DYNAMIC ENGINEERING

150 DuBois St., Suite C Santa Cruz, CA 95060
(831) 457-8891 **Fax** (831) 457-4793
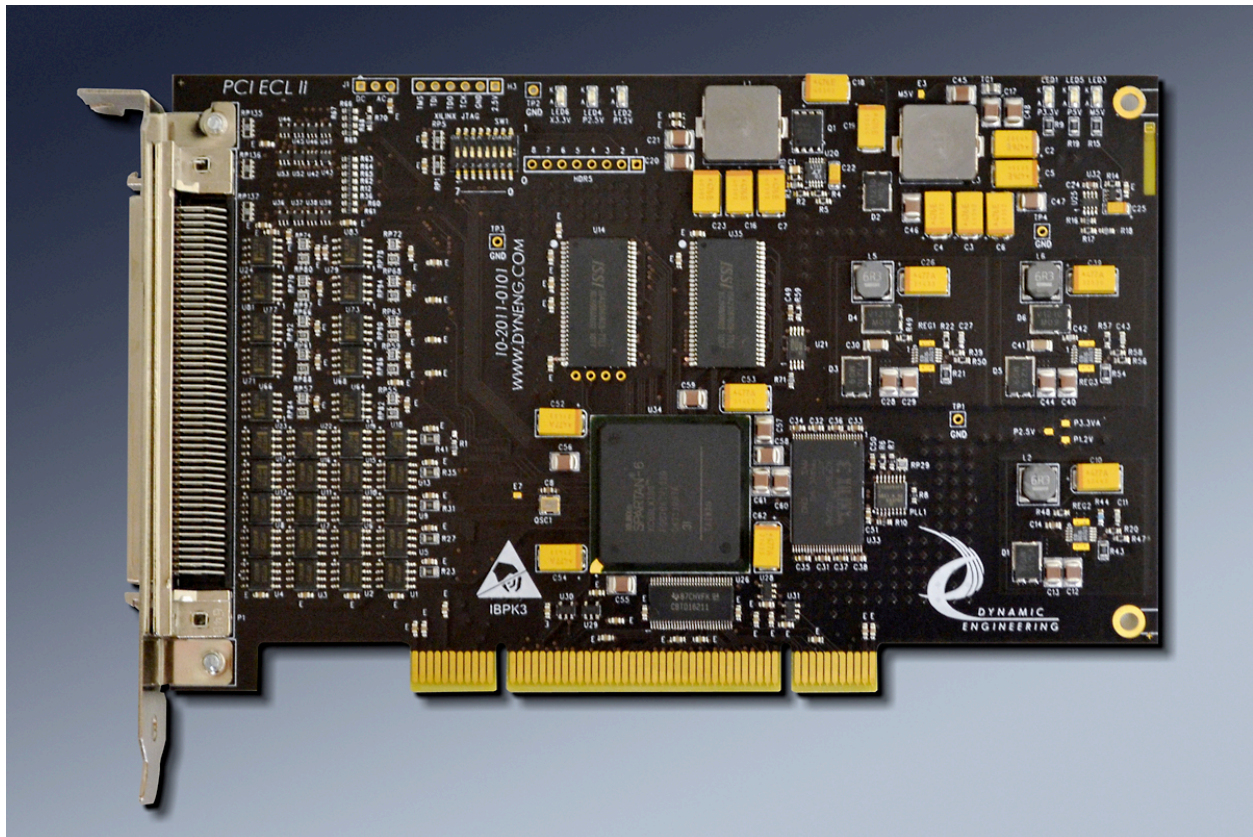http://www.dyneng.com sales@dyneng.com
Est. 1988

User Manual

PCI-NECL2-STE3

Bidirectional NECL I/O with DMA
12 bit GPIO port [TTL]

Revision A1
Corresponding Hardware: Revision A/B
10-2011-0101/2
Corresponding Firmware: Revision A/B
11/9/15

PCI-NECL2-STE3
PCI based Bidirectional DMA
With NECL and TTL I/O

Dynamic Engineering
150 DuBois St., Suite C
Santa Cruz, CA 95060
(831) 457-8891
FAX: (831) 457-4793

Table of Contents

List of Figures

Product Description Part I

Naming: PCI-ECL-II is the base design and can be outfit with NECL and other ECL family components. The specific versions of the implementation will have the "ECL" changed to "NECL" etc. plus an extension added for the VHDL set and any other alterations made to the board.

STE3 uses the classic negative ECL or NECL plus a specific FPGA file set designed to provide byte wide data transfers at a programmable IO rate. A receiver is also designed in which can operate in parallel with the transmitter. One of the two available channels is tied to the IO [channel 0]. The second channel has memory and loop-back capabilities but no IO in this implementation.

PCI-NECL2-STE3 is part of the PCI Compatible family of modular I/O components. The PCI-NECL2-STE3 provides a Spartan 6 FPGA, along with 40 ECL [NECL] and 12 TTL I/O lines, a programmable PLL and FIFO support with full bidirectional DMA capabilities in a half-length single slot card.

With NECL2 the FIFO support has been greatly expanded. The FIFO's are now based on SDRAM devices with programmable offsets to allow the memory to be partitioned into 1 or 2 sections per channel. The sections can be the same size or programmed for different RX and TX sizes. The total memory per channel is 32 Mbytes.

The other main difference from the NECL design is the combined PCI and ECL interface within the Spartan 6 FPGA. Better levels of integration mean higher through-put and more DMA channels to work with.

The PCI bus implementation is 32 bits at 33 MHz, universal voltage. The hardware supports direct access software controlled read/write access to all locations plus DMA support to the high bandwidth FIFO ports. The hardware is optimized for simultaneous bidirectional DMA access to support the high data rates available on the PCI-NECL2-STE3.

The Cypress 22393 PLL is handy for creating user specific frequencies with which to operate the state-machines and I/O. The driver supports programming the PLL over a serial I²C bus. Four clocks are received from the PLL into the FPGA - see figure 1 below. The clock routing uses matched lengths to provide in-phase references should they be necessary in your design. The FPGA clock features provide further clock functionality. STE3 uses the on-board oscillator (50 MHz) as the reference for the PLL and PLLA is used to set the transmit frequency when used for loop-back. STE3.jed can be used to set this frequency.

Cypress has a utility available for calculating the frequency control words for the PLL. http://www.dyneng.com/CyberClocks.zip is the URL for the Cypress software used to calculate the PLL programming words. The PLL responds to one of two device ids (only one works). As part of our ATP our software determines the address of the PLL. The software is part of the engineering kit and can be ported to your application.

FIGURE 1   PCI-NECL2-STE3 PLL

An 8-bit "dip switch" is provided on the PCI-NECL2-STE3 for user-defined purposes. The switch configuration is readable via a memory-mapped I/O port. We envision the switch being used for software configuration control, PCI board identification or test purposes.

LED's are provided on the board to indicate that the 3.3, 2.5, 1.2, and -5V regulators are operating properly plus the PCI 5V and 3.3V. Local regulation is provided for 3.3, 2.5, 1.2, and -5V volts. The 3.3V regulator is used for the Spartan 6 due to a lower Vmax than the PCI supplied 3.3V. Each of the voltages other than the PCI 3.3V and 5V have voltage comparators controlling the LED's. For example, when the –5V LED is illuminated it is within tolerance.

FIGURE 2   PCI-NECL2-STE3 BLOCK DIAGRAM

PCI-NECL2-STE3 has both ECL and TTL I/O interfaced to the D100 connector. There is circuitry for 20 NECL inputs and 20 NECL outputs on the board as indicated in figure 2 above. STE3 uses the first 10 IO for transmit and receive plus number 18 for a reference clock. 7-0 = data, 8 = enable, 9 = clock.

12 TTL I/O lines are supported with open drain drivers with pull-ups. The LVC1G125 open drain drivers have 64+ mA of sink capability. The IO can be referenced to 3.3 or 5V. Please note the '125's are 3.3V devices and when enabled and driven "hi" the output will be pulled down to 3.3V. To achieve 5V on the IO, the driver is disabled allowing the resistor to pull to the 5V level. The reference can also be set to 3.3V in which case the driven [hi] and off states are the same. Each IO can be set to drive or be an input independently. STE3 has a GPIO port for this purpose.
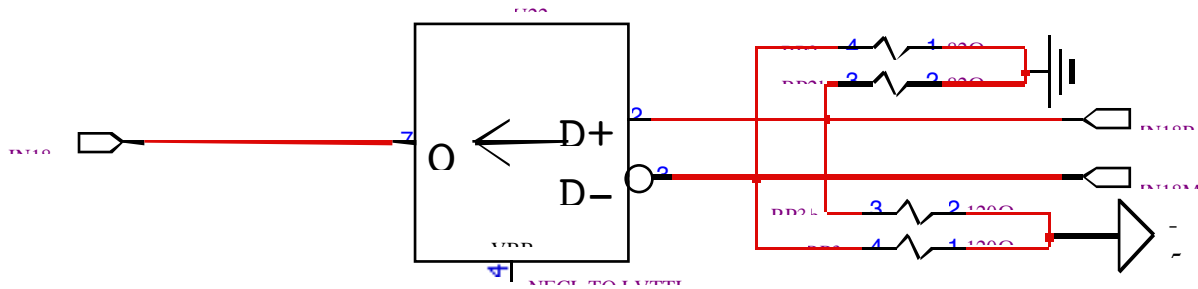
FIGURE 3   PCI-NECL2-STE3 INPUT TERMINATION

The ECL inputs are terminated with 50Ω to -2V using a parallel 82Ω / 120Ω equivalent circuit as shown in figure 3 above.   The ECL input lines are routed as differential pairs with matched lengths and impedance control [trace and space]. The lengths are matched from the connector edge to the Xilinx ball to allow for high-speed low-skew operation.
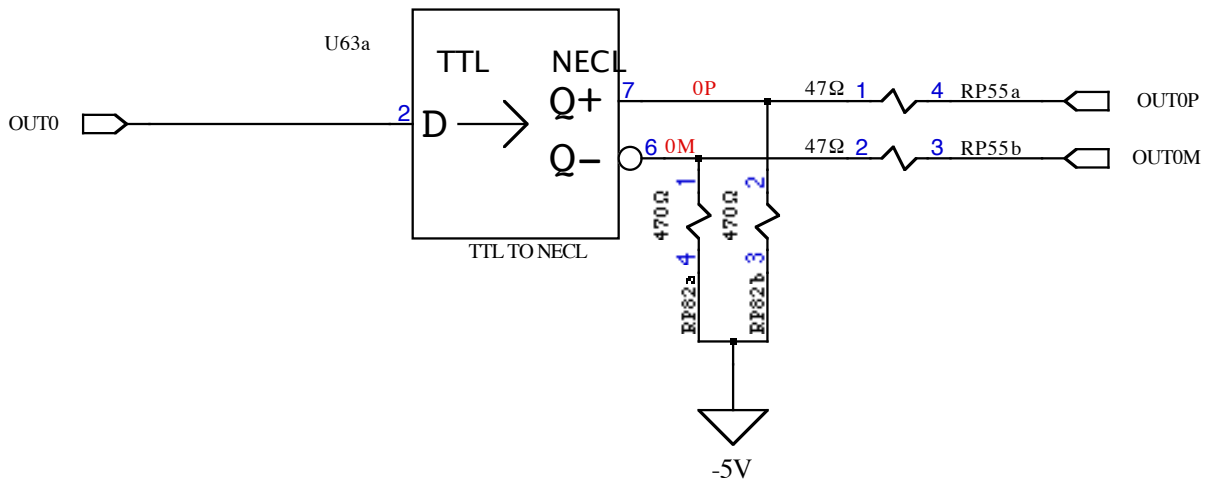
FIGURE 4   PCI-NECL2-STE3 OUTPUT TERMINATION

The ECL outputs are terminated with 470Ω terminations to –5V. The ECL output lines are routed as differential pairs with matched lengths and impedance control [trace and space]. The lengths are matched from the Xilinx ball to the connector edge to allow for high-speed low-skew operation.

The Xilinx FPGA is re-configurable by loading a new programming file into the FLASH storage device. The file can be generated with the standard Xilinx design software. The standard Xilinx Parallel JTAG cable is connected to the on-board header to program the FLASH using the Xilinx iMPACT software.
The engineering kit includes a cable and the HDEterm100. The HDEterm100 serves as

a breakout from the cable to screw terminal block. The HDEterm100 has matched length, differential routing and several termination options that can be installed. For more information on the HDEterm100 please visit the web page.
http://www.dyneng.com/HDEterm100.html

In addition a "cross over" cable is available which interconnects the ECL IN and OUT IO 1:1 to allow two boards to be cross-connected for data transfer.

Product Description Part II

A wide range of interfaces and protocols can be implemented with the PCI-ECL-2; UART, Manchester encoding, serial or parallel, ECL/NECL and/or TTL. The interfaces can be created using the hardware and development tools provided with the PCI-ECL along with the Xilinx software.

Once your requirements are known the design can be implemented with VHDL, Verilog, or schematics and compiled with the Xilinx design software. The output file can then be programmed into the Xilinx Flash PROM on the PCI-ECL. Because the Flash PROM is reprogrammable, your design can be implemented in phases. You can experiment and test out concepts and partial implementations during the design phase or perhaps simulate other hardware that needs to be implemented.

As an example consider a parallel interface with 16 data lines and 3 control signals. The PCI-ECL has 40 ECL differential I/O, so there is enough I/O for a full duplex implementation. The parallel channel would be supported with the 8Mx32 external FIFO plus any internal FIFO's that were instantiated out of block RAM. The FPGA is a Spartan 6 LX100 industrial temp part in the –3 speed grade. The PCI interface and SDRAM controller only occupy a small percentage of the device. STE3 including everything uses approximately 19%. There is of room for more complex data formatting requirements.

For systems with an external reference clock, ECL input bits 19, 18, 17, 16 and 9 are received by the FPGA on a long line pins. They can be routed through a Digital Clock Manager to create a low skew clock distribution based on an external reference.

The data flow for transmission is Host memory transferred into the transmit FIFO via DMA transfers. The user state machine reads the data from the FIFO on the output side and applies the user protocol before transmitting. On the receive side the data flows into the FPGA, is processed to convert to a format suitable for storing, and then loaded into the receive FIFO. The data is read from the receive FIFO by the PCI control state-machine and moved into the host memory via DMA transfers.

The path to the Transmit or from the Receive local FIFO's is indirect.   In the Tx path the PCI DMA transfers to the SDRAM TX DMA FIFO within the SDRAM group. The FIFO is read by the SDRAM controller and the data moved to the SDRAM. The data stored in the SDRAM is moved to the output FIFO within the SDRAM group. The SDRAM controller uses the state of the FIFO's and SDRAM to determine when to move data or not. The Tx state-machine reads from the Tx IO FIFO in the SDRAM group to interface with the Tx encoding for that design. In STE3 a secondary TX FIFO is used to convert from the 32 bits presented by the SDRAM interface and the 8 bits needed at the IO plus the 33 MHz clock rate at the SDRAM FIFO and the STE3 rate at the IO [29.4 MHz].

The receive path is similar with the receiver controller moving data to the SDRAM RX IO FIFO. The RX IO FIFO is read by the SDRAM controller and moved to the SDRAM. From the SDRAM the data is loaded to the SDRAM RX DMA FIFO for transfer to the host memory.

The SDRAM controller block has FIFO counts available to allow the TX and RX SM to properly interact with the FIFO's built into the SDRAM controller.

The SDRAM controller has bypass transfer units which if enabled move data directly from the SDRAM TX DMA FIFO to the SDRAM TX IO FIFO and similarly from the SDRAM RX IO to the SDRAM RX DMA FIFO's. Each bypass is independent.   This allows all of the SDRAM to be tied to one port or the other. The engineering kit has an example of using TX or RX only operation as well as a split definition with each TX and RX getting 1/2 of the memory.

The SDRAM controller has an arbitration unit which determines on a round-robin basis which channel to service next. When an operation completes the arbiter looks at each of the other options before checking the same channel again. The logic has skip-ahead to save states when a channel is disabled.

The TX and RX have load and unload operations with separate enables to allow the software to configure TX only, RX only, TX load before start-up etc. The design includes a FIFO mode and a retransmit mode. The retransmit mode has the ability to send a header, body, and tail. The FIFO mode makes the SDRAM act like a FIFO and is the mode used in the STE3. Please see the programming section for more details.

The full bandwidth of the PCI bus is utilized during DMA transfers. There is some overhead on the PCI bus side, which will limit the actual sustainable transfer rate somewhat compared to the theoretical limit. Looking at the other side of the equation; if we assume parallel data with 1 channel operating at 35 Mbytes/second in each direction, this creates a total of 70 Mbytes/second on the PCI bus – approximately 53% loading of the theoretical maximum.

Using the same example and looking at the storage, one can see that the OS can "go away" for 4M words x 4 bytes/word / 70Mbytes/sec => 239.6 mS without receive data over-running the FIFO when used for receive or under-running the FIFO when used for transmit. With Windows® and other high level OS based system the OS can neglect data movement due to other requirements - dealing with the keyboard or HDD for example. Having adequate storage can make a big difference in system performance.

Current Feature List

• User Definable Xilinx Spartan 6 series FPGA

- Bidirectional DMA capable 32/33 PCI bus interface
- PLL with 4 programmable clocks tied to FPGA
- 8M x 32 Data buffer per channel [RX/TX pair]
- 20 ECL Outputs
- 20 ECL Inputs
- 12 TTL I/O
- 8 position "DIP Switch"
- Power LED's
- On-going development with a "FLASH" program

As Dynamic Engineering adds features to the hardware we will update the PCI-ECL-II page on the Dynamic Engineering website. If you want some of the new features, and have already purchased hardware, we will support you with a FLASH update. We will reprogram the FLASH on your board for you or if you have the engineering kit and your own download cable, send you the new bit file. If you are interested please contact sales@dyneng.com for arrangements.

The basic PCI identifying information will not change with the updates. The revision field will change to allow configuration control. STE3 is design 1 for the PCI-ECL-II. The revision of the FLASH is 1 for revision 01 fabs and 2 for revision 2 fabs.

Revision 2 is the current fab for PCI-ECL-II. 10-2011-0102 is the complete fab number.

Revision 2 updates include integrating the rework required on Revision 1 and reassigning the ECL input bit 9 FPGA ball assignment to better support byte wide data transfers with a local clock input.   STE3 has a jumper to accomplish this on revision 1.

Programming

PCI-NECL2-STE3 is tested in a Windows® XP environment. We use our driver to support our test software. Please consider purchasing the engineering kit for the PCI-NECL2-STE3; the kit comes with a PDF of the schematic, cable and breakout. The Windows® driver is included with the purchase of the board. Please specify which driver you want as we plan to add Linux and Windows 7® support.

The internal registers for the Xilinx are defined in the following pages.

The architecture is designed as a hierarchy with a Base and Channels. There are two channels implemented for STE3. Channel 0 has is tied to the NECL IO and is the "active" channel. Channel 1 has a separate memory and can be accessed. No IO is connected to channel 1 at this time.

The address maps are offsets from the base address. In the case of the Base the offset is the system assigned memory space address. The channels are shown relative to 0x00 for the channel . The channels are offset from the base and each other as shown.

Feature	Address offset	Decodes
Base	0x00	19-0
Channel 0	0x50	39-0 (59-20)
Channel 1	0xF0	39-0 (99-60)

The VendorID is 0x10EE
The CardID for STE3 is 0x0043

The address map provided is for the local decoding performed within PCI-NECL2-STE3. The addresses are all offsets from a base address. The base address and interrupt level are provided by the host.

The host system will search the PCI bus to find the assets installed during power-on initialization and allocate memory and interrupt resources.   PCIView or other third party utilities can be useful to view your system configuration.

Once the initialization process has occurred and the system has assigned an address range to the PCI-NECL2-STE3 card, the software will need to determine what the address space is. We refer to this address as base0 in our software.

The next step is to initialize the PCI-NECL2-STE3. The local Xilinx registers need to be configured.

Address Map

Base Addresses
STE3_BASE_PLL	0x00000000	// 0 Base control register
STE3_BASE_SWITCH	0x00000004	// 1 Switch, Xilinx Revision, Design Number
STE3_BASE_STATUS	0x00000008	// 2 Base Status
STE3_BASE_GPIO_DIR	0x00000010	// 4 GPIO Direction register
STE3_BASE_GPIO_DATA	0x00000014	// 5 GPIO Data register
STE3_BASE_GPIO_IO	0x00000018	// 6 GPIO Data in: read only

FIGURE 5   PCI-NECL2-STE3 XILINX BASE ADDRESS MAP

Channel Addresses
STE3_CHAN_CNTRL	0x00000000	// 0 General Control Register
STE3_CHAN_STATUS	0x00000004	// 1 Interrupt Status Port Read Only
STE3_CHAN_INT_CLEAR	0x00000004	// 1 Interrupt Clear Port Write Only
STE3_CHAN_WR_DMA_PNTR	0x00000008	// 2 TX Wr DMA Port physical PCI address register
STE3_CHAN_TX_FIFO_COUNT	0x00000008	// 2 TX FIFO data count – read only
STE3_CHAN_RD_DMA_PNTR	0x0000000C	// 3 RX Read DMA Port Physical PCI address reg
STE3_CHAN_RX_FIFO_CNT	0x0000000C	// 3 RX FIFO data count - read
STE3_CHAN_FIFO	0x00000010	// 4 Single Word Access port for TX, RX FIFO
STE3_CHAN_TX_AMT_LVL	0x00000014	// 5 TX almost empty level register R/W
STE3_CHAN_RX_AFL_LVL	0x00000018	// 6 RX Almost Full level register R/W
STE3_CHAN_TX	0x0000001C	// 7 TX Control Register R/W
STE3_CHAN_TX_READY_REG	0x0000002C	// 11 Num of bytes req. in final FIFO before TX
STE3_CHAN_RX	0x00000034	// 13 RX Control Register R/W
STE3_CHAN_RX_MEM_A	0x00000050	// 20 Receive address A
STE3_CHAN_TX_MEM_A	0x00000054	// 21 Transmit address A
STE3_CHAN_SET_MEM	0x00000058	// 22 Spare – reserved for direct memory set
STE3_CHAN_RX_MEM_B	0x0000005C	// 23 Receive address B
STE3_CHAN_TX_MEM_B	0x00000060	// 24 Transmit address B
STE3_CHAN_RX_MEM_C	0x00000064	// 25 Receive address C
STE3_CHAN_TX_MEM_C	0x00000068	// 26 Transmit address C
STE3_CHAN_RX_MEM_D	0x0000006C	// 27 Receive address D
STE3_CHAN_TX_MEM_D	0x00000070	// 28 Transmit address D
STE3_CHAN_RX_LOOP_CNT	0x00000074	// 28 Receive Body Loop Count 7-0
STE3_CHAN_TX_LOOP_CNT	0x00000078	// 28 Transmit Body Loop Count 7-0
STE3_CHAN_RX_SDRAM_CMD	0x0000007C	// 28 Receive SDRAM Control Register 7-0
STE3_CHAN_TX_SDRAM_CMD	0x00000080	// 28 Transmit SDRAM Control Register 7-0

FIGURE 6   PCI-NECL2-STE3 XILINX CHANNEL ADDRESS MAP

Programming

Programming the PCI-NECL2-STE3 requires only the ability to read and write data in the host's PCI space.

Once the initialization process has occurred, and the system has assigned addresses to the PCI-NECL2-STE3 card the software will need to determine what the address space is for the PCI interface [BAR0]. The offsets in the address tables are relative to the system assigned BAR0 base address.

The next step is to initialize PCI-NECL2-STE3. The PLL will need to be programmed to use the loop-back function. The Cypress CyberClocks software can be used to create new .JED files if desired. PLLA should be set to the transmit reference frequency output by the transmitter.

The driver comes with a .JED file prepared. "STE3.JED" The driver has a utility to load the PLL and read back. The reference application software has an example of the use of PLL programming. The reference application software also includes XLATE.c which converts the .JED file from the CyberClocks tool to an array that can be programmed into the PLL.

For Windows™ (WIN32) and Linux systems the Dynamic Drivers can be used. The driver will take care of finding the hardware and provide an easy to use mechanism to program the hardware. The Driver comes with reference software showing how to use the card and reference frequency files to allow the user to duplicate the test set-up used in manufacturing at Dynamic Engineering. Using simple, known to work routines is a good way to get acquainted with new hardware.

To use the STE3 specific functions the Channel Control, and PLL interface plus DMA will need to be programmed. To use DMA, memory space from the system should be allocated and the link list stored into memory. The location of the link list is written to STE3 to start the DMA. Please refer to the Burst IN and Burst Out register discussions.

DMA should be set-up before starting the channel port function. For transmission this will result in the FIFO being full or close to it when the transfer is started or at least the Packet loaded if shorter than the FIFO size. For reception it means that the FIFO is under HW control and the delay from starting reception to starting DMA won't cause an overflow condition.

DMA can be programmed with a specific length. The length can be as long as you want within standard memory limitations. At the end of the DMA transfer the Host will receive an interrupt. The receiver can be stopped and the FIFO reset to clear out any extra data captured. For on-the-fly processing multiple shorter DMA segments can be

programmed; at the interrupt restart DMA to point at the alternate segment to allow processing on the previous one. This technique is sometimes referred to as "ping-pong".

Please see the channel control register bit maps for more information.

Register Definitions - Base

STE3_BASE_PLL

[0x0000 Main Control Register Port read/write]

DATA BIT	DESCRIPTION
31-21	**spare**
20	bit 19 read-back of pll_dat register bit
19	pll_dat [write to PLL, read-back from PLL]
18	pll_s2
17	pll_sclk
16	pll_en
15-0	spare

FIGURE 7   PCI-NECL2-STE3 BASE CONTROL REGISTER

This is the base control register for STE3. The features common to all channels are controlled from this port. Unused bits are reserved for additional new features. Unused bits should be programmed '0' to allow for future commonality.

pll_en: When this bit is set to a one, the signals used to program and read the PLL are enabled.

pll_sclk/pll_dat : These signals are used to program the PLL over the I^{2}C serial interface. Sclk is always an output whereas Sdata is bi-directional. This register is where the Sdata output value is specified or read-back.

pll_s2: This is an additional control line to the PLL that can be used to select additional pre-programmed frequencies. Set to '0' for most applications.

The PLL is programmed with the output file generated by the Cypress programming tool. [CY3672 R3.01 Programming Kit or CyberClocks R3.20.00 Cypress may update the revision from time to time.] The .JED file is used by the Dynamic Driver to program the PLL. Programming the PLL is fairly involved and beyond the scope of this manual. For clients writing their own drivers it is suggested to get the Engineering Kit for this board including software, and to use the translation and programming files ported to your environment. This procedure will save you a lot of time. For those who want to do it themselves the Cypress PLL in use is the 22393. The output file from the Cypress tool can be passed directly to the Dynamic Driver [Linux or Windows] and used to program the PLL without user intervention. The reference frequency for the PLL is 50 MHz**.**

STE3_BASE_SWITCH

[$04 Switch and Design number port read only]

DATA BIT	DESCRIPTION
31-24	Design Revision
23-16	Design ID
15-8	PCI Core Revision
7-0	DIP switch

FIGURE 8   PCI-NECL2-STE3 ID AND SWITCH BIT MAP

The DIP Switch is labeled for bit number and '1' '0' in the silk screen. The DIP Switch can be read from this port and used to determine which PCI-NECL2-STE3 physical card matches each PCI address assigned in a system with multiple cards installed.   The DIPswitch can also be used for other purposes – software revision etc. The switch shown would read back 0x12.
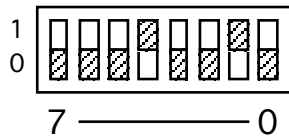
The Design ID and Revision are defined by a 16 bit field allowing for 256 designs and 256 revisions of each. The STE3 design is 0x01 the current revision is 0x04.

The PCI revision is updated in HW to match the design revision.   The board ID will be updated for major changes to allow drivers to differentiate between revisions and applications.

STE3_BASE_STATUS

[$08 Board level Status Port read only]

DATA BIT	DESCRIPTION
31-8	set to '0
7-2	set to '0', reserved for additional channels
1	Unmasked Ch1 Interrupt
0	Unmasked Ch0 Interrupt

FIGURE 9   PCI-NECL2-STE3 STATUS PORT BIT MAP

Channel Interrupt – The local interrupt status from the channel. Each channel can have different interrupt sources. DMA Write, DMA Read , IntForce or TX/RX request are typical sources. Polling can be accomplished using the channel status register and leaving the channel interrupt disabled.

STE3_BASE_GPIO_DIR

[$10 Board level GPIO Direction Register]

DATA BIT	DESCRIPTION
31-16	undefined
15-0	'1' = TX, '0' = RX 11-0 tied to GPIO

FIGURE 10   PCI-NECL2-STE3 GPIO DIRECTION BIT MAP

The GPIO port consists of 12 TTL programmable IO. The IO control are register based and independent. Each IO can be enabled for transmit, or disabled for receive modes. The IO can [as a bank] be tied to 3.3V or 5V. If set to 5V the IO will "float" at 5V unless the transmitter is enabled on the board or the "other end" is driving the cable. When the transmitter is enabled, the IO level will be near 3.3V or ground depending on the register IO bit definition. The 3.3V level comes from the 5V tolerant IO being driven with a 3.3V device. For seamless transitions the 3.3V reference is recommended [STE3 selection] for true 5V operation the 5V high can be obtained with the output disabled, and the 0V level reached with the output enabled and defined as a low.

STE3_BASE_GPIO_DATA

[$14 Board level GPIO Data Register]

DATA BIT	DESCRIPTION
31-16	undefined
15-0	GPIO Output Data [11-0]

FIGURE 11   PCI-NECL2-STE3 GPIO OUTPUT DATA BIT MAP

The GPIO port consists of 12 TTL programmable IO. The lower 12 bits correspond to the 12 port bits. When the transmitter is enabled with the corresponding Direction bit, the definition from this register will be driven onto the IO line.

Read-back provides the register contents – not necessarily the same as the IO definition as that can be affected by the external device.

STE3_BASE_GPIO_IO

[$18 Board level GPIO Input Register]

DATA BIT	DESCRIPTION
31-16	undefined
15-0	GPIO Input Data [11-0]

FIGURE 12   PCI-NECL2-STE3 GPIO INPUT DATA BIT MAP

This port is read-only and returns the state of the TTL IO. If an external device is not connected the value will match the Output Data register if the direction bits are set. If an external device is connected the IO will match the IO and may not match the internal register definition.

Register Definitions – Channel

The STE3 design has 1 active channel and 1 passive channel. The basic control signals are the same for the channel base, channel status, FIFO and DMA interfaces across multiple designs.

Notes:
The offsets shown are relative to the channel base address not the card base address.

STE_CHAN_CNTRL

[0x0] Channel Control Register (read/write)

Channel Control Register	
Data Bit	**Description**
31-19	spare
18	ByEnChRx
17	ByEnChTx
16	Init SDRAM
15-9	Spare
8	OutUrgent
7	InUrgent
6	Read DMA Interrupt Enable
5	Write DMA Interrupt Enable
4	Force Interrupt
3	Channel Interrupt Enable
2	Bypass
1	Receive FIFO Reset
0	Transmit FIFO Reset

FIGURE 13   PCI-NECL2-STE3 CHANNEL CONTROL REGISTER

FIFO Transmitter/Receiver Reset: When set to a one, the transmit and/or receive FIFO's will be reset. When these bits are zero, normal FIFO operation is enabled. In addition the Transmit and Receive State Machines are also reset.

Write/Read DMA Interrupt Enable: These two bits, when set to one, enable the interrupts for DMA writes and reads respectively.

Channel Interrupt Enable: When this bit is set to a one, all enabled interrupts (except the DMA interrupts) will be gated through to the PCI interface level of the design; when

this bit is a zero, the interrupts can be used for status without interrupting the host. The channel interrupt enable is for the channel level interrupt sources only.

Force Interrupt: When this bit is set to a one, a system interrupt will occur provided the Channel Interrupt enable is set. This is useful for interrupt testing.

InUrgent / OutUrgent when set causes the DMA request to have higher priority under certain circumstances. Basically when the TX FIFO is almost empty and InUrgent is set the TX DMA will have higher priority than it would otherwise get. Similarly if the RX FIFO is almost full and OutUrgent is set the read DMA will have higher priority. The purpose is to allow software some control over how DMA requests are processed and to allow for a higher rate channel to have a higher priority over other lower rate channels.

ByPass when set allows the FIFO to be used in a loop-back mode internal to the device. A separate state-machine is enabled when ByPass is set and the TX and RX are not enabled. The state-machine checks the TX and RX FIFO's and when not empty on the TX side and not Full on the RX side moves data between them. Writing to the TX FIFO allows reading back from the RX side. An example of this is included in the Driver reference software.

Init SDRAM when set causes the SDRAM controller to initialize the SDRAM. Initializing SDRAM is a specific sequence of steps the HW goes through to enable SDRAM operation. The bit is self clearing. Read-back can be used to determine if the initialization process has completed. The sequence is pretty rapid – the bit may be cleared by the time the SW returns to read it.

When Init SDRAM is set, the STE3_CHAN_RX_MEM_A register is used to set the SDRAM parameters after which the register is returned to normal use. This register must be initialized with the SDRAM parameters prior to the "Init" bit being set.

Once running the init bit will not have effect on the SDRAM unless the SDRAM controller is reset – returned to the idle state. The FIFO resets can be used for this purpose.

Normally a once per power up cycle operation.

ByEnChTx and ByEnChRx when set enable the bypass data movers to operate for the Tx or Rx channels respectively.

The main memory is organized with the SDRAM "in the middle" and FIFO's on either side for RX or TX operation. SW causes data to move to the SDRAM DMA TX FIFO, data is automatically moved from the SDRAM DMA TX FIFO to the SDRAM or to the SDRAM IO TX FIFO directly. When the Bypass bit is set, the data is moved directly from FIFO to FIFO avoiding the SDRAM. This will improve latency [slightly] but remove

the SDRAM memory from the path. The input and output FIFO's for the TX and RX direction are 512 x 32. Removing the SDRAM means a 1Kx32 FIFO between the PCI bus and the TX state machine or the RX state machine and the PCI bus.

The RX and TX state-machines have additional memory for the STE3 design.

When the bypass bits are disabled the data flows through the SDRAM allowing for a much larger storage space. The total memory is 8Mx32 and can be partitioned with SW. Please refer to the MEM registers for more information.

LM9_CHAN_STATUS
[0x4] Channel Status Read/Clear Latch Write Port

Channel Status Register

Data Bit	Description
31	Interrupt Status
30	LocalInt
29-24	Spare
23	BurstInIdle
22	BurstOutIdle
21	TxIdleState
20	RxIdleState
19	RxFifoOverFlowLat
18	TxUDoneLat
17	RXUDoneLat
16	TxCompletedLat
15	Read DMA Interrupt Occurred
14	Write DMA Interrupt Occurred
13	Read DMA Error Occurred
12	Write DMA Error Occurred
11	RxAFLvlIntLat
10	TxAELvlIntLat
9	RxAflInt
8	TxAmtInt
7	spare
6	Rx FIFO Full
5	Rx FIFO Almost Full
4	Rx FIFO Empty
3	Spare
2	Tx FIFO Full
1	Tx FIFO Almost Empty
0	Tx FIFO Empty

FIGURE 14   PCI-NECL2-STE3 CHANNEL STATUS PORT

STE3 FIFO: Two 512 x 32 FIFO's are used to create the internal DMA Tx and Rx memory. The status for the Tx FIFO and Rx FIFO refer to these FIFO's. The status is active high. 0x13 would correspond to empty Rx and empty Tx internal FIFO's.

Please note with the Rx side status; the status reflects the state of the FIFO and does not take the 4 deep pipeline into account. For example the FIFO may be empty and there may be valid data within the pipeline. The data count with the combined FIFO and pipeline value and can also be used for read size control. [see later in register

descriptions]

In addition there are two IO FIFO's to interface with the IO state-machines. The empty and full status for each FIFO is provided. When neither is set, there is data in the FIFO and room for more.

The Almost flags are set based on the entire memory including the SDRAM. The flags should be interpreted independently.

Rx FIFO Empty: When a one is read, the FIFO contains no data; when a zero is read, there is at least one data word in the FIFO.

Rx FIFO Almost Full: When a one is read, the number of data words in the data "FIFO" is greater than the value written to the corresponding RX_AFL_LVL register; when a zero is read, the FIFO level is less than that value. The comparison is done on the entire FIFO / SDRAM chain [SDRAM section only]

Rx FIFO Full: When a one is read, the receive data FIFO is full; when a zero is read, there is room for at least one more data-word in the FIFO.

Tx FIFO Empty: When a one is read, the FIFO contains no data; when a zero is read, there is at least one data word in the FIFO.

Tx FIFO Almost Empty: When a one is read, the number of data words in the data "FIFO" is less than or equal to the value written to the corresponding TX_AMT_LVL register; when a zero is read, the FIFO level is more than that value. The level compared is that of the SDRAM chain – DMA FIFO, SDRAM as defined by SW, TX IO FIFO.

Tx FIFO Full: When a one is read, the transmit data FIFO is full; when a zero is read, there is room for at least one more data-word in the FIFO.

RxFifoOverFlowLat: When a one is read, an error has been detected. This will occur if FIFO is full when the loader function tries to write to it. A zero indicates that no error has occurred. This bit is latched and can be cleared by writing back to the Status register with a one in the appropriate bit position.

Write/Read DMA Error Occurred: When a one is read, a write or read DMA error has been detected. This will occur if there is a target or master abort or if the direction bit in the next pointer of one of the chaining descriptors is incorrect. A zero indicates that no write or read DMA error has occurred. These bits are latched and can be cleared by writing back to the Status register with a one in the appropriate bit position.

Write/Read DMA Interrupt Occurred: When a one is read, a write/read DMA interrupt is latched. This indicates that the scatter-gather list for the current write or read DMA has completed, but the associated interrupt has yet to be processed. A zero indicates that no write or read DMA interrupt is pending.

Tx_IDLE is set when the state-machine is in the idle state. When lower clock rates are used it may take a while to clean-up and return to the idle state. If SW has cleared the start bit to terminate the data transfer; SW can use the IDLE bit to determine when the HW has completed its task and returned.

Rx_IDLE is set when the state-machine is in the idle state. When lower clock rates are used it may take a while to clean-up and return to the idle state. If SW has cleared the start bit to terminate the transfer; SW can use the IDLE bit to determine when the HW has completed its task and returned. RX IDLE is set to '0' this design – since SW controlled.

BO and BI Idle are Burst Out and Burst In IDLE state status for the Receive and Transmit DMA actions. The bits will be 1 when in the IDLE state and 0 when processing a DMA. A new DMA should not be launched until the State machine is back in the IDLE state. Please note that the direction implied in the name has to do with the DMA direction – Burst data into the card for Transmit and burst data out of the card for Receive.

Local Interrupt is the masked combined interrupt status for the channel not including DMA. The status is before the master interrupt enable for the channel.

Interrupt Status is the combined Local Interrupt with DMA and the master interrupt enable. If this bit is set this channel has a pending interrupt request.

RxAFLvlIntLat: When set the Rx Data FIFO has become almost Full based on the programmed count. The software can do a looped read or use DMA to retrieve the programmed count amount of data from the storage FIFO. The signal is latched and can be cleared via write back with this bit set. The signal can be used to generate an interrupt if desired.

TxAELvlIntLat: When set the Tx Data FIFO has become almost Empty based on the programmed count. The software can do a looped write or use DMA to load the programmed count amount of data to the storage FIFO. The signal is latched and can be cleared via write back with this bit set. The signal can be used to generate an interrupt if desired.

TxUDoneLat, RxUDoneLat should be treated as "don't care" for STE3. These bits apply to the retransmit modes designed into the SDRAM controller and not used for STE3.

<u>TxCompletedLat</u> When set '1' indicates the SM has completed a transfer – the FIFO has gone empty and the data transferred. This is a sticky bit cleared by writing back with the same bit set.   This bit is anded with the transmit interrupt enable to create an interrupt request if desired.

STE3_CHAN_WR_DMA_PNTR

[0x8] Write DMA Pointer (write only)

<table>
<tr><td colspan="2" align="center">BurstIn DMA Pointer Address Register</td></tr>
<tr><td>Data Bit</td><td>Description</td></tr>
<tr><td>31-2</td><td>First Chaining Descriptor Physical Address</td></tr>
<tr><td>1</td><td>direction [0]</td></tr>
<tr><td>0</td><td>end of chain</td></tr>
</table>

FIGURE 15   PCI-NECL2-STE3 WRITE DMA POINTER REGISTER

This write-only port is used to initiate a scatter-gather write [TX] DMA. When the address of the first chaining descriptor is written to this port, the DMA engine reads three successive long words beginning at that address. Essentially this data acts like a chaining descriptor value pointing to the next value in the chain.

The first is the address of the first memory block of the DMA buffer containing the data to read into the device, the second is the length in bytes of that block, and the third is the address of the next chaining descriptor in the list of buffer memory blocks. This process is continued until the end-of-chain bit in one of the next pointer values read indicates that it is the last chaining descriptor in the list.

All three values are on LW boundaries and are LW in size. Addresses for successive parameters are incremented. The addresses are physical addresses the HW will use on the PCI bus to access the Host memory for the next descriptor or to read the data to be transmitted.   In most OS you will need to convert from virtual to physical. The length parameter is a number of bytes, and must be on a LW divisible number of bytes.

Status for the DMA activity can be found in the channel control register and channel status register.

<u>Notes</u>:
1. Writing a zero to this port will abort a write DMA in progress.
2. End of chain should not be set for the address written to the DMA Pointer Address Register. End of chain should be set when the descriptor follows the last length parameter.

3. The Direction should be set to '0' for Burst In DMA in all chaining descriptor locations.

STE3_CHAN_TX_FIFO_COUNT
[0x8] TX FIFO data count (read only)

TX FIFO Data Count Port	
Data Bit	**Description**
31-0	TX Data Words Stored

FIGURE 16   PCI-NECL2-STE3 TX FIFO DATA COUNT PORT

This read-only register port reports the number of 32-bit data words in the Transmit FIFO. This design has a variable number of locations possible due to the programmable ranges for the TX and RX functions. 0 extended to a 32 bit value.

Maximum TX memory is 511[SDRAM DMA FIFO] + 511 [SDRAM IO FIFO] + 8M if all of the SDRAM is allocated to the TX process.

STE3_CHAN_RD_DMA_PNTR
[0xC] Read DMA Pointer (write only)

BurstIn DMA Pointer Address Register	
Data Bit	**Description**
31-2	First Chaining Descriptor Physical Address
1	direction [1]
0	end of chain

FIGURE 17   PCI-NECL2-STE3 READ DMA POINTER REGISTER

This write-only port is used to initiate a scatter-gather read [RX] DMA. When the address of the first chaining descriptor is written to this port, the DMA engine reads three successive long words beginning at that address. Essentially this data acts like a chaining descriptor value pointing to the next value in the chain.

The first is the address of the first memory block of the DMA buffer to write data from the device to, the second is the length in bytes of that block, and the third is the address of the next chaining descriptor in the list of buffer memory blocks. This process is continued until the end-of-chain bit in one of the next pointer values read indicates that it is the last chaining descriptor in the list.

All three values are on LW boundaries and are LW in size. Addresses for successive

parameters are incremented. The addresses are physical addresses the HW will use on the PCI bus to access the Host memory for the next descriptor or to read the data to be transmitted.   In most OS you will need to convert from virtual to physical. The length parameter is a number of bytes, and must be on a LW divisible number of bytes.

Status for the DMA activity can be found in the channel control register and channel status register.

Notes:
1. Writing a zero to this port will abort a write DMA in progress.
2. End of chain should not be set for the address written to the DMA Pointer Address Register. End of chain should be set when the descriptor follows the last length parameter.
3. The Direction should be set to '1' for Burst Out DMA in all chaining descriptor locations.

STE3_CHAN_RX_FIFO_COUNT

[0xC] RX FIFO data count (read only)

<div align="center">

RX FIFO Data Count Port

Data Bit	Description
31-0	RX Data Words Stored

</div>

FIGURE 18   PCI-NECL2-STE3 TX FIFO DATA COUNT PORT

This read-only register port reports the number of 32-bit data words in the Receive FIFO. This design has a variable number of locations possible due to the programmable ranges for the TX and RX functions. 0 extended to a 32 bit value.

Maximum RX memory is 511[SDRAM DMA FIFO] + 511 [SDRAM IO FIFO] + 8M + DMA Pipeline if all of the SDRAM is allocated to the RX process.

STE3_CHAN_FIFO

[0x10] Write TX/Read RX FIFO Port

RX and TX FIFO Port	
Data Bit	**Description**
31-0	FIFO data word

FIGURE 19   PCI-NECL2-STE3 RX/TX FIFO PORT

This port is used to make single-word accesses to and from the FIFO. Data read from this port will no longer be available for DMA transfers. Writing to the port loads the Tx FIFO, Reading unloads the Rx FIFO.

STE3_CHAN_TX_AMT_LVL

[0x14] Tx almost-empty level (read/write)

Tx Almost-Full Level Register	
Data Bit	**Description**
31-0	Tx FIFO Almost-Empty Level

FIGURE 20   PCI-NECL2-STE3 TX ALMOST EMPTY LEVEL REGISTER

This read/write port accesses the almost-empty level register. When the number of data words in the transmit data FIFO is less than than this value, the almost-empty status bit will be set. The register is R/W for 16 bits. The mask is valid for a size matching the depth of the FIFO. The count is based on the total memory allocated in the SDRAM memory chain.

STE3_CHAN_RX_AFL_LVL

[0x18] Rx almost-full (read/write)

Rx Almost-Full Level Register	
Data Bit	**Description**
31-0	Rx FIFO Almost-Full Level

FIGURE 21   PCI-NECL2-STE3 RX ALMOST FULL LEVEL REGISTER

This read/write port accesses the almost-full level register. When the number of data words in the receive data FIFO is equal or greater than this value, the almost-full status bit will be set. The register is R/W for 16 bits. The mask is valid for a size matching the depth of the FIFO. The count is based on the total memory allocated in the SDRAM memory chain. The level includes the pipeline for an additional 4 locations.

STE3_CHAN_TX
[0x1C] Channel Transmit Control Register (read/write)

Channel TX Control Register	
Data Bit	**Description**
31-10	Spare
9	EnTxUnLoad
8	EnTxLoad
7	spare
6	TxFifoAmtIntEnLvl
5	spare
4	spare
3	TxAEIntEn
2	TxIntEn
1	spare
0	TxEn

FIGURE 22   PCI-NECL2-STE3 CHANNEL TRANSMIT CONTROL REGISTER

Set spare bits to '0'.

TxEn when set causes the Transmit State Machine to begin operation. When the Transmitter state-machine has determined that Data is in the Transmit FIFO, and sufficient data is available transmission will begin. Clearing TxEn will return the State Machine to the idle state.

TxIntEn when set enables the transmitter to generate an interrupt when the transmitter has run dry and returned to the idle state. Please note the enable is not self clearing, adding data to the FIFO will retrigger the state machine unless software removes the enable.

TxAEIntEn when set enables the interrupt based on the TX FIFO Almost Empty flag. When the interrupt occurs a programmable amount of data can be stored into the FIFO

making for an efficient DMA or burst of writes to load the FIFO.

EnTxLoad and EnTxUnLoad when set enable the SDRAM state-machine to operate in the TX section of SDRAM. UnLoad is the function of taking data from the SDRAM and moving to the IO FIFO. Load is the function of taking data from the DMA FIFO and writing to the SDRAM. When using the retransmit modes in memory the unload function can be left disabled until the pattern is loaded and then enabled. For FIFO mode both enables can be set together.

STE3_CHAN_TX_READY_REG

[0x2C] Tx hold off (read/write)

<table>
<tr><td colspan="2" align="center">**Rx Almost-Full Level Register**</td></tr>
<tr><td>**Data Bit**</td><td>**Description**</td></tr>
<tr><td align="center">31-0</td><td align="center">Hold Off Byte Count</td></tr>
</table>

FIGURE 23   PCI-NECL2-STE3 TX HOLD OFF REGISTER

This read/write port is used to set a minimum threshold of data to be in the TX State-machine local FIFO before the State-Machine starts transmission. By programming this register to a non-zero amount the start and DMA start are decoupled somewhat.

Setting the level to 100 bytes, and the TX enabled before starting the DMA function will allow 100 bytes to accumulate in the IO local buffer before enable is set and data is transmitted. This gives the DMA function a head start and will allow the data transfer to be done in one contiguous transfer.

If having the enable set and disabled a few times during start-up does not affect your application the count can be set to 0. During DMA start-up the system may break the first segments into small transfers causing the FIFO to only load 1 LW [4 bytes] several times before the DMA really gets going. This is an OS dependent phenomenon.   The FIFO has 2044 as a maximum byte count. The odd count is due to using the block RAM for the memory and having unbalanced port sizes.

STE3_CHAN_RX
[0x34] Channel RX Control Register (read/write)

<table>
<tr><th colspan="2">Channel Control Register</th></tr>
<tr><th>Data Bit</th><th>Description</th></tr>
<tr><td>31-10</td><td>spare</td></tr>
<tr><td>9</td><td>EnRxUnLoad</td></tr>
<tr><td>8</td><td>EnRxLoad</td></tr>
<tr><td>7</td><td>spare</td></tr>
<tr><td>6</td><td>RxFifoAflIntEnLvl</td></tr>
<tr><td>5</td><td>spare</td></tr>
<tr><td>4</td><td>RxFifoOvFlIntEn</td></tr>
<tr><td>3</td><td>RxFifoAFIntEn</td></tr>
<tr><td>2</td><td>RxIntEn</td></tr>
<tr><td>1</td><td>spare</td></tr>
<tr><td>0</td><td>RxEn</td></tr>
</table>

FIGURE 24   PCI-NECL2-STE3 CHANNEL RX CONTROL REGISTER

RxEn when set causes the Rx State Machine to begin operation. The receiver uses the free running reference clock and looks for the enable signal to be in the active state when the falling edge of the clock is received. With each active clock [falling edge and enable set] data is captured [byte] and stored into a local FIFO with a x8 port on the IO side. The data is converted to 32 bit words. The data is stored LSB first MSB last. Data is continued to be captured until RxEn is disabled ['0'].

RxIntEn when set enables the completion of the Rx SDRAM loop to cause an interrupt. Unused in STE3. Set to '0'.

RxFifoAFIntEn when set enables the interrupt based on the Rx FIFO Almost Full flag. When the interrupt occurs a programmable amount of data can be read from the FIFO making for an efficient DMA read or burst of reads to unload the FIFO. Please see the programmable definition for the Almost Full Flag.

RxFifoOvFlIEn when set enables the interrupt based on the Rx FIFO OverFlow condition. When the State-machine writes to the FIFO the status is tested. If the FIFO is full when time to write the OverFlow status is set. If the interrupt is enabled the status is gated out to drive the interrupt request to the host.

The local IO FIFO data is transferred to the SDRAM RX IO FIFO automatically. For the full condition to happen the entire defined "FIFO" path would need to be filled.

EnRxLoad and EnRxUnLoad when set enable the SDRAM state-machine to operate in

the RX section of SDRAM. UnLoad is the function of taking data from the SDRAM and moving to the DMA FIFO. Load is the function of taking data from the IO FIFO and writing to the SDRAM. When using the retransmit modes in memory the unload function can be left disabled until the pattern is loaded and then enabled. For FIFO mode both enables can be set together.

While it is unlikely that the RX retransmit function would be used, it can be used as a self generating data stream for testing system operation.   Due to schedule requirements this function has not been tested for STE3.

Rx and Tx Memory Definitions. The SDRAM is a block of memory comprised of banks and pages. The hardware flattens the memory into one large page – that is the handling of the memory is hidden from the SW definitions. The SW can partition the memory into an RX and a TX segment or one large RX or TX segment or neither depending on the settings for the bypass, Load and UnLoad, plus memory address definitions.

The next set of registers is used to set the addresses to be used for the memory. The addresses are inclusive. For example setting Mem A = 0x00 will start the memory at 0x00. Setting MemD 0 0x10 will use address 0x10 as the last address in the segment.

For "normal" bidirectional operation the start, middle,and end of memory would be used to separate the Rx and Tx spaces.

Tx = 0x00 ⇔ 0x7FFFFF : MemA = 0x00, MemD = 0x7FFFFF
Rx = 0x800000 ⇔ 0xFFFFFF : MemA = 0x800000, MemD = 0xFFFFFF

Setting the control bits for FIFO operation will result in two large FIFO's with 1/2 the memory allocated to each. Asymmetrical memory allocation is fine too.

Each of the locations is 16 bits or 2 bytes => 16 Mbytes per FIFO in the example case.

STE3_CHAN_RX_MEM_A
[0x50] Rx Memory A offset definition (read/write)

Rx Memory Definition	
Data Bit	**Description**
31-0	RX Memory A

FIGURE 25   PCI-NECL2-STE3 RX MEMORY A REGISTER

The primary use of this port is to set the initial address of the RX segment of the SDRAM memory. The address is a word boundary and should be set to a LW boundary 0,2,4 etc.

For initialization the register should be set to x37 for loading to the SDRAM control register. Once initialization is complete the register should be set to the memory definition prior to enabling in the RX state-machine control register.

STE3_CHAN_TX_MEM_A
[0x54] Tx Memory A offset definition (read/write)

Tx Memory Definition	
Data Bit	**Description**
31-0	TX Memory A

FIGURE 26   PCI-NECL2-STE3 TX MEMORY A REGISTER

The primary use of this port is to set the initial address of the TX segment of the SDRAM memory. The address is a word boundary and should be set to a LW boundary 0,2,4 etc.

STE3_CHAN_SET_MEM
[0x58] Rx Memory A offset definition (read/write)

Memory Definition	
Data Bit	**Description**
31-0	Set Memory

FIGURE 27   PCI-NECL2-STE3 SET MEMORY REGISTER

This port is unused in STE3 and reserved for direct mapped memory applications.

STE3_CHAN_RX_MEM_B

[0x5C] Rx Memory B offset definition (read/write)

Rx Memory Definition	
Data Bit	**Description**
31-0	RX Memory B

FIGURE 28   PCI-NECL2-STE3 RX MEMORY B REGISTER

STE3_CHAN_TX_MEM_B

[0x60] Tx Memory B offset definition (read/write)

Tx Memory Definition	
Data Bit	**Description**
31-0	TX Memory B

FIGURE 29   PCI-NECL2-STE3 TX MEMORY B REGISTER

When operating with retransmitted data or single set data, the Memory B pointer is the start of the body of the data. For single non-looped or FIFO operations these registers are not used. When in looped mode, the memory pointers start at A, increase to B, increase to C and then loop back to B – N times – before incrementing to D.
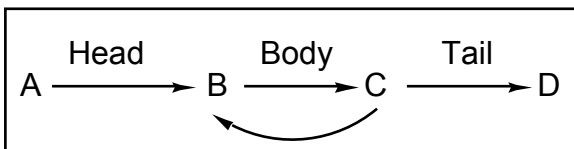
When operating in the looped mode it is important to enable the Load function without the UnLoad function to allow the memory to be set sequentially before being unloaded. The memory is assumed to be loaded before the unload function is enabled. Undefined memory could be moved to the Tx section if enabled before the memory is defined.

STE3_CHAN_RX_MEM_C

[0x64] Rx Memory C offset definition (read/write)

Rx Memory Definition	
Data Bit	**Description**
31-0	RX Memory C

FIGURE 30   PCI-NECL2-STE3 RX MEMORY C REGISTER

STE3_CHAN_TX_MEM_C

[0x68] Tx Memory C offset definition (read/write)

Tx Memory Definition	
Data Bit	**Description**
31-0	TX Memory C

FIGURE 31   PCI-NECL2-STE3 TX MEMORY C REGISTER

When operating with retransmitted data or single set data, the Memory C pointer is the end of the body of the data. For single non-looped or FIFO operations these registers are not used. When in looped mode, the memory pointers start at A, increase to B, increase to C and then loop back to B – N times – before incrementing to D.

The segment defined by C-B can be any length within the overall definition set by D-A.

The count is set by a separate register with an 8 bit field allowing up to 255 loops of the body per overall loop of the pattern.

The pattern can be sent once or loop back to A from D to repeat over and over.

Unused for STE3 operation.

STE3_CHAN_RX_MEM_D
[0x6C] Rx Memory D offset definition (read/write)

Rx Memory Definition	
Data Bit	**Description**
31-0	RX Memory D

FIGURE 32   PCI-NECL2-STE3 RX MEMORY D REGISTER

STE3_CHAN_TX_MEM_D
[0x70] Tx Memory D offset definition (read/write)

Tx Memory Definition	
Data Bit	**Description**
31-0	TX Memory D

FIGURE 33   PCI-NECL2-STE3 TX MEMORY D REGISTER

Register D is the end of the memory segment [Rx or Tx]. The address is included in the transmission. In FIFO mode the next address after D is A. In the retransmit modes the next address can be A if set to indefinite or the transmission can stop once D is reached.

It is important that the definitions for RX and TX Mem spaces are separate. The hardware does not check if the ranges overlap etc.   "Interesting" results can occur if the Rx and Tx spaces are overlapped.

Writing with TX and unloading with RX defined the same won't work in FIFO mode as the counters for the amount of available data are separate and writing to the TX side won't create available data for the RX side. On the other hand loading to TX and then unloading from Rx would [in theory] work with overlapped spaces. This mode has not been tested as it does not apply to STE3.

STE3_CHAN_RX_LOOP_CNT

[0x74] Rx Memory LoopCount definition (read/write)

Rx Memory Definition	
Data Bit	**Description**
7-0	RX Memory Loop Count

FIGURE 34   PCI-NECL2-STE3 RX LOOP COUNT REGISTER

STE3_CHAN_TX_LOOP_CNT

[0x78] Tx Memory LoopCount definition (read/write)

Tx Memory Definition	
Data Bit	**Description**
7-0	TX Memory Loop Count

FIGURE 35   PCI-NECL2-STE3 TX LOOP COUNT REGISTER

When operating in looped modes the Loop Count for the number of times the body is sent is defined in these registers. Normally used in the TX side only, but defined for both to allow for test cases and data generation. 1-255 are valid counts.

STE3_CHAN_RX_SDRAM_CMD

[0x7C] Rx Memory SDRAM Command definition (read/write)

Rx Memory Definition	
Data Bit	**Description**
7-4	Spare
3	Continuous
2	Ymode
1	Spare
0	Standard /Test mode

FIGURE 36   PCI-NECL2-STE3 RX SDRAM COMMAND REGISTER

STE3_CHAN_TX_SDRAM_CMD

[0x80] Tx Memory LoopCount definition (read/write)

Tx Memory Definition	
Data Bit	**Description**
7-4	Spare
3	Continuous
2	Ymode
1	Spare
0	Standard /Test mode

FIGURE 37   PCI-NECL2-STE3 TX SDRAM COMMAND REGISTER

Standard mode is used for FIFO and sequenced modes. Test is used for looping back data through the SDRAM and is for factory use only. Set to '0'.

Ymode when set causes the Loop Count to be applied and data to be read out with a header, body, and tail. Ymode when not set '0' selects FIFO mode. Select FIFO mode for STE3.

Continuous mode when set causes the memory to continuously loop in the Ymode. When I Ymode and not set the sequence is sent once. In FIFO mode the memory is always operating in continuous mode with data continually loaded and read out.

For STE3 set to 0x08.

Loop-Back

The Engineering kit uses the HDEterm100 with loop-back connections to provide a path to test the ECL I/O. The inputs are tied directly to the outputs. Please note: ECLIN/OUT 18 can be used to provide the external clock reference required for the transmitter.

FROM	TO		
OUT0P/OUT0M	IN0P/IN0M	24/74	1/51
OUT1P/OUT1M	IN1P/IN1M	25/75	2/52
OUT2P/OUT2M	IN2P/IN2M	26/76	3/53
OUT3P/OUT3M	IN3P/IN3M	27/77	4/54
OUT4P/OUT4M	IN4P/IN4M	28/78	5/55
OUT5P/OUT5M	IN5P/IN5M	29/79	6/56
OUT6P/OUT6M	IN6P/IN6M	30/80	7/57
OUT7P/OUT7M	IN7P/IN7M	31/81	8/58
OUT8P/OUT8M	IN8P/IN8M	32/82	9/59
OUT9P/OUT9M	IN9P/IN9M	33/83	10/60
OUT11P/OUT11M	IN11P/IN11M	35/85	12/62
OUT12P/OUT12M	IN12P/IN12M	36/86	13/63
OUT13P/OUT13M	IN13P/IN13M	37/87	14/64
OUT14P/OUT14M	IN14P/IN14M	38/88	15/65
OUT15P/OUT15M	IN15P/IN15M	39/89	16/66
OUT16P/OUT16M	IN16P/IN16M	40/90	17/67
OUT17P/OUT17M	IN17P/IN17M	41/91	18/68
OUT18P/OUT18M	IN18P/IN18M	42/92	19/69
OUT19P/OUT19M	IN19P/IN19M	43/93	20/70
TTL_0	TTL_1	45	95
TTL_2	TTL_3	46	96
TTL_4	TTL_5	47	97
TTL_6	TTL_7	48	98
TTL_8	TTL_9	49	99
TTL_10	TTL_11	50	100

D100 Standard Pin Assignment

The pin assignment for the PCI-ECL P1 connector.

IN0P	IN0M	1	51
IN1P	IN1M	2	52
IN2P	IN2M	3	53
IN3P	IN3M	4	54
IN4P	IN4M	5	55
IN5P	IN5M	6	56
IN6P	IN6M	7	57
IN7P	IN7M	8	58
IN8P	IN8M	9	59
IN9P	IN9M	10	60
IN10P	IN10M	11	61
IN11P	IN11M	12	62
IN12P	IN12M	13	63
IN13P	IN13M	14	64
IN14P	IN14M	15	65
IN15P	IN15M	16	66
IN16P	IN16M	17	67
IN17P	IN17M	18	68
IN18P	IN18M	19	69
IN19P	IN19M	20	70
GND	GND	21	71
GND	GND	22	72
GND	GND	23	73
OUT0P	OUT0M	24	74
OUT1P	OUT1M	25	75
OUT2P	OUT2M	26	76
OUT3P	OUT3M	27	77
OUT4P	OUT4M	28	78
OUT5P	OUT5M	29	79
OUT6P	OUT6M	30	80
OUT7P	OUT7M	31	81
OUT8P	OUT8M	32	82
OUT9P	OUT9M	33	83
OUT10P	OUT10M	34	84
OUT11P	OUT11M	35	85
OUT12P	OUT12M	36	86
OUT13P	OUT13M	37	87
OUT14P	OUT14M	38	88
OUT15P	OUT15M	39	89
OUT16P	OUT16M	40	90
OUT17P	OUT17M	41	91
OUT18P	OUT18M	42	92
OUT19P	OUT19M	43	93
GND	GND	44	94
TTL_0	TTL_1	45	95
TTL_2	TTL_3	46	96
TTL_4	TTL_5	47	97
TTL_6	TTL_7	48	98
TTL_8	TTL_9	49	99
TTL_10	TTL_11	50	100

FIGURE 38   PCI-NECL2-STE3 STANDARD D100 PINOUT

DYNAMIC ENGINEERING

Connector / Pin definition notes:

Note: IN0..19P/M and OUT0..19P/M refer to the ECL I/O.

IN7-0 are the data inputs.
IN8 is used for the Enable.   Active High
IN9 is used for the reference clock in. Falling edge stable data

OUT7-0 are the data outputs
OUT8 is used for the Enable. Active High
OUT9 is used for the TX clock reference. Data stable on the falling edge, switched on the rising edge

OUT18 is available as a looped-back clock. Used for ATP testing. Single board operation.

IN18 is the external Clock reference in. Used to generate the TX output clock [OUT9].

TTL IO are used for the GPIO port.

Undefined IO are tied to the FPGA and not utilized for STE3.

Applications Guide

Interfacing

Some general interfacing guidelines are presented below. Do not hesitate to contact the factory if you need more assistance.

ESD
Proper ESD handling procedures must be followed when handling the PCI-NECL2-STE3. The card is shipped in an anti-static, shielded bag. The card should remain in the bag until ready for use. When installing the card the installer must be properly grounded and the hardware should be on an anti-static work-station.

Start-up
Make sure that the "system" can see your hardware before trying to access it. Many BIOS will display the PCI devices found at boot up on a "splash screen" with the VendorID and CardId and an interrupt level. Look quickly! If the information is not available from the BI/OS, then a third party PCI device cataloging tool will be helpful; we use PCIView.

Watch the system grounds
All electrically connected equipment should have a fail-safe common ground that is large enough to handle all current loads without affecting noise immunity. Power supplies and power consuming loads should all have their own ground wires back to a common point.

Construction and Reliability

PCI Modules while commercial in nature can be conceived and engineered for rugged industrial environments. The PCI-NECL2-STE3 is constructed out of 0.062 inch thick high temperature ROHS compliant FR4 material.

The D100 connector has Phosphor Bronze pins with Nickel plating for durability and Gold plating on the contact area on both plugs and receptacles. The connectors are keyed and shrouded. The pins are rated at 1 Amp per pin, 500 insertion cycles minimum [at a rate of 800 per hour maximum]. These connectors make consistent, correct insertion easy and reliable.

Thermal Considerations

The PCI-NECL2-STE3 design consists of CMOS circuits for the memory and FPGA sections. The power dissipation due to internal circuitry is very low. The ECL buffers may require forced-air cooling. Heavy planes help to spread the power dissipated across the board for more effective cooling. In an enclosed space or when operating at elevated temperatures forced air is required. Thermocouples can be used to determine if forced air cooling is required for your application.

Warranty and Repair

Please refer to the warranty page on our website for the current warranty offered and options.

http://www.dyneng.com/warranty.html

Service Policy

Before returning a product for repair, verify as well as possible that the suspected unit is at fault. Then call the Customer Service Department for a RETURN MATERIAL AUTHORIZATI/ON (RMA) number. Carefully package the unit, in the original shipping carton if this is available, and ship prepaid and insured with the RMA number clearly written on the outside of the package. Include a return address and the telephone number of a technical contact. For out-of-warranty repairs, a purchase order for repair charges must accompany the return. Dynamic Engineering will not be responsible for damages due to improper packaging of returned items. For service on Dynamic Engineering Products not purchased directly from Dynamic Engineering contact your reseller. Products returned to Dynamic Engineering for repair by other than the original customer will be treated as out-of-warranty.

Out of Warranty Repairs

Out of warranty repairs will be billed on a material and labor basis. The current minimum repair charge is $150. Customer approval will be obtained before repairing any item if the repair charges will exceed one half of the quantity one list price for that unit. Return transportation and insurance will be billed as part of the repair and is in addition to the minimum charge.

For Service Contact:

Customer Service Department
Dynamic Engineering
150 DuBois St., Suite C
Santa Cruz, CA 95060
(831) 457-8891
FAX: (831) 457-4793
support@dyneng.com

Specifications

PCI Interfaces: PCI Interface 33 MHz. 32-bit

Access types: Configuration and Memory space utilized

CLK rates supported: 33 MHz. PCI, PLL with 50 MHz reference to provide programmable frequencies. PLL has 4 inputs to FPGA, 1 used for STE3.

Memory   Memory is provided 32Mbyte SDRAM per channel and user defined boundaries. FIFO and retransmit modes.

I/O   20 NECL Transmitters. 20 NECL receivers. 12 TTL with programmable direction. STE3 uses 10 ea. Tx and Rx plus clock input and reference clock output for loop-back

Interface: D100 connector. [AMP] 787082-9 is the board side part number

Software Interface: Control Registers within Xilinx.

Initialization: Programming procedure documented in this manual

Access Modes: Registers on long-word boundaries. Standard target access read and write to registers and memory. DMA access to memory.

Access Time: No wait states in DMA modes. 1-2 wait states in target access.

Interrupt: 1 interrupt to the PCI bus is supported with multiple sources. The interrupts are maskable and are supported with a status register.

Onboard Options: All Options are Software Programmable

Dimensions: Half-length PCI board.

Construction: High temp ROHS compliant FR4 Multi-Layer Printed Circuit, through Hole and Surface Mount Components.

Power: 5V, 3.3V from PCI bus. Local 3.3, 2.5, 1.2 and -5 created with on-board power supplies.

User   8 position software readable switch
                       Power LED's for each of the supplies plus PCI power

Order Information

Standard temperature range -40-70$^{\varnothing}$C

PCI-NECL2-STE3   http://www.dyneng.com/pci_ecl.html
half length PCI card with user re-configurable Xilinx, 40 ECL, 12 TTL I/O, 1 PLL

PCI-NECL2-STE3-ENG   Engineering Kit for the PCI-NECL2-STE3 Software, Schematic, Cable and HDEterm100. See webpage for more details and options including **software drivers**.

HDEterm100   http://www.dyneng.com/HDEterm100.html
100-pin connectors (2) matching the PCI-NECL2-STE3 D100 interconnected with 100 screw terminals. DIN rail mounting. Optional terminations and testpoints.

HDEcable100   http://www.dyneng.com/HDEcabl100.html
100 pin connector matching PCI-NECL2-STE3 and HDEterm100. Length options

All information provided is Copyright Dynamic Engineering