID==1240
searching for "q15064 interface test" VENDOR_ID==1234, DEVICE_ID==5678
searching for "q15064 64MB dram:LFCSR" VENDOR_ID==1234, DEVICE_ID==5679
searching for "q15064 PowerPC bridge" VENDOR_ID==11e3, DEVICE_ID==6
searching for "q15064 PowerPC bridge (old VID/DID)" VENDOR_ID==1010, DEVICE_ID==5064
searching for "q15064 AntiFuse test #1" VENDOR_ID==71f3, DEVICE_ID==24b4
searching for "q15064 AntiFuse test #2" VENDOR_ID==507, DEVICE_ID==2367
searching for "q15064 AntiFuse test #3" VENDOR_ID==bc92, DEVICE_ID==2e6c
searching for "q15064 AntiFuse test #4" VENDOR_ID==e125, DEVICE_ID==c38c
searching for "q15064 AntiFuse test #5" VENDOR_ID==e62c, DEVICE_ID==ca76
searching for "q15064 AntiFuse test #6" VENDOR_ID==448b, DEVICE_ID==e6a
searching for "q15064 Emulated with 8051" VENDOR_ID==1243, DEVICE_ID==4321
searching for "Greg's PCI Device" VENDOR_ID==3143, DEVICE_ID==2
searching for "Cohu's PCI Device (sensor board)" VENDOR_ID==dead, DEVICE_ID==beef
searching for "LYNX 9610" VENDOR_ID==10b5, DEVICE_ID==9610

Didn't find known device in the following list:
  vendor_id=5045, device_id=1
  vendor_id=abcd, device_id=1234
  vendor_id=abcd, device_id=1235
  vendor_id=abcd, device_id=1236
  vendor_id=abcd, device_id=1237
  vendor_id=abcd, device_id=1240
  vendor_id=1234, device_id=5678
  vendor_id=1234, device_id=5679
  vendor_id=11e3, device_id=6
  vendor_id=1010, device_id=5064
  vendor_id=71f3, device_id=24b4
  vendor_id=507, device_id=2367
  vendor_id=bc92, device_id=2e6c
  vendor_id=e125, device_id=c38c
  vendor_id=e62c, device_id=ca76
  vendor_id=448b, device_id=e6a
  vendor_id=1243, device_id=4321
  vendor_id=dead, device_id=beef
  vendor_id=10b5, device_id=9610

Hit a key to continue
```

Figure 9-2  AETEST Startup Screen, No PCI Peripheral Recognized

AETEST will still run, but many DINI product-specific options will not be available.
The AETEST Main Screen is shown in Figure 9-3.

--- ASIC Emulator PCI Controller Driver --- v8

0) Read FPGA revision
1) PCI Menu
2) Memory Menu
3) Flash Menu
4) Clock Menu
5) Dedicated Multiplier Test

Q) Quit

--- PCI BASE ADDRESS ---
0 : 0xFD800000 1 : 0xE0000000 2 : 0x00000000
3 : 0x00000000 4 : 0x00000000 5 : 0x00000000

Please select option:

Figure 9-3   AETEST Main Screen

Options

Read FPGA Revision. Display the revision ID of the FPGA. We will update the revision ID of the FPGA every time we change the reference design.

PCI Menu. Display the PCI utilities menu.

Memory Menu. Display the Memory Menu.

Flash Menu. Display the Flash Utilities Menu (DN2000k10 series only!).

Clock Menu. Display the Clock Utilities Menu.

Dedicated Multiplier Test. Execute the multiplier test.

Q. Quit and return to the DOS prompt

The selections are sometimes case sensitive, so be aware of the status of the CAPS LOCK on your keyboard. The base addresses for each of the configured BARs is displayed on all screens. You will need these addresses if you want to manually read and write to address locations within the PCI reference design. In this example (Figure 9-3, above), BAR0 is configured to 0xFD800000 and BAR1 is configured to 0x00000000. BAR[5:2] are not configured so they show up as 0x0.
**PCI Menu**

The AETEST PCI menu is shown in Figure 9-4.

--- ASIC Emulator PCI Controller Driver -- v8
PCI Device/Function Num: 0x7F, 0x00

S) Set PCI Device Number
F) Set PCI Function Number
D) Display all Configured PCI Devices
1) Display Vendor and Device ID for PCI device-function: 7f-0
2) Loop on PCI device-fun: 7f-0 and Display Vendor and Device ID
3) Loop on PCI device-fun: 7f-0 and Don’t Display Vendor and Device ID
4) Loop on all PCI device numbers and Display Device/Vendor ID’s
5) Display all PCI information for PCI device-function: 7f-0
6) Write config dword
7) Read config dword

C) Configure BAR's from File
V) Save BAR Configuration to File
M) Main Menu
Q) Quit

--- PCI BASE ADDRESS ---
0 : fdd800000 1 : e00000000 2 : 00000000
3 : 00000000 4 : 00000000 5 : 00000000

Please select option:

--- Figure 9-4  AETEST PCI Menu ---

**Set PCI Device Number.** Sets a PCI device number of your choice as the “active” device (hex input). This option lists the available Device Numbers to help you match up your Device ID and Vendor ID with the device number.

**Set PCI Function Number.** Sets a PCI function number of your choice as the “active” function of a multi-function device (hex input). This option lists the Device ID and Vendor ID of each function within the “active” device number to help you to choose the desired function.

**Display all Configured PCI Devices.** Displays the PCI Device Numbers and corresponding Device ID and Vendor ID of all devices seen on the bus. This does not display device numbers with a Device ID and Vendor ID of all ones (0xFFFF).

**Display Vendor and Device ID for PCI device-function.**

Displays the Vendor ID and Device ID of the active device and function number. In the example above, this would display the Vendor ID and Device ID of the PCI device at device number 0x7F, function number 0x00.

**Loop on PCI device-fun: 7f-0 and Display Vendor and Device ID.** Reads and displays the Vendor ID and Device ID of the “active” device number and function number. Repeats this action until the user hits a key to stop it.
Loop on PCI device-function: 7f-0 and Don’t Display Vendor and Device ID. Same as previous menu option, except doesn’t display results. This menu option is useful when using an oscilloscope to debug configuration reads.

Loop on all PCI device numbers and Display Device/Vendor ID’s. Loops on each device number, reading the Vendor ID and Device ID for each. It moves onto the next device number when you press any key. That is, it continually reads the Vendor ID and Device ID from device number 0 until you hit a key, at which point it continually reads the Vendor ID and Device ID from device number 1. It moves all the way through device number 0 to device number \(0x7F\) (in case there are any bridges on your PCI bus).

Display all PCI information for PCI device-function: 7f-0.
Reads and displays all of the configuration space for the “active” device and function number. Use options “S” and “F” to change between the “active” device number and function number, and then use this option to view the entire configuration space.

Write config(uration) DWORD. Allows write to configuration space. The following text will appear to remind you what is in configuration space for a PCI device:

```
PCI_CS_VENDOR_ID 0x00
PCI_CS_DEVICE_ID 0x02
PCI_CS_COMMAND 0x04
PCI_CS_STATUS 0x06
PCI_CS_REVISION_ID 0x08
PCI_CS_CLASS_CODE 0x09
PCI_CS_CACHE_LINE_SIZE 0x0c
PCI_CS_MASTER_LATENCY 0x0d
PCI_CS_HEADER_TYPE 0x0e
PCI_CS_BIST 0x0f
PCI_CS_BASE_ADDRESS_0 0x10
PCI_CS_BASE_ADDRESS_1 0x14
PCI_CS_BASE_ADDRESS_2 0x18
PCI_CS_BASE_ADDRESS_3 0x1c
PCI_CS_BASE_ADDRESS_4 0x20
PCI_CS_BASE_ADDRESS_5 0x24
PCI_CS_EXPANSION_ROM 0x30
PCI_CS_INTERRUPT_LINE 0x3c
PCI_CS_INTERRUPT_PIN 0x3d
PCI_CS_MIN_GNT 0x3e
PCI_CS_MAX_LAT 0x3f
```

Input config offset (hex 0x00-0xff):
word to write (in hex):

Loop indefinitely? (y or n)?

If looping was selected, any keypress will stop the loop.
Utilities

Read config(uration) DWORD. Allows read from configuration space. Has options for single read, loop read with display, and loop read without display.

Configure BARs from File. Reloads the PCI configuration of the “active” device from a file. It writes 0x001F to the command register, and writes the 6 bars with the values from the file. This is useful for hot-swapping devices (power switch still required on extender), or reinitializing a device when its configuration has been altered.

WARNING: Because the PCI BIOS is not assigning the BARs for this device, you may induce a memory conflict by using this option. This option is for advanced users only!

Save Bar Configuration to File. Writes PCI Device ID, Vendor ID and the BARs into a file (from the “active” device). This option is for advanced users only!

Memory Menu

The memory menu (Figure 9-5) allows you to perform a variety of tests of PCI memory along with some DN5000k10S specific tasks.

--- ASIC Emulator PCI Controller Driver --- v8
1) Write To Memory Test    2) Read Memory Test
3) Write/Read Test        4) Memory Fill
8) Memory Display
9) Write Memory Byte
a) Read Memory Byte
b) Write/Read Memory Byte
c) memory test on SSRAM 1
d) memory test on SSRAM 2
e) memory test on SSRAM 3
f) memory test on SDRAM
g) full memory test (including blockram)
h) memory test on FPGA block memory
p) bar memory range test
u) SRAM memory test
M) Main Menu
Q) Quit

--- PCI BASE ADDRESS ---
0 : fd800000 1 : e0000000 2 : 00000000
3 : 00000000 4 : 00000000 5 : 00000000

Figure 9-5  ATEST Memory Menu
Utilities

Write to Memory Test. Write a selected number of long words to a specific PCI memory location (Figure 9-6).

```
Memory location (hex) :fb000000
Numbers of long words to write (in decimal) ? 2
long word to write (in hex) :aaaaaaaa
long word to write (in hex) :55555555
Loop indefinitely? (y or n)?
Hit a key to continue....
```

Figure 9-6 AETEST Write to Memory Test

You will be prompted for the memory location (in hex). The physical address is needed. All 4 gigabytes of PCI memory can be accessed. A minimum of 1 to a maximum of 1024 long words can be written, in sequential order, to the same address. A looping option is available if you want to use an oscilloscope. If you are in a scope loop, any keypress will terminate the loop and return you to the main menu.

Read Memory Test. Read a single long word from a specific PCI memory location (Figure 9-7).

```
--------Input address :fb000000
fb000000
  1. Display result
  2. Display result and loop indefinitely
  3. Don't display result and loop indefinitely
Please select:
```

Figure 9-7 AETEST Read Memory Test

You will be prompted for the memory location (in hex). The physical address is needed. All 4 gigabytes of PCI memory can be read. Three options are available:

1. Read once and display.
2. Read indefinitely and display.
3. Read indefinitely and don't display.
Write/Read Test. Write a long word to a specific PCI memory location and immediately read what was written. Repeat for a selected number of long words (Figure 9-8).

```
Input address :fb000000
Numbers of long words to write (in decimal) ?  2
long word to write (in hex) :000
long word to write (in hex) :aaa
  1. Display result
  2. Display result and loop indefinitely
  3. Don't display result and loop indefinitely
Please select:
```

Figure 9-8  ATEST Write/Read Test
You will be prompted for the memory location (in hex). The physical address is needed. All 4 gigabytes of PCI memory can be read. The program will prompt for the number of long words you wish to write (1 to 1024). Three options are available:

1. Read once and display.
2. Read indefinitely and display.
3. Read indefinitely and don’t display.

Option 3 is a very useful scope loop.

Memory Fill. Fill memory with a selected pattern (Figure 9-9).

```
Input starting address (hex and 32 bit aligned): fb000000
Input number of bytes (divisible by 4): 1000
  1 -- fill with 0
  2 -- address=data
  3 -- 0x55555555, 0xaaaaaaaa
  4 -- 0xffffffff
  5 -- data=~address
```

Figure 9-9  ATEST Memory Fill
You will be prompted for the memory location (in hex). The physical address is needed. All 4 gigabytes of PCI memory can be written. The program will prompt for the number of bytes (in hex) you wish to fill (4 to 0xffffffff). The following fill options are available:

1. **fill with 0** — fill all the locations with 0x00000000 (clear the memory)
2. **address=data** — fill each long word with its address
3. **alternating** 0x55555555, 0xaaaaaaaa
4. **0xffffffff** — set all of memory
5. **data=~address** — fill each long word with the address (each bit inverted).
**Utilities**

**Memory Display.** Display 160 long words of memory. You are prompted for the starting address (in hex):

**Input starting address (hex and 32 bit aligned):**

The following screen is displayed (Figure 9-10):

<table>
<thead>
<tr>
<th>0</th>
<th>4</th>
<th>8</th>
<th>c</th>
<th>10</th>
<th>14</th>
<th>18</th>
<th>lc</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000020</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000040</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000060</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000080</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>0000a0</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>0000c0</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>0000e0</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000100</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000120</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000140</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000160</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000180</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>0001a0</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>0001c0</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>0001e0</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000200</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000220</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000240</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
<tr>
<td>000260</td>
<td>000000</td>
<td>000000</td>
<td>000000</td>
<td>0000000</td>
<td>00000000</td>
<td>00000000</td>
<td>00000000</td>
</tr>
</tbody>
</table>

(f) **forward** — pages the screen forward in memory.

(b) **back** — pages the screen backwards in memory.

(j) **jump** — jump to a specific location (in hex).

(0) **goto** — jump back to the original address location specified at the beginning.

(d) **delay and display loop** — display, wait for a second, and display again. Loop until a key is struck.

**Write Memory Byte.** Write a specific number of bytes to a single memory address (Figure 9-11).

Input address : **fb000000**

Numbers of long words to write (in decimal) ? 2

Byte to write (in hex) : **ff**
byte to write (in hex) : **aa**

1. Display result
2. Display result and loop indefinitely
3. Don't display result and loop indefinitely

Please select:

**Figure 9-11  AETEST Write Memory Byte**

You will be prompted for the memory location (in hex). The physical address is needed. All 4 gigabytes of PCI memory can be accessed. A minimum of 1 to a maximum of 1024 bytes will be written, in sequential order, to the same address. A looping option is available if you want to use
an oscilloscope. If you are in a scope loop, any keypress will terminate the loop and return you to the main menu.

**Read Memory Byte.** Read a single byte from a specific PCI memory location (Figure 9-12).

```
Input address : fb000000
fb000000
  1. Display result
  2. Display result and loop indefinitely
  3. Don't display result and loop indefinitely
Please select:
```

**Figure 9-12  AETEST Read Memory Byte**

You will be prompted for the memory location (in hex). The physical address is needed. All 4 gigabytes of PCI memory can be read. Three options are available:

1. Read once and display.
2. Read indefinitely and display.
3. Read indefinitely and don’t display.

**Write/Read Memory Byte.** Write and read a single DWORD from a specific PCI memory location. After entering a memory address (hex, 32 bits), you specify how many DWORDS you want written and read back, and the data. Then, you choose from the 3 options as above. The menu option does not perform any data checking. (Figure 9-13)

```
Numbers of long words to write (in decimal) ? 2
byte to write (in hex) : 88888888
byte to write (in hex) : 99999999
  1. Display result
  2. Display result and loop indefinitely
  3. Don’t display result and loop indefinitely
Please select:
```

**Figure 9-13  AETEST Write/Read Memory Byte**

**Memory test on SSRAM1.** Tests one of the SSRAM chips on the DN5000k10S.

**Memory test on SSRAM2.** Tests one of the SSRAM chips on the DN5000k10S.

**Memory test on SSRAM3.** Tests one of the SSRAM chips on the DN5000k10S.

**Memory test on SDRAM.** Tests the SDRAM chip on the DN5000k10S.

**Full Memory Test (Including BlockRAM).** Tests all of the memories. This includes the SSRAM chips, the SDRAM, and the BlockRAM internal to the FPGA.
**Utilities**

**Memory test on FPGA block memory.** Tests the BlockRAM inside the FPGA. On the DN2000k10, the BlockRAM is only in FPGA F.

**BAR memory range test.** Generic memory test that prompts the user for BAR number, starting address offset, DWORD count, and number of iterations. The user is also prompted if the program should stop if error occurs, or if the program should display any errors that occur. This allows for maximum flexibility when debugging a design with an oscilloscope, or debugging any memories or memory locations on your PCI bus. The memory test is very complete, performing a write then a read to every location, a read from every location, and then a read/write/read test to every location. All other memory test options listed in the memory menu are based on this generic memory test function.
Appendix A

Berg Connector Datasheets

Figure A-1 and Figure A-2 contain the schematics for the Berg 91403-003 Connector.

Figure A-3 through Figure A-5 contain the schematics for the Berg 91294-003 connector.
Figure A-1 Berg 91403-003 Datasheet Page 1 of 2
Figure A-2  Berg 91403-003 Datasheet Page 2 of 2
Figure A-3  Berg 91294-003 Datasheet Page 1 of 3
Figure A-4  Berg 91294-003 Datasheet Page 2 of 3
## Glossary and Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>µP</td>
<td>microprocessor</td>
</tr>
<tr>
<td>BAR</td>
<td>Base Address Register</td>
</tr>
<tr>
<td>BGA</td>
<td>ball grid array</td>
</tr>
<tr>
<td>BIOS</td>
<td>Basic Input/Output Services</td>
</tr>
<tr>
<td>CMOS</td>
<td>complementary metal-oxide semiconductor</td>
</tr>
<tr>
<td>CPLD</td>
<td>Complex Programmable Logic Device</td>
</tr>
<tr>
<td>CSF</td>
<td>Configuration Settings File</td>
</tr>
<tr>
<td>DSP</td>
<td>digital signal processing</td>
</tr>
<tr>
<td>EEPROM</td>
<td>Electrically Erasable PROM</td>
</tr>
<tr>
<td>EIA</td>
<td>Electronic Industries Association</td>
</tr>
<tr>
<td>ESD</td>
<td>Electro-Static Discharge</td>
</tr>
<tr>
<td>FAQ</td>
<td>frequently asked questions</td>
</tr>
<tr>
<td>FAT</td>
<td>file allocation table</td>
</tr>
<tr>
<td>FPGA</td>
<td>field programmable gate array</td>
</tr>
<tr>
<td>FT</td>
<td>flowthrough</td>
</tr>
<tr>
<td>HDL</td>
<td>Hardware Description Language</td>
</tr>
<tr>
<td>I/O</td>
<td>input/output</td>
</tr>
<tr>
<td>IP</td>
<td>intellectual property</td>
</tr>
<tr>
<td>LAB</td>
<td>Logic Array Blocks</td>
</tr>
<tr>
<td>LE</td>
<td>logic element</td>
</tr>
<tr>
<td>LSI</td>
<td>large scale integration</td>
</tr>
<tr>
<td>LUT</td>
<td>look-up table</td>
</tr>
<tr>
<td>LVCMOS</td>
<td>low voltage complementary metal-oxide semiconductor</td>
</tr>
<tr>
<td>LVDS</td>
<td>Low-Voltage Differential Signaling</td>
</tr>
<tr>
<td>LVTTL</td>
<td>low voltage transistor-transistor logic</td>
</tr>
<tr>
<td>MDR</td>
<td>Mini D Ribbon</td>
</tr>
<tr>
<td>PCI</td>
<td>peripheral component interconnect</td>
</tr>
<tr>
<td>PCI-X</td>
<td>peripheral component interconnect (extended)</td>
</tr>
<tr>
<td>PL</td>
<td>pipelined</td>
</tr>
<tr>
<td>PLL</td>
<td>phase lock loop</td>
</tr>
<tr>
<td>PNP</td>
<td>plug-and-play</td>
</tr>
<tr>
<td>PWB</td>
<td>printed wire board</td>
</tr>
<tr>
<td>RAM</td>
<td>random access memory</td>
</tr>
<tr>
<td>RBF</td>
<td>Raw Binary File</td>
</tr>
<tr>
<td>REGE</td>
<td>register enable</td>
</tr>
<tr>
<td>RISC</td>
<td>reduced instruction set computer</td>
</tr>
<tr>
<td>SDRAM</td>
<td>synchronous dynamic random access memory</td>
</tr>
<tr>
<td>SRAM</td>
<td>shadow random access memory</td>
</tr>
<tr>
<td>SSRAM</td>
<td>synchronous static random access memory</td>
</tr>
<tr>
<td>TTL</td>
<td>transistor-transistor logic</td>
</tr>
<tr>
<td>VCO</td>
<td>voltage controlled oscillator</td>
</tr>
<tr>
<td>VHDL</td>
<td>VHSIC Hardware Description Language</td>
</tr>
<tr>
<td>VREF</td>
<td>reference voltage</td>
</tr>
<tr>
<td>ZBT</td>
<td>zero-bus-turnaround</td>
</tr>
</tbody>
</table>
Index

Symbols

[C-F]DS 4-8, 4-10
µP. See microprocessor

Numerics

3.3Vaux 3-4
66MHZ_ENABLE 3-5

A

ACLK 4-2, 4-15, 7-8
ADSC# 5-7, 5-9
ADSP# 5-7, 5-9
ADV# 5-9
AETEST 9-1 to 9-14
Altera 2-1, 2-3 to 2-5, 2-7, 2-14 to 2-15, 2-20, 2-25, 4-14, 5-12
ASIC 2-1 to 2-2, 2-26, 4-1, 4-14, 5-1, 7-1, 7-6 to 7-7
ATmega128L 2-1, 2-9 to 2-10, 2-13, 2-17

B

B22 4-14
BA 5-9
BCLK 4-2, 4-15, 7-8
BCPUCLK 2-17
Berg A-1 to A-6
Berg Connectors A-1 to A-6
BIOS 9-1, 9-9
BlockRAM 7-1, 9-13 to 9-14
board termination voltage 2-8
Bridges2Silicon 2-17
BUFINA 4-3 to 4-4
BUFINB 4-2 to 4-5
BUP_CLK 2-15
bus bars 8-4
BWE# 5-9
BWx# 5-9

C

C380 4-4
C381 4-4
C382 4-4
C383 4-4
carry chains 2-4
CAS* 5-9
CCLK 4-8, 4-11 to 4-12, 4-15, 7-8
CE# 5-9
CE2# 5-9
CK 5-9, 5-12
CK# 5-12
CLK 2-15
CLKOUT 4-2 to 4-4
clock 2-7, 2-17, 4-1 to 4-2, 4-4, 4-14 to 5-1, 5-7 to 5-8, 5-12
arrays 2-26
buffer 2-15, 4-1 to 4-2, 4-5, 4-11, 6-2
buffer input 2-15
configuration 4-1
CPLD input 2-15
DDR clock select jumper 5-13
DDR PLL 5-13
DDR SDRAM 5-12
DDR select jumper 5-12
differential 4-11
distribution 4-1, 4-3
divider function 2-1, 2-17
division 2-17, 4-9
enable 2-4
feedback 4-9
FPGA 2-15
frequency 3-1, 4-2, 4-12, 4-15, 7-6
frequency multipliers 4-9
grid 2-15, 4-2 to 4-4, 4-11
header clocks 4-15
input 4-3, 4-5, 4-8 to 4-9, 4-11, 4-15
interfaces 2-9
inputs 4-12
multiplication 4-9
OE pin jumper settings 4-13
output 2-15, 4-8 to 4-9
output signals 2-8
outputs 4-2, 4-9, 4-14 to 4-15, 5-12 to 5-13
PCI 2-1
PCI_CLK 4-14
PLL 5-13
processor signal 2-17
PWB network 2-17
scheme 4-2
signal descriptions 4-2 to 4-3
signals 4-2, 4-4, 4-11 to 4-12, 4-14 to 4-15, 5-12 to 5-13, 7-6
Index (Continued)

skew  4-2, 4-9 to 4-10
settings 4-11
speed 2-7
SSRAM 5-1
status reporting 2-15
syncburst 5-7

Clock Utilities Menu  9-6
CLOCKA  4-2 to 4-4
CLOCKB  4-2 to 4-5
CONF_DONE/TDO  2-15, 2-17
configuration 2-1, 2-7, 2-15, 2-20, 2-23, 9-8 to 9-9
µP 2-9 to 2-10
bar 9-9
circuitry 4-2
clock 4-3 to 4-4
DDR SDRAM 5-12
debug 9-8
expansion 3-5
file 2-19, 2-23
file names 2-21
FPGA 2-1, 2-3, 2-9 to 2-10, 2-15, 2-17, 2-20 to 2-21, 2-23, 3-4
headers 2-17
JTAG 2-17
jumper settings 2-22
menu 2-23
PCI_CLK  4-14
SDRAM 5-11 to 5-12
serial 2-17
slave-serial 2-17
SmartMedia 2-17, 2-22, 2-24
stand-alone 6-3
status 2-23
tab 2-18
via Fast Passive Parallel 2-1, 2-17, 2-20
via fast passive parallel 2-18
via SmartMedia 2-1, 2-3
configuration space 9-2, 9-5, 9-8 to 9-9
CPLD 2-1, 2-3, 2-9, 2-13, 2-15, 2-17, 2-24, 4-2 to 4-4, 6-2
CPLD_CLK 2-15
CPLD_LED 2-15
CPLD_TCK 2-15 to 2-16
CPLD_TDI 2-15 to 2-16
CPLD_TDO 2-15 to 2-16
CPLD_TMS 2-15 to 2-16
CPLD_TRST 2-15
CPUCLK 2-17
CSF 2-8, 2-18
custom daughter cards 2-2

CY7B993V  4-2, 4-9 to 4-10, 4-12

D

D1  7-4
D2  7-4
D3  7-4
DATA0/TDI  2-15, 2-17
daughter cards 2-2, 6-5, 7-1 to 7-15
DCLK 2-17, 4-3, 4-8, 4-11, 4-15, 7-11
DCLK/TCK 2-15, 2-17
DCLK[7][R] 4-14 to 4-16
DDR clock select jumper 5-12 to 5-13
DDR SDRAM 2-3, 2-8, 4-14, 5-12
DIMM 5-1, 5-12, 6-2
DDR_CLK 5-12
DDR_CLKn 5-12
debug 2-1, 9-1 to 9-14
   analyzer-based 9-1 to 9-14
Device ID 2-20, 9-4, 9-7 to 9-9
differential LVDS pairs 7-1
DIMM 5-1, 5-9, 5-12, 6-2
DIP1_0 2-15
DIR 7-7
divider function 4-8 to 4-9
   settings 4-10
DLL 7-1
DN3000k10SD  2-2, 7-1 to 7-15
DN5000k10S  1-1, 3-1, 4-2
   +1.5V Power 6-3
   +2.5V Power 6-2
   +3.3V Power 6-2
   µP 2-9 to 2-10
ATmega128L 2-13 to 2-14
block diagram 2-2
clock scheme 4-1
CPLD 2-15
DDR SDRAM 5-12
description 2-2
dimensions 3-4
Fast Passive Parallel configuration 2-22
features 2-1
FPGA 2-3, 2-18, 2-21
FPGA configuration 2-9
I/O Issues 2-8
JTAG interface 2-10
LEDs 8-3
memories 5-1

I–2  The DINI Group
Index (Continued)

- oscillators 4-2, 4-12
- PCI 3-1
- PCI-X 3-1
- pipeline SSRAM 5-7
- power supplies 6-1
- reset 8-1
- ribbon cable 4-4
- SDRAM 5-9
- Serial Port 2-18 to 2-19
- SmartMedia 2-24
- SSRAM 5-1
- stand-alone operation 6-3 to 6-5
- stuffing options 2-3
- synthesis 2-26
- technical support 1-1
- timing devices 4-4 to 4-5
- DNPCIEXT-S3 3-1
- DOS 9-1 to 9-4, 9-6
- DOS extender 9-2
- drive power connector 6-3
- DS1 2-22, 4-10
- DS2 2-22
- DSP 2-4 to 2-5, 2-7
- dual-port 2-5

E

- ECLK 4-8, 4-11 to 4-12, 4-14 to 4-15, 5-1, 5-9, 7-8, 7-11
- EEPROM 2-10, 5-11 to 5-12
- embedded memory 2-1, 2-5, 2-26
- EP1S40 2-1, 2-3
- EP1S60 2-1, 2-3, 2-26
- EP1S80 2-1 to 2-5, 2-14, 2-24, 2-26
- EPM3256A 2-15
- ESD 1-1
- extender card 3-1
- external memories 2-2, 5-1

F

- Fast Passive Parallel 2-17 to 2-18, 2-20, 2-22 to 2-23
- FBDIS 4-8 to 4-9
- FBDS 4-8 to 4-9
- FBF0 4-8, 4-11
- FCLKOUT 4-8, 4-14
- feedback disable 4-8
- feedback output divider function 4-8
- feedback output phase function 4-8
- FLASH 2-1, 2-3, 2-9 to 2-10, 2-14, 9-6
- FlashPath 2-22, 2-24 to 2-25
- flowthrough 5-1, 5-6 to 5-8
- FPGA 1-2, 2-1, 2-3, 2-9 to 2-10, 2-14 to 2-15, 2-17 to 2-18, 2-20 to 2-24, 2-26 to 3-1, 3-4 to 3-6, 4-14 to 4-15, 5-9, 5-12 to 5-13, 6-2 to 6-3, 7-8, 9-1, 9-6, 9-13 to 9-14
- pins 3-2
- pin map 7-8
- FPGA D 2-14
- FPGA F 2-21 to 2-22
- FPGA_CDONEF 2-15
- FPGA_CEIf 2-15
- FPGA_CSif 2-15
- FPGA_D 2-15
- FPGA_DOF 2-15
- FPGA_DCLK 2-15
- FPGA_GRStn 2-15
- FPGA_IODONEF 2-15
- FPGA_MSEl 2-15
- FPGA_nCONFF 2-15
- FPGA_RDYnBUSYF 2-15
- frequency select 4-8
- FS 4-8 to 4-10, 4-12
- FT. See flowthrough

G

- GCLKOUT 4-14 to 4-16
- GND 2-16, 2-18, 4-3, 4-11, 7-5, 7-8 to 7-15
- GW# 5-9

H

- HDR_CLKOUT 4-8
- header clocks 4-8
- HyperTerminal 2-14, 2-19

I

- impedance 2-8
- input clock select 4-8
- INTB# 3-4
- INTC# 3-4
- INTD# 3-4
- Interconnect 7-8, 9-2
Index (Continued)

interconnect 9-2
Interconnects 7-8
INV1 4-12
INV2 4-8, 4-12

LTC1326 8-1
LUT 2-4
LVCMOS33 2-8
LVDS 7-1, 7-6
LVPECL 4-2, 4-11

J

J1 2-24, 7-7 to 7-8
J10 7-7
J16 7-7
J19 5-10 to 5-11
J2 5-12, 6-2
J3 5-9, 6-2
J8 4-2, 4-5, 7-5
J9 7-7
JP1 2-22
JP10 4-2, 4-5, 4-7
JP11 4-2, 4-5, 4-7
JP2 3-5
JP3 3-5 to 3-6
JP4 4-14, 5-12 to 5-13
JP5 4-14 to 4-15
JP6 4-2, 4-4, 4-11
JP7 5-12
JP8 4-7
JP9 4-2, 4-5, 4-7, 4-13
JTAG 2-9 to 2-10, 2-12 to 2-13, 2-17 to 2-18, 2-23 to 2-24, 3-4
cable 2-15, 2-17
chain 3-4
configuration 2-17
interface 2-2, 2-10, 2-12
programming 2-17 to 2-18, 2-23 to 2-24
signals 2-15 to 2-16, 3-4

M

M66EN 3-5 to 3-7
main configuration file. See main.txt
main.txt 2-18, 2-20 to 2-23
MDR 7-6
memories 2-2, 2-26, 4-14, 5-1, 9-13 to 9-14
microprocessor 2-1, 2-3, 2-9 to 2-10, 2-12, 2-15, 2-17, 6-2, 9-1
MODE 4-8
multiplexing 5-9
multiplication 2-7, 4-9
Multiplier 2-5, 2-7
multiplier 2-1, 2-4 to 2-5, 2-7, 2-25 to 2-26, 4-9, 9-2, 9-6
multiplier blocks 2-4
multiplier logic 2-7

N

nCONFIG/TMS 2-15, 2-17
nSTATUS 2-15, 2-17

O

OE# 7-7
oscillator 2-1, 2-17, 4-1 to 4-5, 4-9, 4-12 to 4-13, 6-2
oscilloscope 7-1, 9-8, 9-10, 9-13 to 9-14
output divider function 4-8, 4-10
output mode 4-8
output phase function 4-8
output-enable 7-7

P

P1 2-10, 2-12, 6-1, 6-3 to 6-5, 7-6
P2 2-18 to 2-19, 7-6
P3 2-17, 7-6
Index (Continued)

P4 2-15 to 2-16, 7-6
P5 2-13, 2-17, 7-6
P54 4-5
P58 2-24
P7 2-10, 2-12, 7-6
P8 4-15, 7-8
P9 4-15, 7-8
pattern generator 2-1 to 2-2
PCI 3-1, 3-4 to 3-7
   AETEST PCI menu 9-7
   board 6-5
   bus 3-1, 9-1, 9-14
   capability 3-6
   clock 2-1
   configuration 9-9
   connector 4-14, 6-1, 6-5
   controller 3-1
   debug 9-1
   device function 9-7 to 9-8
   device number 9-7
   edge connector 3-1, 3-3
   fingers 6-2, 6-5
   function number 9-7
   JTAG signals 3-4
   mechanical specifications 3-1
   memory 9-9 to 9-13
   power 3-1, 6-5
   power management 1-2
   present header 3-5
   present signals 3-5
   reference design 9-6
   signals 3-1 to 3-2
   slot 2-2, 3-1, 6-1, 6-3
   Special Interest Group 1-1
   specification 1-1 to 1-2, 2-8, 3-1, 3-4
   target design 2-2

PCI Menu 9-7

PCI_CLK 4-14 to 4-15

PCI-X 2-8, 3-4, 3-6
   capability 3-6
   capability header 3-6
   controller 3-1
   edge connector 3-1, 3-3
   frequency 3-7
   power 3-1
   present header 3-5
   present signals 3-5
   slot 2-2, 3-1
   specification 2-8

PCIXCAP 3-6 to 3-7

PCPLD_CLKOUT 2-15

PECL 4-4, 4-11
pipeline 2-7, 5-1, 5-6 to 5-8
PL See. pipeline
PLL 2-1, 4-1 to 4-2, 4-5, 4-9, 4-14 to 4-15, 5-12 to 5-13
PLL1A 4-2 to 4-3, 4-8
PLL1B 4-11
PLL1B_PRE 4-3
PLL1BN 4-8, 4-11
PLL1BN_PRE 4-3 to 4-4
PLL2B 4-11 to 4-12
PLL2B_PRE 4-3
PLL2BN 4-8, 4-11 to 4-12
PLL2BN_PRE 4-3
PLL5 4-14 to 4-15
PLLSEL2 4-8

PME- 3-6
polarity 4-13

power 2-13, 2-22 to 2-23, 3-1, 3-4, 3-6, 6-1, 6-3
   +1.5 V 6-3
   +2.5 V 6-2
   +3.3 V 6-2
   connector 2-1, 6-3
   distribution 3-1, 6-1
   limit 6-5
   rail 6-2
   rails 6-1 to 6-2, 6-4 to 6-5, 8-1
   rating 7-5
   reference 6-3
   requirements 3-5
   source 6-5
   sources 7-4
   specification 6-3
   supplies 6-1 to 6-5, 8-1
   supply 2-24, 5-11 to 5-12, 6-1 to 6-3, 6-5, 7-4 to 7-5
   switch 9-9

power distribution ?? to 6-5

Power Management Enable. See PME-

power supplies 6-1 to 6-5, 8-1

power supply 2-24, 5-11 to 5-12, 6-1 to 6-3, 6-5, 7-4 to 7-5

power-up 9-1

prototyping board 7-1, 7-6 to 7-7

PWB 1-1, 2-1 to 2-3, 2-17, 2-24, 3-1

PWR_RSTn 2-15

Q

Quartus 2-18, 2-20 to 2-21, 2-25 to 2-26
Index

Index (Continued)

R

R1  5-6
R128  5-6
R129  5-6
R130
  R3  5-6
R138  5-6
R140  5-6
R142  5-6
R198  4-4
R2  5-6, 7-8
R206  4-4
R207  4-4
R209  4-4
R210  4-4
R211  4-4
R212  4-4
R214  4-4
R215  5-6
R216  5-6
R217  5-11
R218  5-11
R59  5-12
R70  5-6
RAS*  5-9
RB[C-F]F  4-8
reference design  2-18, 2-22, 9-6
REGE  5-12
regulator  2-1, 3-1, 6-2
reset  2-4, 2-14, 8-1
  button  2-23
  functionality  8-1
  schemes  2-15, 8-1 to 8-2
ROBO_LOCK1  2-15
ROBO_LOCK2  2-15
Roboclock  2-1, 4-1 to 4-7, 4-10 to 4-12, 4-14 to 4-16, 5-9, 6-2
RoboclockII  5-9
Roboclocks  6-2
RS232  2-1, 2-3, 2-18 to 2-19
RST#  3-4, 9-1

S

S1  2-14, 2-23, 4-10
SDRAM  2-3, 4-14, 5-9, 5-11 to 5-12, 6-2, 6-5
  bus signals  5-10 to 5-11

DIMM  5-1, 5-9
EEPROM  5-11
test  9-2, 9-13

Select/i/O  7-1
serial port  2-13 to 2-14, 2-18 to 2-20, 2-22
  location  2-19

Signals

  µP
    BUP_CLK  2-15
    SRAM_CSn  2-15
    UP_ALE  2-15
    UP_RDn  2-15
    UP_WRn  2-15
    UPAD  2-15
    UPPADDR  2-15
   3.3Vaux  3-4
   66MHZ_ENABLE  3-5
   ACLK  4-2, 4-15, 7-8
   ADSC#  5-7, 5-9
   ADSP#  5-7, 5-9
   ADV#  5-9
   BA  5-9
   BCLK  4-2, 4-15, 7-8
   BCPUCCLK  2-17
   BUFINA  4-3 to 4-4
   BUFINB  4-2 to 4-5
   BUP_CLK  2-15
   BWE#  5-9
   BWx#  5-9
   CAS*  5-9
   CCLK  4-8, 4-11 to 4-12, 4-15, 7-8
   CE#  5-9
   CE2#  5-9
   CK  5-9, 5-12
   CK#  5-12
   CLK  2-15
   CLKOUT  4-2 to 4-4

  Clock Signals
   ACLK  4-2, 4-15, 7-8
   BCLK  4-2, 4-15, 7-8
   BCPUCCLK  2-17
   BUFINA  4-3 to 4-4
   BUFINB  4-2 to 4-5
   CCLK  4-8, 4-11 to 4-12, 4-15, 7-8
   CLK  2-15
   CLKOUT  4-2 to 4-4
   CLOCKA  4-2 to 4-4
   CLOCKB  4-2 to 4-5
   CPLD_CLK  2-15
   CPUCCLK  2-17
DCLK  2-17, 4-3, 4-8, 4-11, 4-15, 7-11
DCLK[7](R)  4-14 to 4-16
ECLK  4-8, 4-11 to 4-12, 4-14 to 4-15, 5-1, 5-9, 7-8, 7-11
FCLKOUT  4-8, 4-14
GCLKOUT  4-14 to 4-16
HDR_CLKOUT  4-8
INV1  4-12
INV2  4-8, 4-12
PCI_CLK  4-14 to 4-15
PCPLD_CLKOUT  2-15
PLL1A  4-2 to 4-3, 4-8
PLL1B  4-11
PLL1B_PRE  4-3
PLL1BN  4-8, 4-11
PLL1BN_PRE  4-3 to 4-4
PLL2B  4-11 to 4-12
PLL2B_PRE  4-3
PLL2BN  4-8, 4-11 to 4-12
PLL2BN_PRE  4-3
PLLSEL2  4-8
RB[C-F]F  4-8
CLOCKA  4-2 to 4-4
CLOCKB  4-2 to 4-5
CONF_DONE/TDO  2-15, 2-17
CPLD_CLK  2-15
CPLD_LED  2-15
CPLD_TCK  2-15 to 2-16
CPLD_TDI  2-15 to 2-16
CPLD_TDO  2-15 to 2-16
CPLD_TMS  2-15 to 2-16
CPLD_TRST  2-15
CPUCLK  2-17
DATA0/TDI  2-15, 2-17
DCLK  2-17, 4-3, 4-8, 4-11, 4-15, 7-11
DCLK/TCK  2-15, 2-17
DCLK[7](R)  4-14 to 4-16
DDR SDRAM
   CK#  5-12
   DDR_CLK  5-12
   DDR_CLKn  5-12
DDR_CLK  5-12
DDR_CLKn  5-12
DIP1_0  2-15
ECLK  4-8, 4-11 to 4-12, 4-14 to 4-15, 5-1, 5-9, 7-8, 7-11
FBDIS  4-8 to 4-9
FBDS  4-8 to 4-9
FBF0  4-8, 4-11
FCLKOUT  4-8, 4-14
FPGA  2-15
   CPLD_TMS  2-15
   DIP1_0  2-15
   FPGA_CDONEF  2-15
   FPGA_CEnF  2-15
   FPGA_CSnF  2-15
FPGA_D 2-15
FPGA_DCLK 2-15
FPGA_IODONEF 2-15
FPGA_MSEL 2-15
FPGA_nCONFF 2-15
FPGA_RDYnBUSYF 2-15
FPGA_CDONEF 2-15
FPGA_CEnF 2-15
FPGA_CSnF 2-15
FPGA_D 2-15
FPGA_D0F 2-15
FPGA_DCLK 2-15
FPGA_GRSTn 2-15
FPGA_IODONEF 2-15
FPGA_MSEL 2-15
FPGA_nCONFF 2-15
FPGA_RDYnBUSYF 2-15
FS 4-8 to 4-10, 4-12
GCLKOUT 4-14 to 4-16
GND 2-16, 2-18, 4-3, 4-11, 7-5, 7-8 to 7-15
GW# 5-9
HDR_CLKOUT 4-8
INTB# 3-4
INTC# 3-4
INTD# 3-4
INV1 4-12
INV2 4-8, 4-12
JTAG
  CONF_DONE/TDO 2-15, 2-17
  CPLD_TCK 2-15 to 2-16
  CPLD_TDI 2-15 to 2-16
  CPLD_TDO 2-15 to 2-16
  CPLD_TMS 2-15 to 2-16
  CPLD_TRST 2-15
  DATA0/TDI 2-15, 2-17
  DCLK/TCK 2-15, 2-17
  nCONFIG/TMS 2-15, 2-17
  nSTATUS 2-15, 2-17
  TCK 2-15 to 2-17, 3-4
  TDI 2-15 to 2-17, 3-4
  TDO 2-15 to 2-17, 3-4
  TMS 2-15 to 2-17, 3-4
  TRST# 3-4
LD# 5-9
LOCK# 3-1
MODE 4-8
nCONFIG/TMS 2-15, 2-17
nSTATUS 2-15, 2-17
OE# 7-7
PCI
  3.3Vaux 3-4
  INTB# 3-4
  INTC# 3-4
  INTD# 3-4
LOCK#  3-1

PCI Signals
  CE#  5-9
  CE2#  5-9
PCI_CLK  4-14 to 4-15
PCPLD_CLKOUT  2-15
PLL1A  4-2 to 4-3, 4-8
PLL1B  4-11
PLL1B_PRE  4-3
PLL1BN  4-8, 4-11
PLL1BN_PRE  4-3 to 4-4
PLL2B  4-11 to 4-12
PLL2B_PRE  4-3
PLL2BN  4-8, 4-11 to 4-12
PLL2BN_PRE  4-3
PLL2B  4-11 to 4-12
PLL2B_PRE  4-3
PWR_RSTn  2-15
RAS*  5-9
RB[C-F]F  4-8
REGE  5-12
Reset
  FPGA_GRSTn  2-15
  PWR_RSTn  2-15
ROBO_LOCK1  2-15
ROBO_LOCK2  2-15
RST#  3-4, 9-1
SDRAM
  BA  5-9
  CAS*  5-9
  CK  5-9, 5-12
  RAS*  5-9
  REGE  5-12
  WP  5-11 to 5-12
SM_ALE  2-15
SM_CEn  2-15
SM_CLE  2-15
SM_D  2-15
SM_RDYBUSYn  2-15
SM_REn  2-15
SM_WEEn  2-15
SmartMedia
  SM_ALE  2-15
  SM_CEn  2-15
  SM_CLE  2-15
  SM_D  2-15
  SM_RDYBUSYn  2-15
  SM_REn  2-15
  SM_WEEn  2-15
  SM_WPn  2-15
SP_WPn  2-15
SRAM_CSn  2-15
TCK  2-15 to 2-17, 3-4
TDI  2-15 to 2-17, 3-4
TDO  2-15 to 2-17, 3-4
TMS 2-15 to 2-17, 3-4
TRST# 3-4
UP_ALE 2-15
UP_WRn 2-15
UPAD 2-15
UPPADDR 2-15
VCC 2-18
WP 5-11 to 5-12
SM_ALE 2-15
SM_CEn 2-15
SM_CLE 2-15
SM_D 2-15
SM_RDYBUSYn 2-15
SM_REn 2-15
SM_WEn 2-15
SM_WPn 2-15
SmartMedia 2-1, 2-3, 2-9, 2-14 to 2-15, 2-17, 2-22 to 2-25, 3-4, 9-1
speed 3-6, 5-7
clock 2-7
clock buffer 4-2
configuration 2-1
grade 2-3, 2-26 to 3-1
interface 2-15
LVDS 7-6
PWB 3-1
SRAM 2-9, 2-15, 5-1, 6-2
SRAM_CSn 2-15
SSRAM 2-3, 4-14, 5-1, 5-6 to 5-8, 6-2
bus signals 5-2 to 5-5
test 9-2, 9-13
timing 5-8 to 5-9
startup 9-4 to 9-5
static electricity 1-1
Stratix 1-2, 2-1, 2-3 to 2-5, 2-8 to 2-9, 2-14, 2-17, 2-24, 2-26 to 3-1, 4-14 to 4-15, 5-12, 6-2 to 6-3, 7-1
switching regulator 3-1, 6-2
Syncburst 5-6 to 5-9
Synopsys 2-25 to 2-26
Synplicity 2-2, 2-17, 2-25
synthesis 2-25 to 2-26
tools 2-7, 2-25 to 2-26

target design 2-2
TCK 2-15 to 2-17, 3-4
TDI 2-15 to 2-17, 3-4
TDO 2-15 to 2-17, 3-4
technical support 1-1
terminator technology 2-8
TMS 2-15 to 2-17, 3-4
TP13 3-6
TRST# 3-4
U

U1  7-7
U10  5-3, 6-2
U11  2-3, 4-14, 6-2 to 6-3, 7-8
U12  6-2
U13  5-1, 5-5, 6-2
U14  4-2, 4-8, 4-14, 6-2
U15  4-2, 4-8, 6-2
U16  6-2
U17  6-2
U18  6-2
U2  2-18, 7-6 to 7-7
U3  2-9, 7-6 to 7-7
U4  6-2, 7-4 to 7-5
U6  5-1, 6-2
U7  6-2
U8  2-9, 2-13, 5-1, 5-4, 6-2
U9  5-1 to 5-2, 6-2
UP_ALE  2-15
UP_RDn
Signals
   UP_RDn  2-15
UP_WRn  2-15
UPAD  2-15
UPPADDR  2-15

V

VCC  2-18
Vendor ID  9-7 to 9-9
verbose level  2-20 to 2-23
Verilog  1-2, 2-2, 2-7, 2-15, 2-18, 2-25
VHDL  1-2, 2-2, 2-7, 2-25
Virtex  2-14
voltage  2-8, 2-10, 4-11, 6-2, 7-5
   board termination  2-8
differential  2-9, 4-11
   reference  2-8, 6-2 to 6-3
   sources  7-5
   supply  6-2 to 6-3
termination  2-8
   translation  2-18
voltage controlled oscillator  4-9

W

WVP  5-11 to 5-12
X

X1  2-17, 4-1 to 4-2, 6-2
X2  4-1 to 4-2, 4-12 to 4-13, 6-2
X3  4-1 to 4-2, 4-12 to 4-13, 6-2
Xilinx  2-9

Z

ZBT  5-6 to 5-9