

**MASTER TRANSDUCER BUS  
NETWORK INTERFACE CONTROLLER  
(MTBNIC)  
USER'S GUIDE**

<b>AUTHOR:</b> GREG LENIHAN	<b>REVISION:</b> 1 ( <b>CONFIDENTIAL</b> )
--------------------------------	-----------------------------------------------

# TABLE OF CONTENTS

<u>DESCRIPTION</u>	<u>PAGE</u>
<b>1.0 OVERVIEW</b>	<b>5</b>
1.1 APPLICABLE DOCUMENTS	6
1.2 PURPOSE	6
1.3 MTBNIC INTRODUCTION LAYER	6
<b>2.0 NETWORK ARCHITECTURE</b>	<b>8</b>
2.1 DEVELOPMENT OVERVIEW	9
2.2 SYSTEM ARCHITECTURE	9
2.3 SUBNET	10
2.3.1 PHYSICAL LAYER	11
2.3.2 DATA LINK LAYER	12
<b>3.0 HOST BUS I/F</b>	<b>14</b>
3.1 TARGET IMPLEMENTATION	15
3.1.1 PCI Bus	15
3.1.1.1 PCI BUS TRANSACTION TYPES	16
3.1.1.2 CONFIGURATION REGISTERS DESCRIPTION	17
3.1.1.2.1 DEVICE INDEPENDENT REGION REGISTERS	17
3.1.1.2.1.1 BASE ADDRESS REGISTER 0	17
3.1.1.2.1.2 INTERRUPTS	18
3.1.1.2.1.2.1 INTERRUPT PIN REGISTER	19
3.1.1.2.1.2.2 INTERRUPT LINE REGISTER	19
3.1.1.2.2 DEVICE HEADER TYPE REGION REGISTERS	20
3.1.1.2.3 DEVICE COMMAND REGISTER	22
3.1.1.2.4 DEVICE STATUS REGISTER	22
<b>4.0 REGISTER DESCRIPTIONS</b>	<b>24</b>
4.1 MASTER SERIAL COMMUNICATION CONTROLLER	25
4.1.1 MDMSCC3208A DEVICE DESCRIPTION	25
4.1.1.1 INTRODUCTION	27
4.1.1.2 PAYLOAD DESCRIPTION	27
4.1.1.3 REGISTER DESCRIPTION	31
4.1.1.3.1 TRANSMITTER MODULE (TXMOD)	31
4.1.1.3.2 NODE CONFIGURATION (NODECG)	33
4.1.1.3.3 RECEIVER MODULE (RXMOD)	36
4.1.1.3.4 APPLICATION PROGRAMMING	37
4.2 MTBNIC REGISTER DESCRIPTION	38
<b>5.0 USER INTERCONNECT AND SWITCHES</b>	<b>40</b>
5.1 OVERVIEW	41
5.2 CONNECTOR DESCRIPTION	41
5.3 SWITCH DESCRIPTION	43

## LIST OF FIGURES

<u>DESCRIPTION</u>	<u>PAGE</u>
FIGURE 1.3.1: MTBNIC BLOCK DIAGRAM	
FIGURE 2.2: SYSTEM ARCHITECTURE FOR SENSOR PROCESSING SCHEME	
FIGURE 2.3: TBN SUBNET ILLUSTRATION	
FIGURE 2.3.1: TBN PHYSICAL LAYER TRANSCEIVER TOPOLOGY	
FIGURE 3.1.1: TARGET HEADER TYPE 0 CONFIGURATION REGISTER SPACE	
FIGURE 3.1.1.2.1: BASE ADDRESS REGISTER 0 (BAR0)	
FIGURE 3.1.1.2.1.2.1: INTERRUPT PIN REGISTER (IPR)	
FIGURE 3.1.1.2.1.2.2: INTERRUPT LINE REGISTER (ILR)	
FIGURE 3.1.1.2.2.1: PCI ID REGISTERS	
FIGURE 3.1.1.2.2.2: PCI CLASS REGISTERS	
FIGURE 3.1.1.2.2.3: PCI HEADER TYPE REGISTER	
FIGURE 3.1.1.2.3.1: PCI DEVICE COMMAND REGISTER DESCRIPTION	
FIGURE 3.1.1.2.4.1: PCI DEVICE STATUS REGISTER DESCRIPTION	
FIGURE 4.1.1.1: EFFICIENT NETWORK COMMAND SET	
FIGURE 4.1.1.2: BLOCK DIAGRAM	
FIGURE 4.1.1.3: COMPACT NETWORK PAYLOAD	
FIGURE 4.1.1.2.1: TYPICAL NETWORK SENSOR SYSTEM	
FIGURE 4.1.1.2.2: NETWORK COMMAND BYTE (CMD)	
FIGURE 4.1.1.2.3: NETWORK STATUS BYTE (STAT)	
FIGURE 4.1.1.2.4: ADDRESS BYTE (ADD)	
FIGURE 4.1.1.2.5: NUMBER OF BYTES FIELD (NBYT)	
FIGURE 4.1.1.2.6: NETWORK LAYER TRANSPORT BYTE (NWLTB)	
FIGURE 4.1.1.3.1.1: TX BUFFER QUE POINTER FLUSH REGISTER	
FIGURE 4.1.1.3.1.2: CONTROL BUFFER QUE WRITE REGISTER	
FIGURE 4.1.1.3.1.3: DATA BUFFER QUE WRITE REGISTER	
FIGURE 4.1.1.3.1.4: EXECUTE TRANSMIT WRITE BUFFER REGISTER	
FIGURE 4.1.1.3.2.1: NODE CONTROL REGISTER (NCR)	
FIGURE 4.1.1.3.2.2: START OF FRAME BYTE SYNC REGISTER (SOFR)	
FIGURE 4.1.1.3.2.3: END OF FRAME BYTE SYNC REGISTER (EOFR)	
FIGURE 4.1.1.3.2.4: NODE STATUS REGISTER (NSR)	
FIGURE 4.1.1.3.3.1: RX BUFFER QUE POINTER FLUSH REGISTER	
FIGURE 4.1.1.3.3.2: CONTROL BUFFER QUE READ REGISTER	
FIGURE 4.1.1.3.3.3: DATA BUFFER QUE READ REGISTER	
FIGURE 4.1.1.3.4.1: TRANSMIT BUFFER ROUTINE	
FIGURE 4.1.1.3.4.2: RECEIVER BUFFER ROUTINE	
FIGURE 4.2.1: MTBNIC PCI REGISTER SPACE DESCRIPTION	
FIGURE 5.2.1: PC104+ PCI CONNECTOR, J1, SIGNAL DESCRIPTION	
FIGURE 5.2.2: MEZZANINE CONNECTOR SIGNAL DEFINITION	
FIGURE 5.2.3: ZONAL CONNECTOR SIGNAL DEFINITION	
FIGURE 5.2.4: MISCELLANEOUS CONNECTOR SIGNAL DEFINITION	

## LIST OF TABLES

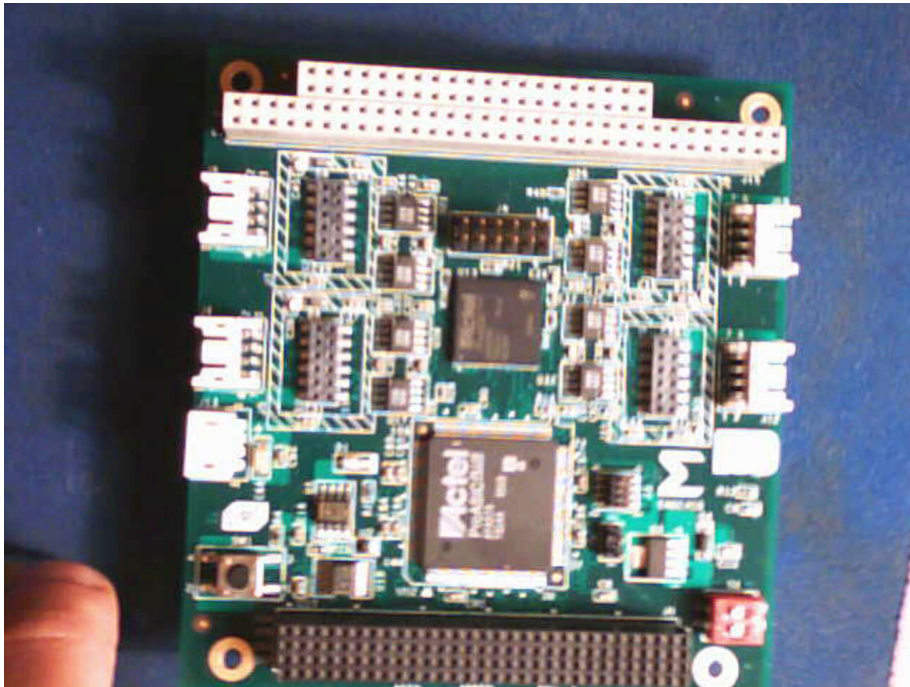
<u>DESCRIPTION</u>	<u>PAGE</u>
TABLE 3.1.1.1: PCI TRANSACTION TYPES	17
TABLE 4.1.1.3: MDMSCC3208A REGISTER SUMMARY	31

## ACRONYMS

ADD	Address Byte
BOP	Byte Oriented Protocol
CF	Catastrophic Failure
CHA	Channel
CMD	Command
CRC	Cyclic Redundancy Code
DB	Data Byte
EDAC	Error Detection and Correction
EOF	End of Frame
FE	Framing Error
FPGA	Field Programmable Gate Array
IP	Intellectual Property
ITT	Illegal Instruction Transaction
MBE	Multiple Bit Error
MIPS	Million of Instructions per Second
NBYT	Number of Bytes
NCAP	Network Capable Application Processor
NWLTB	Network Layer Transport Byte
OSI	Open System Interface
PAR	Parity
PCI	Personal Computer Interface
PID	Packet Identification Number
SAP	Service Application Peer
SBC	Single Board Computer
SBE	Single Bit Error
SCC	Serial Communication Controller
SNAP	Slave Node Acquisition Processor
SOF	Start of Frame
SRED	Slave Response Error Detection
TBI	Transducer Bus Interface
TBIM	Transducer Bus Interface Module
TBN	Transducer Bus Network
TBNIC	Transducer Bus Network Interface Controller
TEDS	Transducer Electronic Data Structure
VHDL	VHSYC High Level Description Language

# SECTION 1

## OVERVIEW



**ILLUSTRATION OF PC 104+ FORM-FACTOR BASED MTBNIC**

## **1.0 APPLICABLE DOCUMENTS**

- > Transducer Bus Network (TBN) Master Serial Communication Controller (MSCC) data sheet, P/N MDMSCC3208A.
- > Zonal Switch Card User's Guide, Endevco document number, TBD.
- > Slave Node Acquisition Processor (SNAP) User's Guide, Endevco document number, TBD.

## **1.1 PURPOSE**

The document provides critical user interface for programming application code over the Endevco Transducer Bus Network (TBN). This User's Guide is intended for programming the Master Transducer Bus Network Interface Controller (MTBNIC), which represents the master controller for the multi-drop TBN.

This document first provides an overview of the MTBNIC, including block diagrams, and then discusses Endevco, proprietary network, TBN. Next, a detailed description is provided for configuring, programming, and thus usage of MTBNIC in a sensor network application.

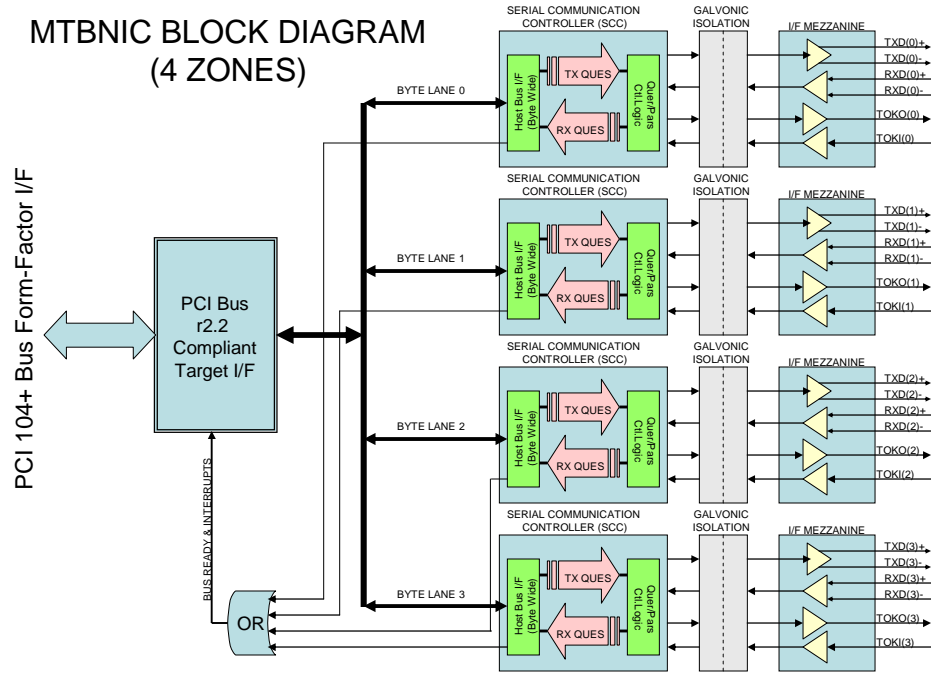
## **1.2 MTBNIC INTRODUCTION**

MTBNIC block diagram is illustrated in Figure 1.2.1.

The MTBNIC is on PC104+ form-factor and can be configure, via 2-position Dip Switch to occupy any of the available 4 PC104+, PCI slots. Four TBN zones are provided for the system programmer. Each zone physically maps to a PCI byte lane. All PCI bus transactions are I/O type. As discussed in Section 4.0, there are 11 32-bit dedicated registers for networking. As for any PCI target interface board, the MTBNIC possesses a configuration space that must be programmed in user bios.

In addition to providing 4 zones, the MTBNIC facilitates, Endevco transceiver mezzanine boards—one for each zone. This allows the system integrator to employ a unique PHY layer transceiver topology for each zone.

An external 3.3 VDC power supply, which is assumed to be isolated from PCI 3.3 VDC bus power, is provided for powering transceiver mezzanine interface boards as well as Galvonic digital isolators. It is assumed that the system integrator will be utilizing the Endevco Zonal Switch Card, which is also PC104+ form-factor based, and provides the isolated 3.3 VDC power, referred to as 3.3VZ.



**FIGURE 1.2.1: MTBNIC BLOCK DIAGRAM.**

## **SECTION 2**

# **NETWORK ARCHITECTURE**

## **2.1 DEVELOPMENT OVERVIEW**

The Transducer Bus Network (TBN) is a multi-drop, single Master, and multiple Slave interfaces, providing cogent communication for acquiring distributed sensor data.

The TBN comprises a variant of the IEEE 802.3 Subnet peer to peer, as well as application layer peer to peer transaction handling.

The TBN incorporates, Master and Slave, TB Network Interface Controller (TBNIC) devices, implemented in Field Programmable Gate Arrays (FPGAs), as well as physical layer interface transceivers, LVDS, RS485, etc., which enable transaction handling between a Host, MIPS based Single Board Computer (SBC), and embedded Slave Data Acquisition Controller (SDAC). The TBNIC provides the Service Application Peer (SAP), Network Layer, and Data Link Layer of the Subnet, while the Host SBC and SDAC perform the peer to peer Application Layer duties.

The TBN logic development takes full advantage of portable IEEE 1076 compliant VHDL IP cores. These cores have been developed by Endevco Engineering staff, and are intended for re-use in a variety of customer driven TBN topologies.

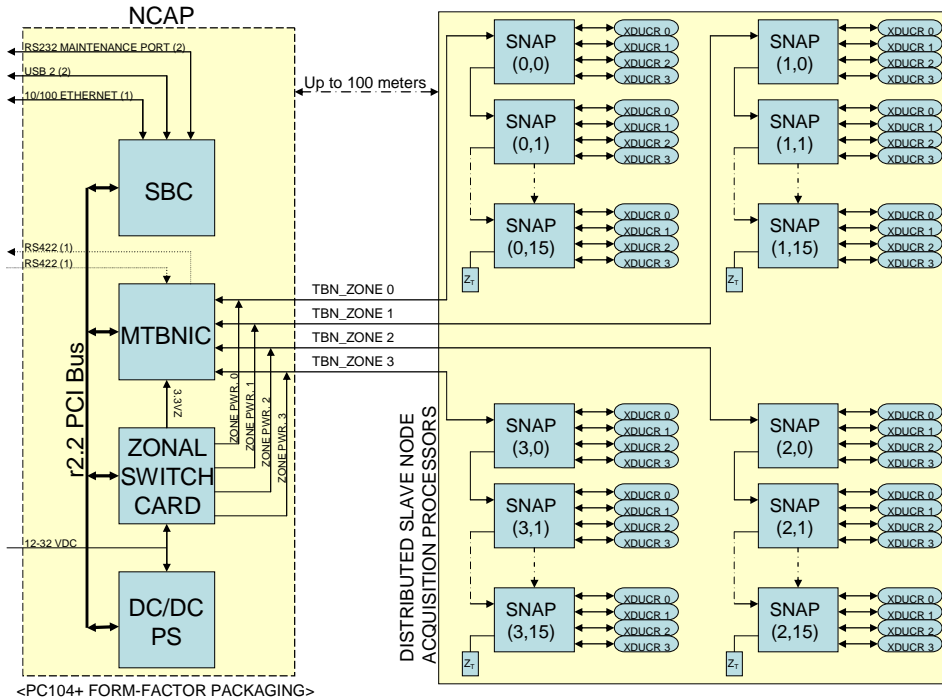
## **2.2 SYSTEM ARCHITECTURE**

Figure 2.2 illustrates Endevco system architecture for implementing acquisition and processing of 256 transducers, which could be sensors and, or actuators, using NCAP, TBN, and a plethora of distributed Slave Node Acquisition Processors (SNAPs)—sometimes referred to as Transducer Bus Interface Modules (TBIMs). An NCAP comprises a 400 MIPS based SBC, MTBNIC, Zonal Switch Card (SWC) for monitoring and controlling TBN zone power, facilitating, via a TBN, acquisition of 16 SNAPs, which process 4 transducer channels. Hence, a single NCAP is capable of acquiring and processing 4 zones X 16 SNAPs/zone X 4 Transducers/SNAP, which yields 256 Transducers.

The PCI Bus allows for 4 slots, thus facilitating one additional MTBNIC and accompanying SWC. This means that a total of 512 Transducers may be acquired and processed per NCAP. Further, multiple NCAPs can be employed via the USB port connection, or the 10/100 Ethernet port connection. The Ethernet connection is also provided for code development, i.e., upload and download.

The architecture is truly “Plug n Play” (PnP) at the Host PCI Bus level, as well as at the SNAP/ Transducer Channel level. The SNAP provides a more detailed, or descriptive 32 byte Transducer Electronic Data Structure (TEDS), which allows a SNAP to acquire and process a plethora of sensor types, ranging from basic pressure transducers to accelerometers. Further, software and digital hardware architecture permits each channel to be configured for personal dynamic range, as well as data format: i.e., 8-bit signed, unsigned, 12-bit signed, unsigned, or 16-bit signed, unsigned.

As will be discussed in sections to follow, the TBN processing architecture provides health and status semaphores, which will maintain sampling synchronicity among all sensor channels, as well as preventing lost packets. As will also be seen, network payload size is variable, thus allowing the system programmer flexibility and bandwidth efficiency.



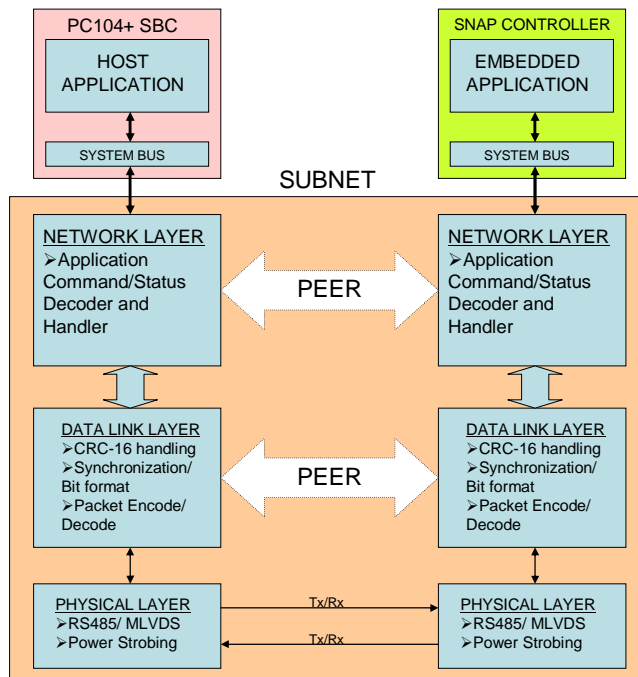
**FIGURE 2.2: SYSTEM ARCHITECTURE FOR SENSOR PROCESSING SCHEME.**

The SNAPs, are true distributed processors, and as such perform acquisition, linearization, digital filtering, and queing of all sampled sensor data.

The sections to follow will discuss each layer of the Subnet, including detailed payload description. Following the Subnet, the Application layer, with register descriptions, memory map, including PCI Bus interface, will be discussed.

### **2.3 SUBNET**

Figure 2.3 illustrates the TBN Subnet, which looks like any Subnet from the Open System Interface (OSI) model. The Subnet is truly “peer to peer.” The only processing aspect which differentiates a MTBNIC from a SNAP is the “Master/ Slave” concept: i.e., Masters are exclusively granted command initiation authority, while a Slave, when addressed by the Master, is granted response authority. This transaction concept is derived from the fact that the current TBN architecture facilitates multi-drop communication only. There can only be one Master, and thus, there is not an arbiter or “token” handler for multi-master access.



**FIGURE 2.3: TBN SUBNET ILLUSTRATION.**

### **2.3.1 PHYSICAL LAYER**

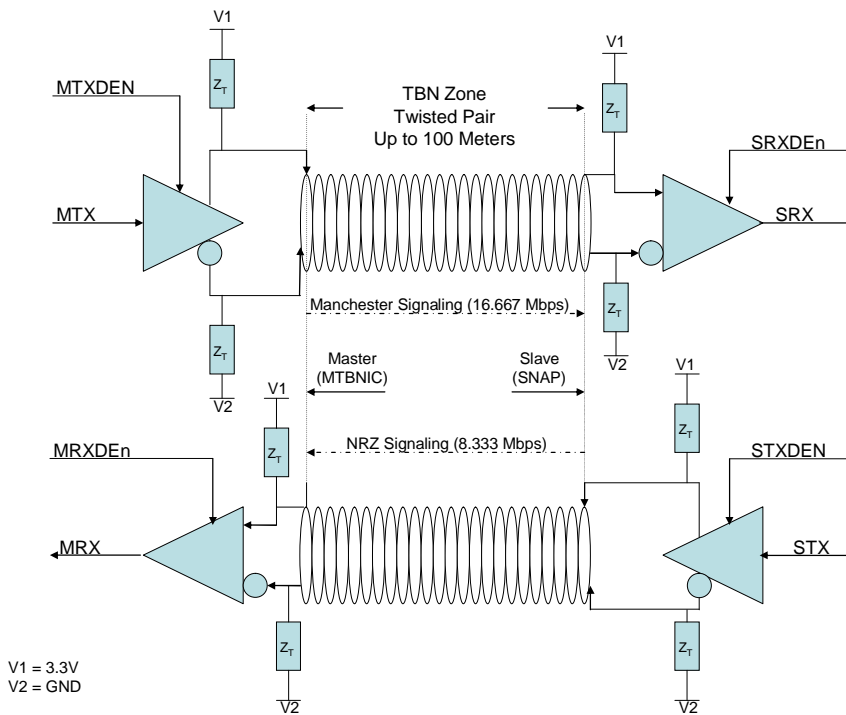
The TBN physical layer comprises a differential pair driver and receiver for transmission and reception. The protocol only facilitates a single signal for transmit and receive, thus there is no clock and request to send signal. This scheme naturally requires a Byte Oriented Protocol (BOP) synchronization scheme. The motivation for employing such a scheme is wire reduction. However, the bit synchronization scheme becomes more complicated. Further complicating bit synchronization is the fact that the Slave transmitter can never be enabled unless addressed, thus prohibiting the Master from maintaining a synchronous clock recovery, such as Manchester.

For TBN transactions, the Master always emits Manchester signaling at 16.667 Mbps, which translates to 8.333 Mbps, baseband, for clock and data recovery at the Slave end. The MTBNIC (Master) transmitter is synchronized to PCI Bus Clock (33.33 Mhz). All SNAPs (Slaves) utilize Manchester Master, signaling data, for clock and data recovery, thus maintaining synchronization with NCAP PCI Bus clock. Slaves respond with NRZ data on their respective transmit line, which is synchronized to Master's 33.33 Mhz clock.

Figure 2.3.1 illustrates TBN transceiver topology. The termination,  $Z_T$ , is threefold: First, it is balanced differential, which facilitates common mode rejection, as well as impedance matching source impedance of the driver with the characteristic impedance of the twisted pair. Finally, the termination is biased about  $V_1$  and  $V_2$ , whereby enough potential

difference is provided to “bias” up to a logic level high. This last aspect is critical for bit synchronization. Being biased up, means the state of the Slave transmitters will be a logic high or “idle arm” state, such that a bit scheme, such as Non-Return to Zero Low (NRZ-L) can be employed, since clock recovery becomes difficult when an emitted preamble cannot be employed.

As previously alluded, depending upon the application, a variety of transceivers can be employed, including Multiple Low Voltage Differential Signaling (MLVDS). Though power consuming, RS485 transceiver implementation is popular for its transmission distance versus bit rate.



**FIGURE 2.3.1: TBN PHYSICAL LAYER TRANSCEIVER TOPOLOGY.**

### **2.3.2 DATA LINK LAYER**

The data link layer (DLL) performs packet handling, including error detection, via CCITT compliant CRC-16 polynomial, as well as bit and BOP synchronization.

Since the TBN data link layer is performing BOP oriented synchronization, it is employing a variant of IEEE 802.3 and Synchronous Data Link Control (SDLC). Similar to IEEE 802.3 and numbered information frames of SDLC, the DLL utilizes a compact sequencing scheme, when transacting packets between Master and Slave, which is based on a Packet Identifier (PID) field to be discussed later.

Both Master and Slave TBN utilize Serial Communication Controller (SCC) VHDL IP, ported to FPGAs. The MTBNIC employs a Quad Master SCC (MSCC), while the SNAP employs a Slave SCC (SSCC). The fundamental difference between the two cores is programming authority, as well as synchronization mechanism. The SSCC must utilize recovered clock and employs simple housekeeping.

This document only discusses MSCC in detail. The SSCC is discussed in detail in the SNAP development document.

## **SECTION 3**

### **HOST BUS I/F**

### **3.0 Host Bus I/F**

The MTBNIC employs target, r2.2 compliant, PCI Bus, interface.

### **3.1 TARGET IMPLEMENTATION**

The target bus interface is implemented with Intellectual Property (IP) core, which, presently, only implements target interface, with future provisions for bus master handling, such that Direct Memory Access (DMA) transactions may be transacted.

The IP is implemented in an Actel, APA075-TQ144I, flash-based, FPGA.

#### **3.1.1 PCI Bus**

PCI requires a minimum configuration space be utilized and processed. The Host Bridge utilizes a North Bridge, identified as Bridge 0, Bus 0, and facilitates for Type 0 transactions. The target TBI boards are identified, during PCI configuration, via signal bits, IDSEL[3:0]. Each target board is connected to a specific IDSEL signal. During a configuration access, the Host Bridge asserts IDSEL(0), IDSEL(1), IDSEL(2), or IDSEL(3). PCI configuration accesses are always begun with configuration read cycles of the selected device—always beginning with Device 1 (IDSEL[3:0]=0001).

#### Header Type 0 PCI Configuration Register Space

D31	D23	D15	D7	D0	LW Number
Device ID		Vendor ID			\$0h
Status Register		Command Register			\$1h
Class Code	Sub-Class Code	Prog. I/F	Revision ID		\$2h
BIST	Header Type	Latency Timer	Cache Line Size		\$3h
Base Address Register 0					\$4h
Base Address Register 1					\$5h
Base Address Register 2					\$6h
Base Address Register 3					\$7h
Base Address Register 4					\$8h
Base Address Register 5					\$9h
Card Bus CIS Pointer					\$Ah
Subsystem ID		Subsystem Vendor ID			\$Bh
Expansion ROM Base Address Register					\$Ch
Reserved					\$Dh
Reserved					\$Eh
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line		\$Fh

**FIGURE 3.1.1: TARGET HEADER TYPE 0 CONFIGURATION REGISTER SPACE.**

Figure 3.1.1 illustrates the minimum implementation of the target configuration space. Actually, not all of the registers need to be configured by the target and host. The following registers are required:

- Device ID
- Vendor ID
- Status Register
- Command Register
- Class Code
- Sub-Class Code
- Prog I/F
- Revision ID
- Header Type
- Base Address Register 0
- Interrupt Pin
- Interrupt Line

PCI only permits a single device ID, with one configuration, to utilize one bus interrupt. Because of this burden, the TBI target boards shall employ the multiple ID option, which permits each zone of each TBI to have ownership of one unique PCI bus interrupt, INTAn, INTBn, INTCn, or INTDn.

Before discussing configuration of the target configuration space, allowable PCI Master/Target interface transaction types must be described.

#### **3.1.1.1 PCI BUS TRANSACTION TYPES**

The targets facilitate the following PCI bus transactions, as signaled by the CBEn[3:0] signal bus:

**TABLE 3.1.1.1: PCI TRANSACTION TYPES**

PCI Bus Transaction Types: Output during Address Phase		
CBEn[3:0]	Command Type	NCAP Usage
0000b	Interrupt Acknowledge	Implemented
0001b	Special Command	Implemented
0010b	I/O Read	Implemented
0011b	I/O Write	Implemented
0100b	Reserved	
0101b	Reserved	
0110b	Memory Read	
0111b	Memory Write	
1000b	Reserved	
1001b	Reserved	
1010b	Configuration Read	Implemented
1011b	Configuration Write	Implemented
1100b	Memory Read Multiple	
1101b	Dual Address Cycle	
1110b	Memory Read Line	
1111b	Memory Write and Invalidate	

### **3.1.1.2 CONFIGURATION REGISTERS DESCRIPTION**

At the onset, there are two types of configuration register spaces in Figure 3.1.1: PCI Device Independent Region, Offset address, \$00h-\$0Fh, and PCI Device Header Type Region, Offset address, \$10h-\$3Fh.

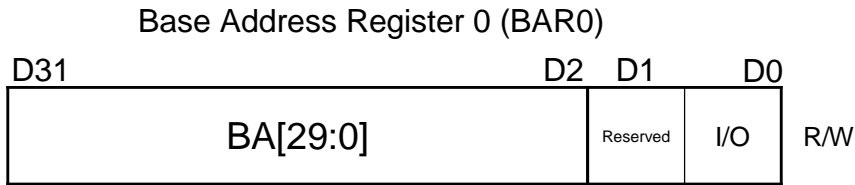
#### **3.1.1.2.1 DEVICE INDEPENDENT REGION REGISTERS**

The following registers must be configured for the sake of the application:

- Base Address Register 0
- Interrupt Pin Register <Read Only>
- Interrupt Line Register

##### **3.1.1.2.1.1 BASE ADDRESS REGISTER 0**

Configuration of Base Address Register 0 is required, so I/O accesses may be transacted. The Target provides power up value data for the Host/ Bridge during configuration access, which informs the Host/ Bridge how much I/O space the Target requires. The Host/ Bridge can then map the Target into its I/O address space by writing a Base Address to the Target's Base Address Register 0.



Bit Definition:

I/O – I/O Space Indicator

Logic “0”: Space is memory access

Logic “1”: Space is I/O access

BA[29:0] – Base Address Space required by Target

Power Up Value = \$FFFFFF01

### **FIGURE 3.1.1.2.1: BASE ADDRESS REGISTER 0 (BAR0).**

As part of the housekeeping PnP algorithm, the software will write \$FFFFFFFFh to the BAR0, and will read back \$FFFFFF01. This indicates I/O space for 256 Bytes is required by the Target. The PnP algorithm will then assign and write a base address to the BAR0.

### **3.1.1.2.1.2 INTERRUPTS**

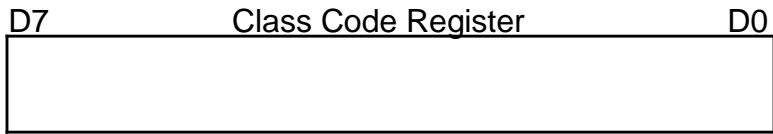
As part of the configuration access, the Interrupt Pin Register (IPR) must be read, and the Interrupt Line Register (ILR) must be programmed. The IPR indicates which PCI Interrupt level the Target will use: INTAn, INTBn, INTCn, or INTDn. The Target can *only* drive *one*, not all 4, 3, or 2. The ILR is programmed into the Target, so the Host processor knows which x86 interrupt vector to service, during its interrupt service routine (ISR) handling. The ILR is accessed in byte lane 0, during a PCI interrupt acknowledge bus cycle.

### **3.1.1.2.1.2.1 INTERRUPT PIN REGISTER (IPR)**

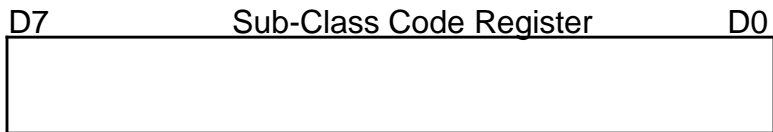




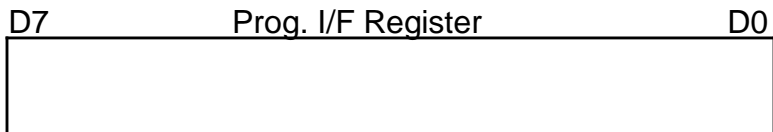
**FIGURE 3.1.1.2.2.1: PCI ID REGISTERS.**



Mandatory register configuration, \$FFh means device does not fit any defined classes. The PCI Target powers up with Class Code = \$02h (Network Controller).

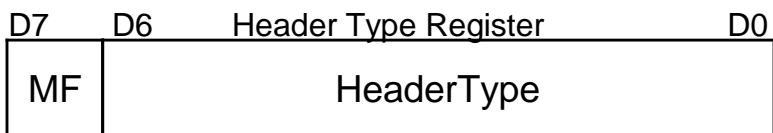


Mandatory register configuration. The PCI Target powers up with Sub-Class Code = \$80h (Other Network Controller).



Mandatory register configuration. The PCI Target powers up with Prog. I/F Register = \$00h

**FIGURE 3.1.1.2.2.2: PCI CLASS REGISTERS.**



Mandatory register configuration.

Bit Definition:

MF – Multi-function select  
 Logic “0”: Single Function  
 Logic “1”: Multi-function

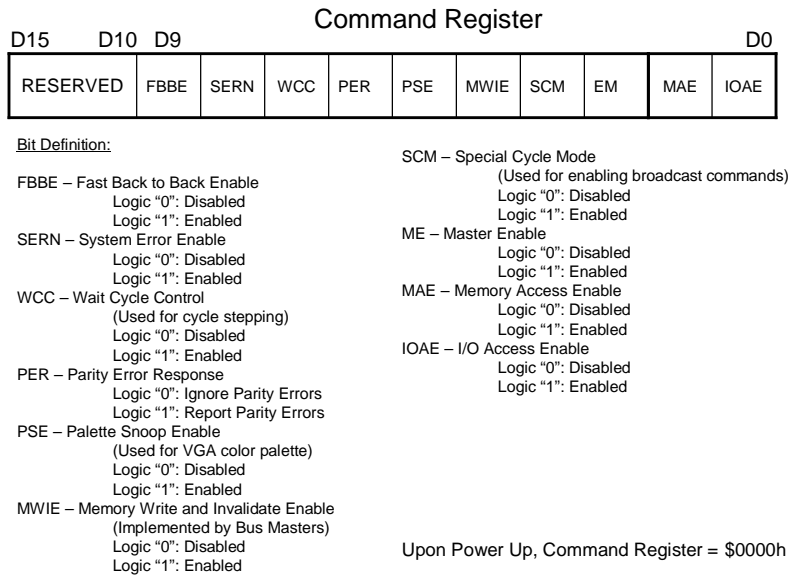
HeaderType[6:0]  
 \$00h <Header Type 0>  
 Bridge to Bus transaction handling  
 \$01h <Header Type 1>  
 Bridge to Bridge transaction handling

The Header Type Register powers up with \$00h. It is recommended the Host not write this register.

**FIGURE 3.1.1.2.2.3: PCI HEADER TYPE REGISTER.**

**3.1.1.2.3 DEVICE COMMAND REGISTER**

The system programmer must configure the Command Register during configuration access. This is the *last* configuration operation performed. Once all other configuration registers have been processed, the Command Register should be written with \$0149h.



**FIGURE 3.1.1.2.3.1: PCI DEVICE COMMAND REGISTER DESCRIPTION.**

**3.1.1.2.4 DEVICE STATUS REGISTER**

The Status Register must be read before the Command Register is configured during configuration access. The pertinent Base Address Registers must also be read before writing the Command Register. Although some bits may be written in the Status Register; they may only be cleared.

### Status Register

D15	D14	D13	D12	D11	D[10:9]	D8	D7	D6	D5	D4	D0
DPE	SSE	RMA	RTA	STA	DEVT	DPR	FBBC	UDF	66MC	RESERVED	

**Bit Definition:**

<p><b>DBE</b> – Detected Parity Error          Logic "0": False          Logic "1": True</p> <p><b>SSE</b> – Signaled System Error          (Set whenever SERRn is asserted)          Logic "0": False          Logic "1": True</p> <p><b>RMA</b> – Received Master Abort          (Set by Master whenever its transaction is terminated by another Master Abort)          Logic "0": False          Logic "1": True</p> <p><b>RTA</b> – Received Target Abort          (Set by Master whenever Target issues Abort)          Logic "0": False          Logic "1": True</p> <p><b>STA</b> – Signaled Target Abort          (Set by Target whenever it issues an abort—if enabled)          Logic "0": False          Logic "1": True</p>	<p><b>DEVT[1:0]</b> – Device Select Timing          (Read only bits configured by Target, and define slowest access, except configuration access)          00b = fast          01b = medium          10b = slow          11b = reserved</p> <p><b>DPR</b> – Data Parity Reported          (Used by Master only)          Logic "0": False          Logic "1": True</p> <p><b>FBBC</b> – Fast Back to Back Capable          (Read only bit indicating whether Target supports fast back-to-back transactions with different Targets)          Logic "0": Disabled          Logic "1": Enabled</p> <p><b>UDF</b> – User Defined Feature          (Read only)          Logic "0": Not supported          Logic "1": Supported</p> <p><b>66MC</b> – 66 Mhz Capable          Logic "0": Not capable          Logic "1": Capable</p> <p style="text-align: center;">Upon Power Up, Command Register = \$0020h</p>
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**FIGURE 3.1.1.2.4.1: PCI DEVICE STATUS REGISTER DESCRIPTION.**

## **SECTION 4**

# **REGISTER DESCRIPTIONS**

## **4.1 MASTER SERIAL COMMUNICATION CONTROLLER (MSCC)**

The MTBNIC utilizes a quad MSCC IP, referred to by P/N MDMSCC3208A, ported to Actel, APA300-FB144I device. Each MSCC device facilitates byte communication, whereby each byte lane is dedicated to a zone. This enables simultaneous or synchronous communication across all 4 zones, since all 4 MSCCs can be transacted, simultaneously, via 32-bit PCI bus cycles.

Before introducing MTBNIC register space definition, understanding of the MDMSCC3208A is paramount.

### **4.1.1 MDMSCC3208A DEVICE DESCRIPTION**

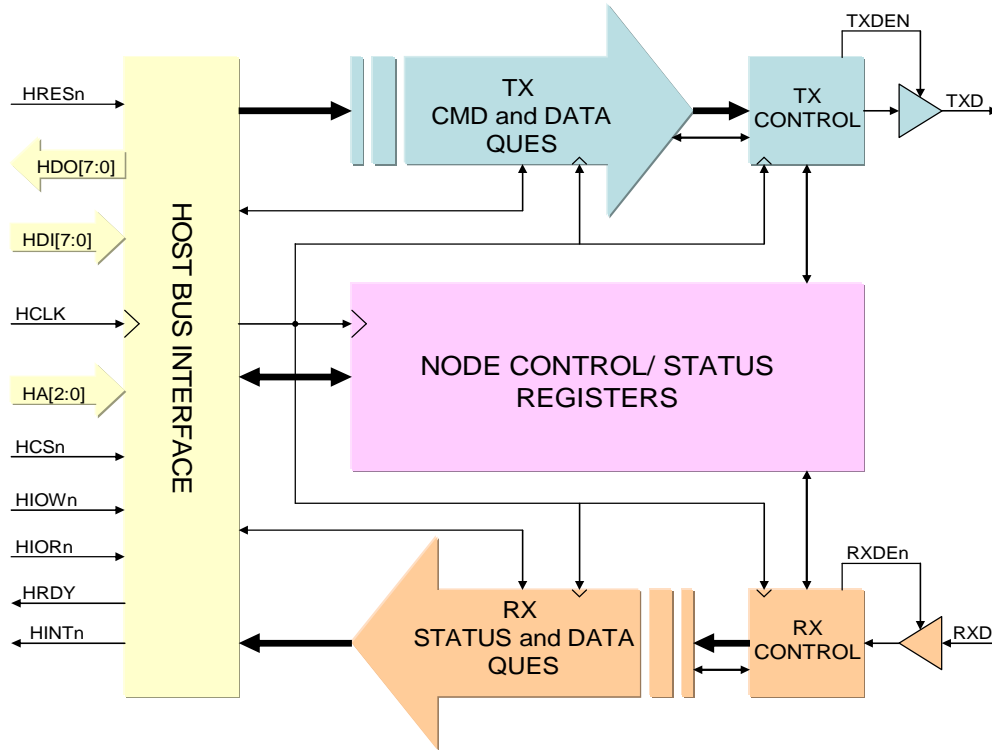
#### **MDMSCC3208A (Multi-drop Master Serial Communications Controller)**

- Master/ Slave operation over TIA-899 compliant multi-drop network.
- Conventional 802 Byte Oriented Protocol (BOP) with CCITT CRC-16.
- Manchester and PLL based clock recovery at both ends.
- Supports Supervisory (Node) and Information (Application) Broadcast and Multicast commands, as well as, Node and Application Write Acknowledge sequences.
- 1 to 256 Byte transfer capability with 98% maximum efficiency.
- Byte wide synchronous host bus interface for direct interface to PCI bus, allowing for 4 synchronized multi-drop networks.
- 8.3325 Mbps baseband (16.665 Mbps, signaling) achievable network bandwidth, employing Manchester, synchronized to r2.2, 33.33 Mhz, PCI Bus.

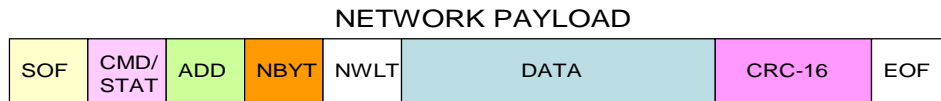
**CFCTN[1:0] - Command Function Mode Bits**

M(1)	M(0)	MODE DESCRIPTION
0	0	NODE BROADCAST FRAME (ND_BC)
0	1	NODE WRITE W/O ACK FRAME (ND_NAK)
1	0	NODE WRITE W/ ACK FRAME (ND_WAK)
1	1	NODE READ FRAME (ND_RD)

**FIGURE 4.1.1.1: EFFICIENT NETWORK COMMAND SET.**



**FIGURE 4.1.1.2: BLOCK DIAGRAM.**



**Field Description:**

- SOF – Start of Frame <1 Byte>
- CMD/ STAT – Command/ Status <1 Byte>
- ADD – Address <1 Byte>
- NBYT – Number of Bytes <1 Byte>
- NWLT – Network Layer Transport Byte <1 Byte>
- DATA – Data Payload <1 to 256 Bytes>
- CRC-16 – Cyclic Redundancy Code 16 <2 Bytes>
- EOF – End of Frame <1 Byte>

**FIGURE 4.1.1.3: COMPACT NETWORK PAYLOAD.**

#### **4.1.1.1 INTRODUCTION**

The MDSCC3208A provides cogent Master/ Slave multi-drop communication, employing a conventional 802 Byte Oriented Protocol (BOP). It is well suited for networked sensor communication, since it minimizes cabling, as well as facilitates transmission distances—when utilizing TIA-899 type transceivers—over 500 feet.

Figure 4.1.1.2.1 illustrates a typical network sensor system, employing a central processor, with 4 multi-drop networks, or zones. Each zone can facilitate up to 32 Slave Node Acquisition Processors (SNAPs). The MDSCC3208A allows for two types of transmission topologies: (1) Two-way simultaneous full-duplex, using Master node driven Manchester and Slave node NRZ data; and, (2) Two-way simplex, or half-duplex, whereby the Master node emits the NRZ clock, and comprises a bidirectional data transceiver.

#### **4.1.1.2 PAYLOAD DESCRIPTION**

The multi-drop network facilitates variable size packet transfers, as well as error handling. Four control bytes are provided for cogent data handling. The user can transmit 1 to 256 data bytes. Frames are demarcated by one, Start of Frame (SOF) sync, and one, End of Frame (EOF) sync. As discussed in Section 4.1.1.3, the frame syncs are programmable.

The Address byte (ADD) allows for accessing up to 256 bytes of Node space. For the NBYT field, the user enters \$00h to represent 1 byte and \$FFh to represent 256 bytes.

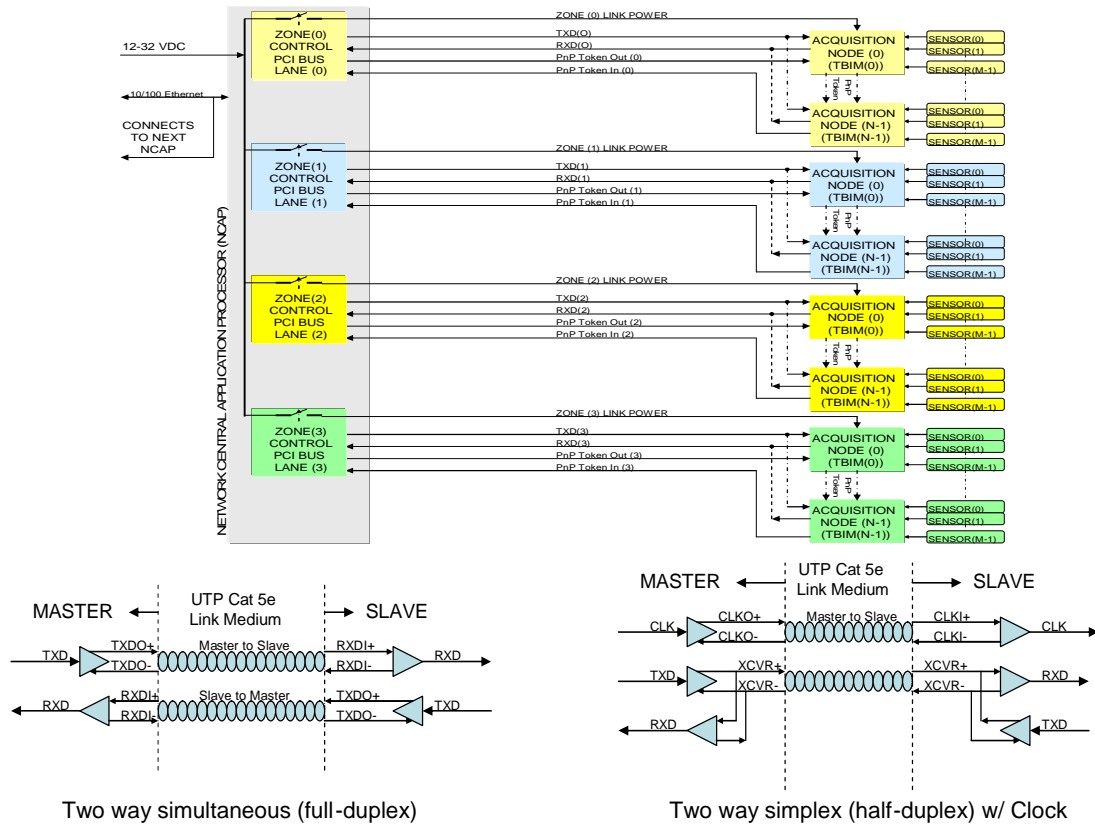
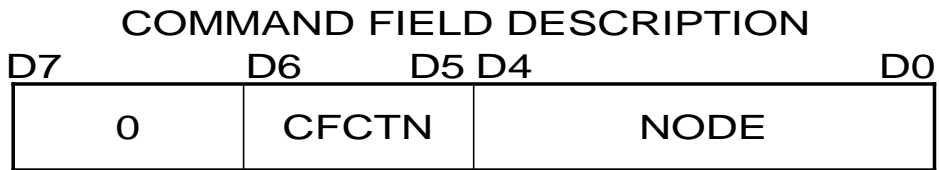


FIGURE 4.1.1.2.1: TYPICAL NETWORK SENSOR SYSTEM.



**Bit Definition:**  
 CFCTN[1:0] : Command Function  
 NODE[4:0] : Node Select  
 Addresses up to 32 Nodes

**CFCTN[1:0] - Command Function Mode Bits**

M(1)	M(0)	MODE DESCRIPTION
0	0	NODE BROADCAST FRAME (ND_BC)
0	1	NODE WRITE W/O ACK FRAME (ND_NAK)
1	0	NODE WRITE W/ ACK FRAME (ND_WAK)
1	1	NODE READ FRAME (ND_RD)

FIGURE 4.1.1.2.2: NETWORK COMMAND BYTE (CMD).

The Network Layer Transport Byte (NWLTB) is a special system user defined byte that is application specific. It allows for 256 user defined modes. It is the gateway between data link layer and application layer, thus facilitating the peer to peer communication in the Subnet.



Bit Definition:

E : Error

Logic "0": Error not detected

Logic "1": Error detected

CFCTN[1:0] : Command Function

NODE[4:0] : Node Select

Addresses up to 32 Nodes

CFCTN[1:0] - Command Function Mode Bits

M(1)	M(0)	MODE DESCRIPTION
0	0	NODE BROADCAST FRAME (ND_BC)
0	1	NODE WRITE W/O ACK FRAME (ND_NAK)
1	0	NODE WRITE W/ ACK FRAME (ND_WAK)
1	1	NODE READ FRAME (ND_RD)

**FIGURE 4.1.1.2.3: NETWORK STATUS BYTE (STAT).**

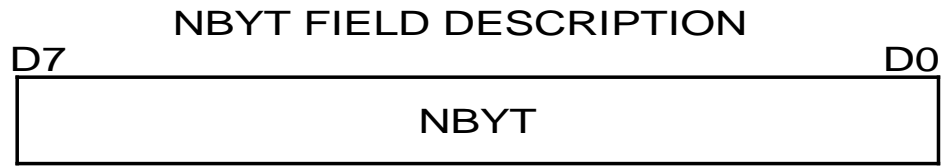


Bit Definition:

ADD[7:0] : Address Select Field

Allows for accessing up to 256 Node functions

**FIGURE 4.1.1.2.4: ADDRESS BYTE (ADD).**



Bit Definition:

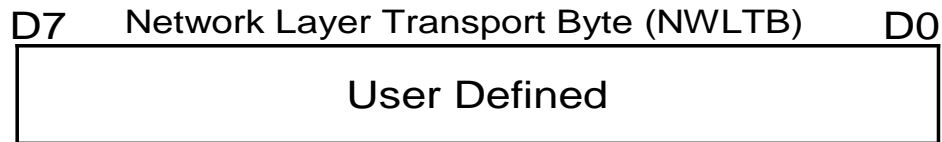
NBYT[7:0] : Number of Bytes

Allows for transfer, from 1 to 256, data bytes, located in DATA payload field.

\$00h – represents 1 byte to be transferred

\$FFh – represents 256 bytes to be transferred

**FIGURE 4.1.1.2.5: NUMBER OF BYTES FIELD (NBYT).**



Bit Definition:

User Defined[7:0] : This register allows the user to transport application specific content (command/ status) between peers.

**FIGURE 4.1.1.2.6: NETWORK LAYER TRANSPORT BYTE (NWLTB).**

### **4.1.1.3 REGISTER DESCRIPTION**

The MDSCC3208A comprises 11 registers.

**Table 4.1.1.3: Register Summary**

INTERFACE UNIT	REGISTER NEUMONIC	BYTE ADDRESS	CYCLE TYPE	DESCRIPTION
<TXMOD> (Transmitter Module)	TXPFR	\$0h	WO	TX Pointer Flush Register
	CBWR	\$1h	WO	Control Buffer Write Register
	DBWR	\$2h	WO	Data Buffer Write Register
	ETXBWR	\$3h	WO	Execute TX Buffer Write Register
<NODECG> (Node Configuration Module)	NCR	\$4h	R/W	Node Control Register
	SOFR	\$5h	R/W	Start of Frame Register
	EOFR	\$6h	R/W	End of Frame Register
	NSR	\$7h	R/W	Node Status Register
<RXMOD> (Receiver Module)	RXPFR	\$8h	WO	RX Pointer Flush Register
	CBRR	\$8h	RO	Control Buffer Read Register
	DBRR	\$9h	RO	Data Buffer Read Register

#### **4.1.1.3.1 TRANSMITTER MODULE (TXMOD)**

The TXMOD register set comprises a Transmit Buffer Pointer Flush write only (WO) register for initializing the both control and data buffer QUE pointers to 0. The Control QUE holds 4 control bytes, and the Data QUE holds up to 256 bytes.

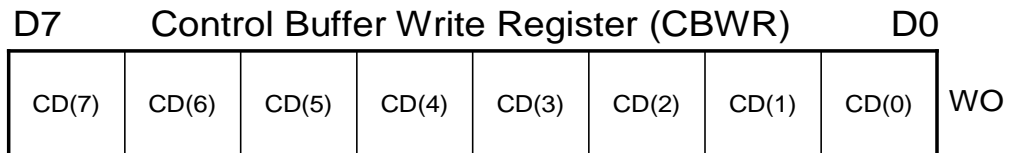
Both Control bytes (CBs) and Data bytes (DBs) are sequentially loaded at CBWR and DBWR respectively.

Once the CB and desired number of DBs are loaded, transmission is executed by writing the Execute TX Buffer Write Register (ETXBWR). The ETXBWR is a write strobe command, thus data is don't care.



Bit Definition:  
 Xs – Don't cares, write only register.  
 Writing to this register causes the Transmit Control and Data Buffers to be reset to \$00h.

**FIGURE 4.1.1.3.1.1: TX BUFFER QUE POINTER FLUSH REGISTER.**



Bit Definition:  
 CD[7:0] – Control Data Byte value  
 Writing to this register causes the Control Buffer Pointer to load current control byte, followed by an auto-increment of the control buffer pointer.

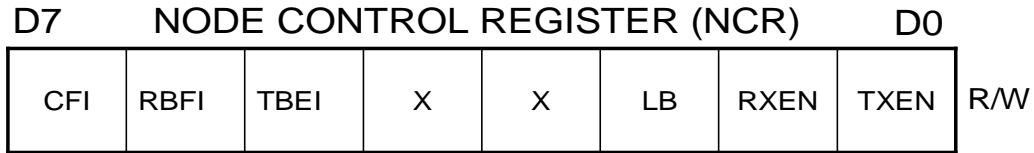
Upon resetting the pointer to \$00h, the user performs the following Byte write sequence to the CBWR:

- Write Cycle 1: CMD Byte
- Write Cycle 2: ADD Byte
- Write Cycle 3: NBYT Byte
- Write Cycle 4: NWLTB Byte

**FIGURE 4.1.1.3.1.2: CONTROL BUFFER QUE WRITE REGISTER.**



EOF power up with \$0Fh and \$FFh, respectively. All semaphore process handling, as well as PHY transceiver control, is performed using the Node Control Register (NCR) and Node Status Register (NSR).

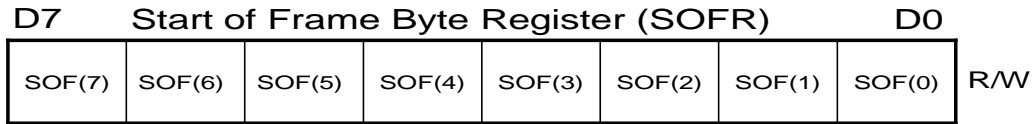


Bit Definition:  
 TXEN – Transmitter Enable  
           Logic “0”: Transmitter disabled  
           Logic “1”: Transmitter enabled  
 RXEN – Receiver Enable  
           Logic “0”: Receiver disabled  
           Logic “1”: Receiver enabled  
 LB – Loop Back  
           Logic “0”: Disabled  
           Logic “1”: Enabled  
 Xs – Don’t Cares (Spares for future use)  
 TBEI – Transmit Buffer Empty Interrupt Enable  
           Logic “0”: Disabled  
           Logic “1”: Enabled  
 RBFi – Receive Buffer Full Interrupt Enable  
           Logic “0”: Disabled  
           Logic “1”: Enabled  
 CFI – Catastrophic Failure Interrupt Enable  
           Logic “0”: Disabled  
           Logic “1”: Enabled

NOTE: For loop back mode, LB and RXEN must be set.

Power Up Value: \$00h

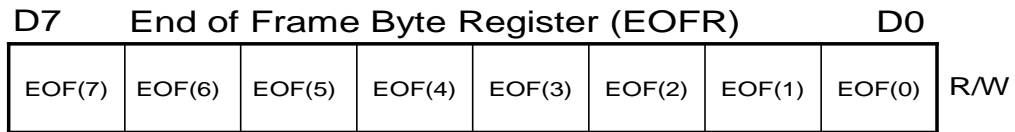
**FIGURE 4.1.1.3.2.1: NODE CONTROL REGISTER (NCR).**



Bit Definition:  
 SOF[7:0] – Start of Frame Sync Byte Value

Power Up Value: \$0Fh

**FIGURE 4.1.1.3.2.2: START OF FRAME BYTE SYNC REGISTER (SOFR).**



**Bit Definition:**  
 EOF[7:0] – End of Frame Sync Byte Value

Power Up Value: \$FFh

**FIGURE 4.1.1.3.2.3: END OF FRAME BYTE SYNC REGISTER (EOFR).**



**Bit Definition:**  
 CRC – CRC Failure  
     Logic “0”: False  
     Logic “1”: True  
 FE – Framing Error  
     Logic “0”: False  
     Logic “1”: True  
 SRED – SLV Response Error Detect  
     Logic “0”: False  
     Logic “1”: True  
 RXBF – Receive Buffer Full  
     Logic “0”: Not Full  
     Logic “1”: Full  
 TXBE – Transmit Buffer Empty  
     Logic “0”: Not Empty  
     Logic “1”: Empty  
 LOCK – Lock Status  
     Logic “0”: Loss of Lock  
     Logic “1”: Locked  
 X – Spares for future use

**NOTES:** (1) If status bit is set, indicating pending condition, then clearing status bit will negate pending interrupt state.  
 (2) Catastrophic Failure, CF = CRC + FE + SRED.  
 (3) Status for CRC, FE, and SRED is only valid upon RXBF being set.

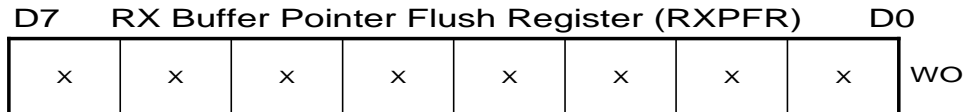
Power Up Value: \$04h

**FIGURE 4.1.1.3.2.4: NODE STATUS REGISTER (NSR).**

### 4.1.1.3.3 RECEIVER MODULE (RXMOD)

The RXMOD register set comprises a Receive Buffer Pointer Flush write only (WO) register for initializing both control and data read buffer QUE pointers to 0. The Control QUE holds 4 control bytes, and the Data QUE holds up to 256 bytes.

Both Control bytes (CBs) and Data bytes (DBs) are sequentially read at CBRR and DBRR respectively.

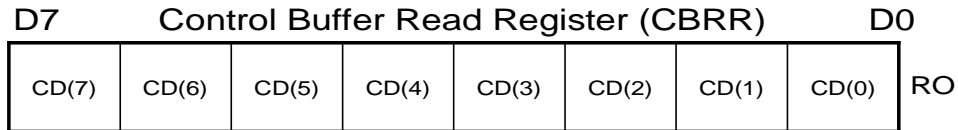


**Bit Definition:**

Xs – Don't cares, write only register.

Writing to this register causes both the Receive Control and Data Buffer Pointers to be reset to \$00h.

**FIGURE 4.1.1.3.3.1: RX BUFFER QUE POINTER FLUSH REGISTER.**



**Bit Definition:**

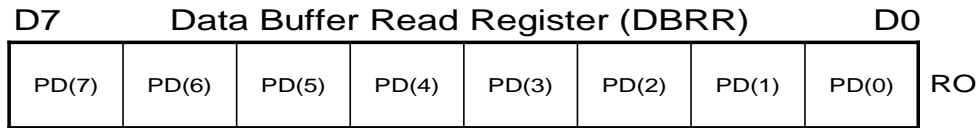
CD[7:0] – Control Data Byte value

Reading from this register causes the Control Buffer Pointer to load current control byte, followed by an auto-increment of the control buffer pointer.

Upon resetting the pointer to \$00h, the user performs the following Byte read sequence to the CBRR:

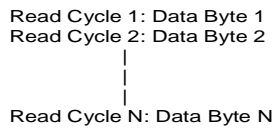
- Read Cycle 1: CMD Byte
- Read Cycle 2: ADD Byte
- Read Cycle 3: NBYT Byte
- Read Cycle 4: NWLTB Byte

**FIGURE 4.1.1.3.3.2: RX CONTROL BUFFER READ REGISTER.**



**Bit Definition:**  
 PD[7:0] – Payload Data byte value  
 Reading from this register causes the Data Buffer Pointer to load current data byte, followed by an auto-increment of the data buffer pointer.

Upon resetting the pointer to \$00h, the user performs the following Byte read sequence to the DBRR:

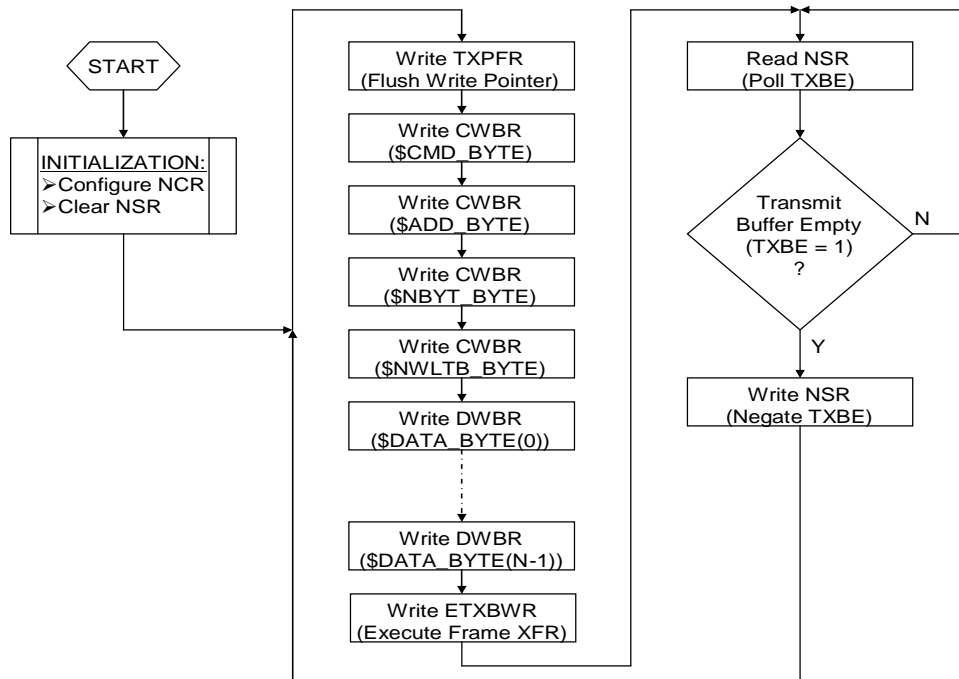


**NOTE:** N cannot exceed 256

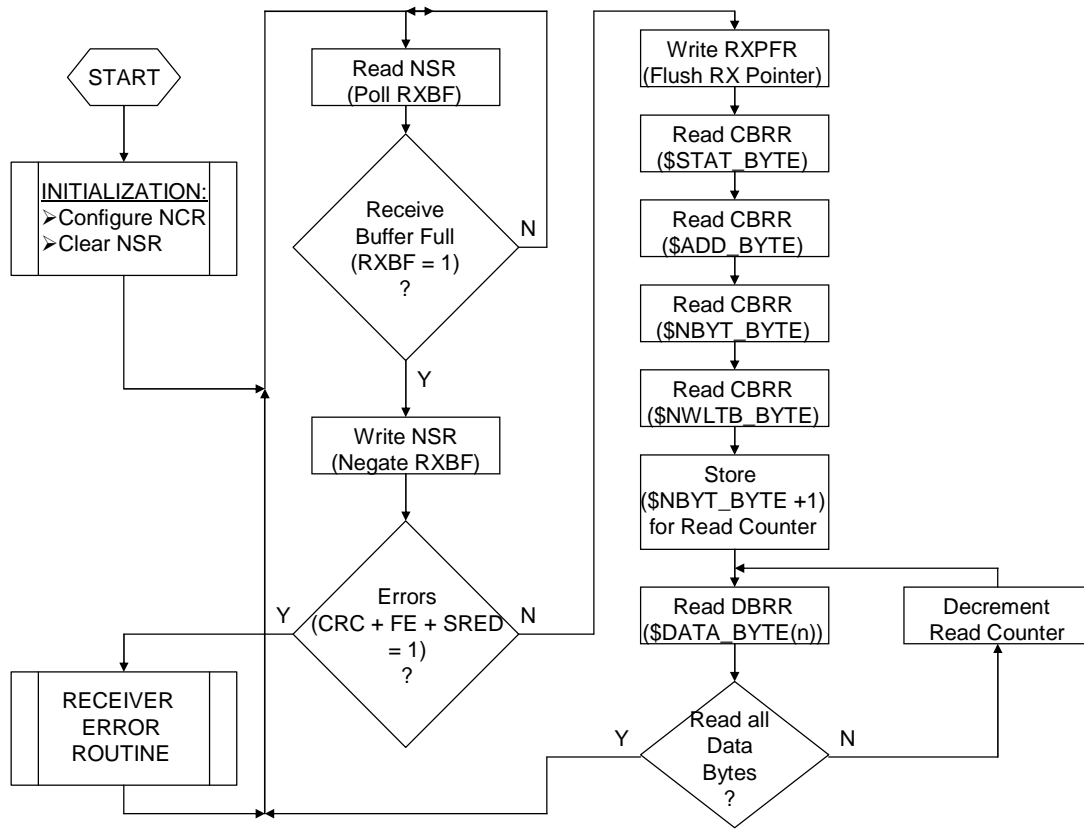
**FIGURE 4.1.1.3.3.3: RX DATA BUFFER READ REGISTER.**

**4.1.1.3.4 APPLICATION PROGRAMMING**

This section provides various application programming examples, using flowcharts and pseudo code.



**FIGURE 4.1.1.3.4.1: TRANSMIT BUFFER ROUTINE.**



**FIGURE 4.1.1.3.4.2: RECEIVER BUFFER ROUTINE.**

## **4.2 MTBNIC REGISTER DESCRIPTION**

Following is a summary of MTBNIC register space definition.

Long word 32-bit I/O Offset Address	Byte Lane 3 A[1:0] = 11 (Zone 3)	Byte Lane 2 A[1:0] = 10 (Zone 2)	Byte Lane 1 A[1:0] = 01 (Zone 1)	Byte Lane 0 A[1:0] = 00 (Zone 0)
\$0h	TX Buffer Pointer Flush Register (TXPFR3) <WO>	TX Buffer Pointer Flush Register (TXPFR2) <WO>	TX Buffer Pointer Flush Register (TXPFR1) <WO>	TX Buffer Pointer Flush Register (TXPFR0) <WO>
\$4h	Control Buffer Write Register (CBWR3) <WO>	Control Buffer Write Register (CBWR2) <WO>	Control Buffer Write Register (CBWR1) <WO>	Control Buffer Write Register (CBWR0) <WO>
\$8h	Data Buffer Write Register (DBWR3) <WO>	Data Buffer Write Register (DBWR2) <WO>	Data Buffer Write Register (DBWR1) <WO>	Data Buffer Write Register (DBWR0) <WO>
\$0Ch	Execute TX Buffer Write Register (ETXBWR3) <WO>	Execute TX Buffer Write Register (ETXBWR2) <WO>	Execute TX Buffer Write Register (ETXBWR1) <WO>	Execute TX Buffer Write Register (ETXBWR0) <WO>
\$10h	Node Control Register (NCR3) <R/W>	Node Control Register (NCR2) <R/W>	Node Control Register (NCR1) <R/W>	Node Control Register (NCR0) <R/W>
\$14h	Start of Frame Register (SOFR3) <R/W>	Start of Frame Register (SOFR2) <R/W>	Start of Frame Register (SOFR1) <R/W>	Start of Frame Register (SOFR0) <R/W>
\$18h	End of Frame Register (EOFR3) <R/W>	End of Frame Register (EOFR2) <R/W>	End of Frame Register (EOFR1) <R/W>	End of Frame Register (EOFR0) <R/W>
\$1Ch	Node Status Register (NSR3) <R/W>	Node Status Register (NSR2) <R/W>	Node Status Register (NSR1) <R/W>	Node Status Register (NSR0) <R/W>
\$20h	RX Buffer Pointer Flush Register (RXPFR3) <WO>	RX Buffer Pointer Flush Register (RXPFR2) <WO>	RX Buffer Pointer Flush Register (RXPFR1) <WO>	RX Buffer Pointer Flush Register (RXPFR0) <WO>
\$20h	Control Buffer Read Register (CBRR3) <RO>	Control Buffer Read Register (CBRR2) <RO>	Control Buffer Read Register (CBRR1) <RO>	Control Buffer Read Register (CBRR0) <RO>
\$24h	Data Buffer Read Register (DBRR3) <RO>	Data Buffer Read Register (DBRR2) <RO>	Data Buffer Read Register (DBRR1) <RO>	Data Buffer Read Register (DBRR0) <RO>

**FIGURE 4.2.1: MTBNIC PCI REGISTER SPACE DESCRIPTION.**

## **SECTION 5**

# **USER INTERCONNECT & SWITCHES**

## 5.1 OVERVIEW

The MTBNIC comprises several interconnection requirements, as well as supervisory switches, that the user must gain full knowledge of. These interconnections include IEEE 1149.1 JTAG I/F for FPGA programming, Interface mezzanine PWB connections, Isolated 3.3 VDC transceiver power connection, zone signal interface connections, and finally, PCI Slot selection Dip Switch and Momentary Reset Switch.

## 5.2 CONNECTOR DESCRIPTION

PIN \ ROW	A	B	C	D
1	DGND	RESERVED	+5V	AD0
2	VIO	AD2	AD1	+5V
3	AD5	DGND	AD4	AD3
4	CBE <sub>n0</sub>	AD7	DGND	AD6
5	DGND	AD9	AD8	DGND
6	AD11	VIO	AD10	M66EN
7	AD14	AD13	DGND	AD12
8	+3.3V	CBE <sub>n1</sub>	AD15	+3.3V
9	SERR <sub>n</sub>	DGND	RESERVED	PAR
10	DGND	PERR <sub>n</sub>	+3.3V	RESERVED
11	STOP <sub>n</sub>	+3.3V	LOCK <sub>n</sub>	DGND
12	+3.3V	TRDY <sub>n</sub>	DGND	DEVSEL <sub>n</sub>
13	FRAME <sub>n</sub>	DGND	IRDY <sub>n</sub>	+3.3V
14	DGND	AD16	+3.3V	CBE <sub>n2</sub>
15	AD18	+3.3V	AD17	DGND
16	AD21	AD20	DGND	AD19
17	+3.3V	AD23	AD22	+3.3V
18	IDSEL0	DGND	IDSEL1	IDSEL2
19	AD24	CBE <sub>n3</sub>	VIO	IDSEL3
20	DGND	AD26	AD25	DGND
21	AD29	+5V	AD28	AD27
22	+5V	AD30	DGND	AD31
23	REQ <sub>n0</sub>	DGND	REQ <sub>n1</sub>	VIO
24	DGND	REQ <sub>n2</sub>	+5V	GNT <sub>n0</sub>
25	GNT <sub>n1</sub>	VIO	GNT <sub>n2</sub>	DGND
26	+5V	CLK0	DGND	CLK1
27	CLK2	+5V	CLK3	DGND
28	DGND	INTD <sub>n</sub>	+5V	RST <sub>n</sub>
29	+12V	INTA <sub>n</sub>	INTB <sub>n</sub>	INTC <sub>n</sub>
30	-12V	REQ <sub>n3</sub>	GNT <sub>n3</sub>	DGND

FIGURE 5.2.1: PC104+ PCI CONNECTOR, J1, SIGNAL DEFINITION.

1	3.3VZ_RTN	2	TXDO_P1
3	TXDZ1	4	TXDO_N1
5	TXDENZ1	6	3.3VZ_RTN
7	RXDZ1	8	RXDI_P1
9	RXDEZn1	10	RXDI_N1
11	3.3VZ	12	CGND

1	3.3VZ_RTN	2	TXDO_P2
3	TXDZ2	4	TXDO_N2
5	TXDENZ2	6	3.3VZ_RTN
7	RXDZ2	8	RXDI_P2
9	RXDEZn2	10	RXDI_N2
11	3.3VZ	12	CGND

1	3.3VZ_RTN	2	TXDO_P0
3	TXDZ0	4	TXDO_N0
5	TXDENZ0	6	3.3VZ_RTN
7	RXDZ0	8	RXDI_P0
9	RXDEZn0	10	RXDI_N0
11	3.3VZ	12	CGND

1	3.3VZ_RTN	2	TXDO_P3
3	TXDZ3	4	TXDO_N3
5	TXDENZ3	6	3.3VZ_RTN
7	RXDZ3	8	RXDI_P3
9	RXDEZn3	10	RXDI_N3
11	3.3VZ	12	CGND

**FIGURE 5.2.2: MEZZANINE CONNECTOR SIGNAL DEFINITION.**

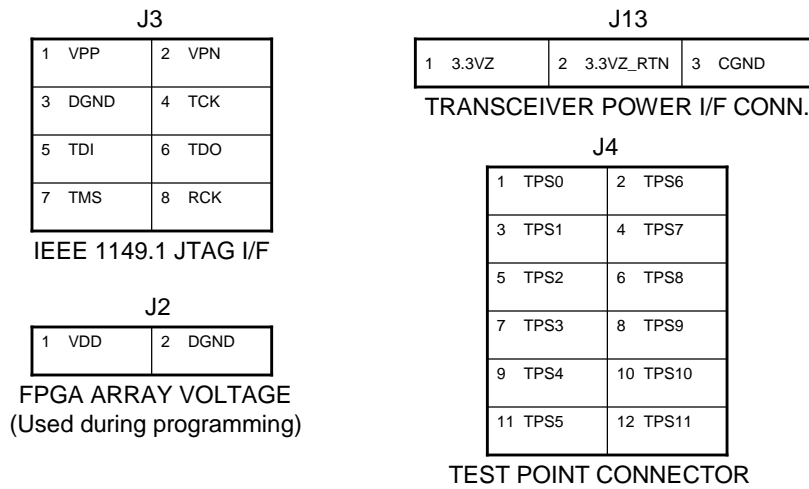
1	3.3VZ_RTN	2	TXDO_P1
3	TOKEN1	4	TXDO_N1
5	TOKEN01	6	RXDI_P1
7	3.3VZ_RTN	8	RXDI_N1

1	3.3VZ_RTN	2	TXDO_P2
3	TOKEN2	4	TXDO_N2
5	TOKEN02	6	RXDI_P2
7	3.3VZ_RTN	8	RXDI_N2

1	3.3VZ_RTN	2	TXDO_P0
3	TOKEN0	4	TXDO_N0
5	TOKEN00	6	RXDI_P0
7	3.3VZ_RTN	8	RXDI_N0

1	3.3VZ_RTN	2	TXDO_P3
3	TOKEN3	4	TXDO_N3
5	TOKEN03	6	RXDI_P3
7	3.3VZ_RTN	8	RXDI_N3

**FIGURE 5.2.3: ZONAL CONNECTOR SIGNAL DEFINITION.**

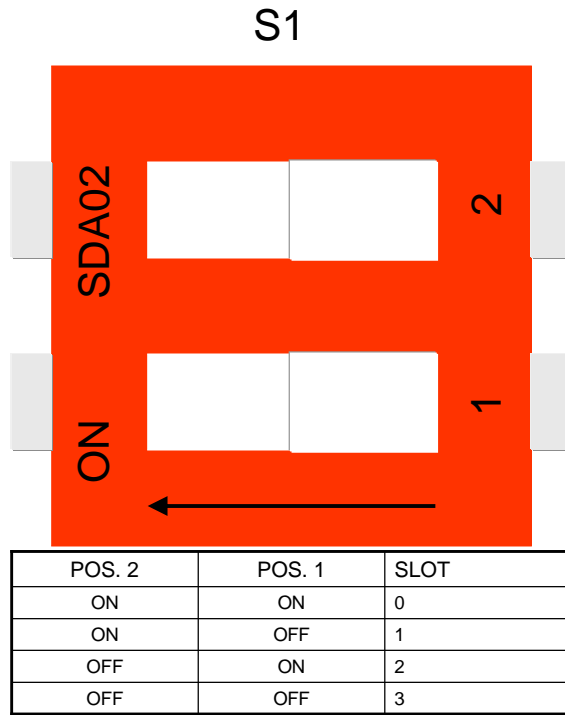


**FIGURE 5.2.4: MISCELLANEOUS CONNECTOR SIGNAL DEFINITION.**

### **5.3 SWITCH DESCRIPTION**

The MTBNIC comprises two switches: PCI Slot Selector switch, S1, and momentary reset switch, SW1.

Reset switch, SW1, resets the Quad SCC FPGA logic only, not PCI Target registers, which allows user to continue any debugging without having to wait for a system boot up. The reset time is 200 milliseconds.



**FIGURE 5.3.1: PCI SLOT SELECTOR SWITCH DEFINITION.**