TNETX4080
ThunderSWITCH II™ 8-PORT 10-/100-MBIT/S ETHERNET™ SWITCH

SPWS050A – AUGUST 1998 – REVISED NOVEMBER 1998

- **Single-Chip 100-Mbit/s Device**
- **Integrated Address-Lookup Engine and Table Memory for 1-K Addresses**
- **Supports IEEE Std 802.1Q Virtual-LAN (VLAN) Tagging Scheme**
- **Provides Data Path for Network Management Information [No External Media-Access Control (MAC) Required]**
- **Full-Duplex IEEE Std 802.3 Flow Control**
- **Half-Duplex Back-Pressure Flow Control**
- **Fully Nonblocking Architecture Using High-Bandwidth Rambus Memory**
- **Port Trunking/Load Sharing for High-Bandwidth Interswitch Links**
- **EEPROM Interface for Autoconfiguration (No CPU Required for Nonmanaged Switch)**
- **Provides Direct Input/Output (DIO) Interface for Configuration and Statistics Information**
- **Supports On-Chip Per-Port Storage for Etherstat™ and Remote Monitoring (RMON) Management Information Bases (MIBs)**
- **Fabricated in 2.5-/3.3-V Low-Voltage Technology**
- **Supports Spanning Tree**
- **Packaged in 352-Pin Ball Grid Array Package**

description

The TNETX4080 is an 8-port 100-Mbit/s nonblocking Ethernet™ switch with an on-chip address-lookup engine. The TNETX4080 provides a low-cost, high-performance switch solution. The TNETX4080 is a fully manageable desktop switch solution achieved by combining the TNETX4080 with physical interfaces and high-bandwidth rambus-based packet memory and a CPU. The TNETX4080 also provides an interface capable of receiving and transmitting simple-network management protocol (SNMP) and bridge protocol data units (BPDU) (spanning-tree) frames.

The TNETX4080 provides eight 10-/100-Mbit/s interfaces. In half-duplex mode, all ports support back-pressure flow control to reduce the risk of data loss for a long burst of activity. In the full-duplex mode of operation, the device uses IEEE Std 802.3 frame-based flow control. With full-duplex capability, ports 0–7 support 200-Mbit/s aggregate bandwidth connections. The TNETX4080 also supports port trunking/load sharing on the 10-/100-Mbit ports. This can be used to group ports on interswitch links to increase the effective bandwidth between the systems.

The internal address-lookup engine (IALE) supports up to 1-K unicast/multicast and broadcast addresses and up to 16 IEEE Std 802.1Q VLANs. For interoperability, each port can be programmed as an access port or non-access port to recognize VLAN tags and transmit frames with VLAN tags to other systems that support VLAN tagging. The IALE performs destination- and source-address comparisons and forwards unknown source- and destination-address packets to ports specified via programmable masks.

Statistics for the Etherstat, SNMP, and remote monitoring management information base (RMON MIB) are independently collected for each of the eight ports. Access to the statistics counters is provided via the direct input/output (DIO) interface. Management frames can be received and transmitted via the DIO interface, creating a complete network management solution. Figure 1 is a block diagram of the TNETX4080.

PRODUCT PREVIEW

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

TI, ThunderSWITCH, and ThunderSWITCH II are trademarks of Texas Instruments Incorporated.
Ethernet and Etherstat are trademarks of Xerox Corporation.
Secure Fast Switching is a trademark of Cabletron Systems, Inc.
Port-trunking and load-sharing algorithms were contributed by Cabletron Systems, Inc. and are derived from, and compatible with, Secure Fast Switching™.

PRODUCT PREVIEW information concerns products in the formative or design phase of development. Characteristic data and other specifications are design goals. Texas Instruments reserves the right to change or discontinue these products without notice.

description (continued)

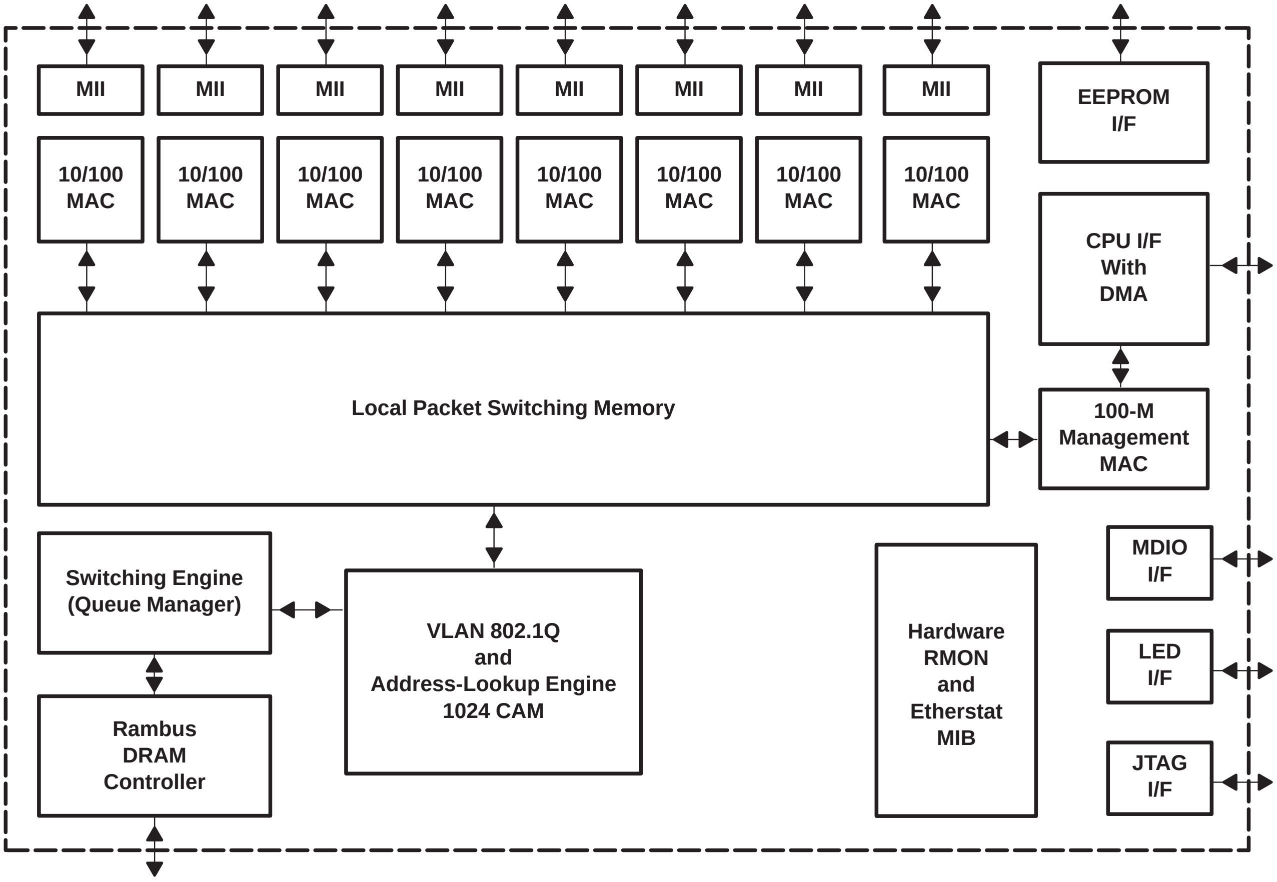

MII
10/100 MAC
EEPROM I/F
CPU I/F With DMA
Local Packet Switching Memory
100-M Management MAC
MDIO I/F
Switching Engine (Queue Manager)
VLAN 802.1Q and Address-Lookup Engine 1024 CAM
Hardware RMON and Etherstat MIB
LED I/F
JTAG I/F
Rambus DRAM Controller

Figure 1. TNETX4080 Block Diagram

The TNETX4080 memory solution combines low cost and extremely high bandwidth, using 600-Mbit/pin/s concurrent RDRAM. The packet memory has been implemented to maximize efficiency with the RDRAM architecture. Data is buffered internally and transferred to/from packet memory in 128-byte bursts. Extremely high memory bandwidth is maintained, allowing all ports to be active without bottlenecking at the memory buffer.

The TNETX4080 is fabricated with a 2.5-V technology. The inputs are 3.3-V tolerant and the outputs are capable of directly interfacing to TTL levels. This provides the customer with a broad choice of interfacing device options.

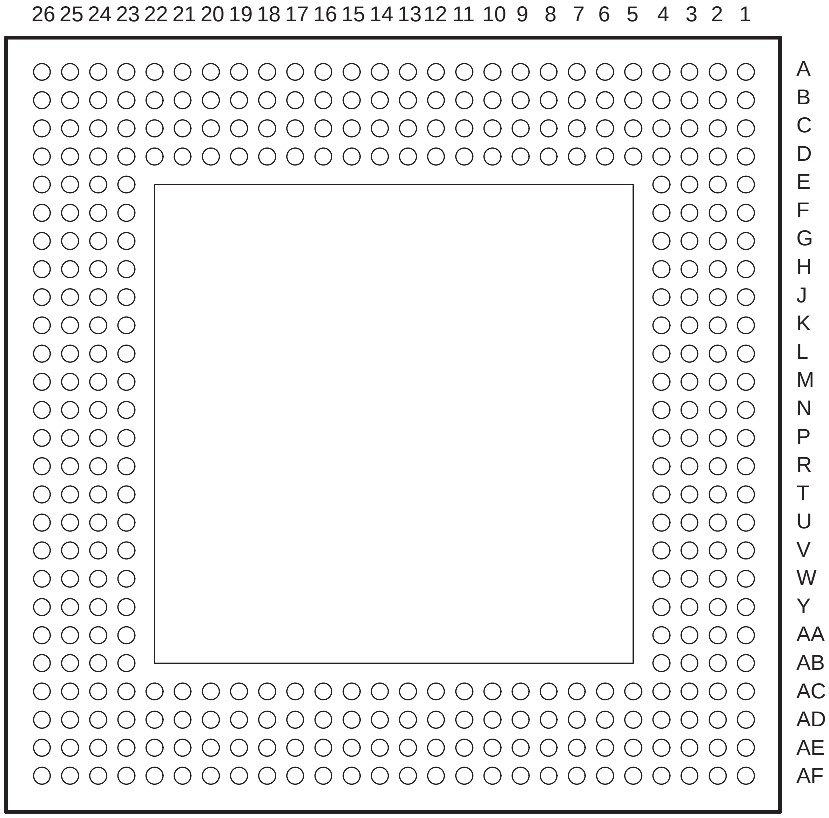
Signal names and their terminal assignments are sorted alphabetically in Table 1.

PRODUCT PREVIEW

Contents

PRODUCT PREVIEW

GGP PACKAGE
(BOTTOM VIEW)



PRODUCT PREVIEW

Table 1. Signal-to-Ball Mapping (Signal Names Sorted Alphabetically)

SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.
DBUS_CTL	Y26	GND	AF26	M03_CRS	B6	M06_RXD2	Y3	MDIO	K25
DBUS_DATA0	AC26	GNDa	U24	M03_LINK	A4	M06_RXD3	Y4	$\overline{\text{MRESET}}$	K24
DBUS_DATA1	AA24	NC	AE18	M03_RCLK	C6	M06_RXDV	W1	NC	A3
DBUS_DATA2	AB26	LED_CLK	AD19	$\overline{\text{M03\_RENEG}}$	C1	M06_RXER	AA2	NC	A24
DBUS_DATA3	Y24	$\overline{\text{LED\_DATA}}$	AE19	M03_RXD0	A5	M06_TCLK	T1	NC	C23
DBUS_DATA4	V24	M00_COL	C21	M03_RXD1	B5	M06_TXD0	U1	NC	D2
DBUS_DATA5	U25	M00_CRS	B21	M03_RXD2	C5	M06_TXD1	U2	NC	D3
DBUS_DATA6	U26	M00_LINK	B19	M03_RXD3	D5	M06_TXD2	U3	NC	D6
DBUS_DATA7	T26	M00_RCLK	A21	M03_RXDV	C4	M06_TXD3	U4	NC	D24
DBUS_DATA8	R25	$\overline{\text{M00\_RENEG}}$	C26	M03_RXER	B4	M06_TXEN	V3	NC	D25
DBUS_EN	T25	M00_RXD0	A20	M03_TCLK	A8	M06_TXER	V2	NC	E1
DCCTRL	P24	M00_RXD1	B20	M03_TXD0	A7	M07_COL	AC6	NC	E2
DRX_CLK	V26	M00_RXD2	C20	M03_TXD1	B7	M07_CRS	AD6	NC	E3
DTX_CLK	V25	M00_RXD3	D20	M03_TXD2	C7	M07_LINK	AF3	NC	E4
DVREF	AA26	M00_RXDV	D19	M03_TXD3	D7	M07_RCLK	AE6	NC	E23
ECLK	L26	M00_RXER	C19	M03_TXEN	B8	$\overline{\text{M07\_RENEG}}$	AD1	NC	E24
EDIO	M26	M00_TCLK	B23	M03_TXER	C8	M07_RXD0	AF7	NC	E25
FLOW	AF8	M00_TXD0	A23	M04_COL	H2	M07_RXD1	AE7	NC	E26
GND	A1	M00_TXD1	A22	M04_CRS	H1	M07_RXD2	AD7	NC	F4
GND	A2	M00_TXD2	B22	M04_LINK	L3	M07_RXD3	AC7	NC	F23
GND	A13	M00_TXD3	C22	M04_RCLK	J3	M07_RXDV	AC8	NC	F24
GND	A14	M00_TXEN	D22	$\overline{\text{M04\_RENEG}}$	F3	M07_RXER	AD8	NC	F25
GND	A25	M00_TXER	D21	M04_RXD0	J1	M07_TCLK	AD4	NC	F26
GND	A26	M01_COL	D16	M04_RXD1	K1	M07_TXD0	AF5	NC	G23
GND	AF13	M01_CRS	C16	M04_RXD2	K2	M07_TXD1	AE5	NC	G24
GND	AF14	M01_LINK	B14	M04_RXD3	K3	M07_TXD2	AD5	NC	G25
GND	B1	M01_RCLK	B16	M04_RXDV	J2	M07_TXD3	AC5	NC	G26
GND	B3	$\overline{\text{M01\_RENEG}}$	D26	M04_RXER	L4	M07_TXEN	AE4	NC	H24
GND	B24	M01_RXD0	A15	M04_TCLK	F1	M07_TXER	AF4	NC	H25
GND	B26	M01_RXD1	B15	M04_TXD0	G1	M08_COL	AD12	NC	H26
GND	C2	M01_RXD2	C15	M04_TXD1	G2	M08_CRS	AC12	NC	J24
GND	C25	M01_RXD3	D15	M04_TXD2	G3	M08_EWRAP	AE13	NC	J25
GND	N1	M01_RXDV	A16	M04_TXD3	G4	M08_GTCLK	AF17	NC	J26
GND	N26	M01_RXER	C14	M04_TXEN	H4	NC	AE17	NC	K23
GND	P1	M01_TCLK	C17	M04_TXER	H3	NC	AF12	NC	N23
GND	P25	M01_TXD0	A19	M05_COL	N2	NC	AC17	NC	N24
GND	P26	M01_TXD1	A18	M05_CRS	P3	NC	AD17	NC	N25
GND	R23	M01_TXD2	B18	M05_LINK	T2	NC	AE12	NC	AA4
GND	R24	M01_TXD3	C18	M05_RCLK	P2	NC	AD13	NC	AB1
GND	R26	M01_TXEN	B17	$\overline{\text{M05\_RENEG}}$	F2	NC	AF9	NC	AB2
GND	T24	M01_TXER	A17	M05_RXD0	R1	NC	AE9	NC	AB3
GND	U23	M02_COL	C11	M05_RXD1	R2	NC	AD9	NC	AB4
GND	W23	M02_CRS	B11	M05_RXD2	R3	NC	AF10	NC	AB23
GND	W24	M02_LINK	A9	M05_RXD3	R4	NC	AE10	NC	AB24
GND	W25	M02_RCLK	A11	M05_RXDV	T4	NC	AD10	NC	AC1
GND	W26	$\overline{\text{M02\_RENEG}}$	D1	M05_RXER	T3	NC	AF11	NC	AC2
GND	Y23	M02_RXD0	A10	M05_TCLK	L2	NC	AE11	NC	AC3
GND	Y25	M02_RXD1	B10	M05_TXD0	M1	NC	AD11	NC	AC24
GND	AA23	M02_RXD2	C10	M05_TXD1	M2	NC	AC11	NC	AC25
GND	AA25	M02_RXD3	D10	M05_TXD2	M3	NC	AF16	NC	AD18
GND	AB25	M02_RXDV	C9	M05_TXD3	M4	NC	AE16	NC	AD26
GND	AD2	M02_RXER	B9	M05_TXEN	L1	NC	AD16	NC	AE8
GND	AD25	M02_TCLK	C13	M05_TXER	N3	NC	AC16	NC	AF6
GND	AE1	M02_TXD0	A12	M06_COL	V1	NC	AF15	NC	AF18
GND	AE3	M02_TXD1	B12	M06_CRS	W3	NC	AE15		
GND	AE24	M02_TXD2	C12	M06_LINK	AA1	NC	AD15		
GND	AE26	M02_TXD3	D12	M06_RCLK	W2	NC	AC15		
GND	AF1	M02_TXEN	B13	$\overline{\text{M06\_RENEG}}$	AA3	NC	AE14		
GND	AF2	M02_TXER	D11	M06_RXD0	Y1	NC	AD14		
GND	AF25	M03_COL	A6	M06_RXD1	Y2	MDCLK	K26		

PRODUCT PREVIEW

Table 1. Signal-to-Ball Mapping (Signal Names Sorted Alphabetically) (Continued)

SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.	SIGNAL NAME	BALL NO.
$\overline{\text{RESET}}$	M23	$\overline{\text{SDMA}}$	AF24	$V_{DD(2.5)}$	B25	$V_{DD(2.5)}$	V4	$V_{DD(3.3)}$	D13
SAD0	AF22	SINT	AF19	$V_{DD(2.5)}$	C3	$V_{DD(2.5)}$	V23	$V_{DD(3.3)}$	D17
SAD1	AE22	$\overline{\text{SRDY}}$	AF23	$V_{DD(2.5)}$	C24	$V_{DD(2.5)}$	AC4	$V_{DD(3.3)}$	H23
$\overline{\text{SCS}}$	AD22	SRNW	AC22	$V_{DD(2.5)}$	D4	$V_{DD(2.5)}$	AC9	$V_{DD(3.3)}$	K4
SDATA0	AF20	SRXRDY	AE23	$V_{DD(2.5)}$	D9	$V_{DD(2.5)}$	AC13	$V_{DD(3.3)}$	P4
SDATA1	AE20	STXRDY	AD23	$V_{DD(2.5)}$	D14	$V_{DD(2.5)}$	AC18	$V_{DD(3.3)}$	W4
SDATA2	AD20	TCLK	L24	$V_{DD(2.5)}$	D18	$V_{DD(2.5)}$	AC23	$V_{DD(3.3)}$	AC10
SDATA3	AC20	TDI	M24	$V_{DD(2.5)}$	D23	$V_{DD(2.5)}$	AD3	$V_{DD(3.3)}$	AC14
SDATA4	AF21	TDO	L23	$V_{DD(2.5)}$	J4	$V_{DD(2.5)}$	AD24	$V_{DD(3.3)}$	AC19
SDATA5	AE21	TMS	M25	$V_{DD(2.5)}$	J23	$V_{DD(2.5)}$	AE2	V_{DDa}(2.5)	T23
SDATA6	AD21	$\overline{\text{TRST}}$	L25	$V_{DD(2.5)}$	N4	$V_{DD(2.5)}$	AE25		
SDATA7	AC21	$V_{DD(2.5)}$	B2	$V_{DD(2.5)}$	P23	$V_{DD(3.3)}$	D8		

Terminal Functions

PRODUCT PREVIEW

IEEE Std 1149.1 (JTAG) interface

TERMINAL		I/O	INTERNAL RESISTOR	DESCRIPTION
NAME	NO.			
TCLK	L24	I	Pullup	Test clock. Clocks state information and test data into and out of the TNETX4080 during operation of the test port.
TDI	M24	I	Pullup	Test data input. Serially shifts test data and test instructions into the TNETX4080 during operation of the test port. An internal pullup resistor is provided on TDI to ensure IEEE Std 1149.1 (JTAG) compliance.
TDO	L23	O	None	Test data out. Serially shifts test data and test instructions out of the TNETX4080 during operation of the test port.
TMS	M25	I	Pullup	Test mode select. Controls the state of the test-port controller. An internal pullup resistor is provided on TMS to ensure IEEE Std 1149.1 (JTAG) compliance.
$\overline{\text{TRST}}$	L25	I	Pullup	Test reset. Asynchronously resets the test-port controller. An internal pullup resistor is provided on $\overline{\text{TRST}}$ to ensure IEEE Std 1149.1 (JTAG) compliance.

control logic interface

TERMINAL		I/O	DESCRIPTION
NAME	NO.		
$\overline{\text{RESET}}$	M23	I	Device reset. Asserted for a minimum of 100 µs after power supplies and clocks have stabilized. The system clock must be operational during reset.
FLOW	AF8	O	Flow control. When flow control is activated (flow in SysControl = 1) and the number of free external memory buffers is below the threshold indicated in FlowThreshold, FLOW is asserted.

Terminal Functions (Continued)

10-/100-Mbit/s MAC interface (MII mode) (ports 0–7)

TERMINAL		I/O	INTERNAL RESISTOR	DESCRIPTION
NAME	NO.			
M00_COL M01_COL M02_COL M03_COL M04_COL M05_COL M06_COL M07_COL	C21 D16 C11 A6 H2 N2 V1 AC6	I	Pulldown	Collision sense. Assertion of Mxx_COL indicates network collision. In full-duplex mode, the port does not start transmitting a new frame if this signal is active; the value of this pin is ignored at all other times.
M00_CRS M01_CRS M02_CRS M03_CRS M04_CRS M05_CRS M06_CRS M07_CRS	B21 C16 B11 B6 H1 P3 W3 AD6	I	Pulldown	Carrier sense. Indicates a frame-carrier signal is being received.
M00_LINK M01_LINK M02_LINK M03_LINK M04_LINK M05_LINK M06_LINK M07_LINK	B19 B14 A9 A4 L3 T2 AA1 AF3	I	Pulldown	Connection status. Indicates the presence of port connection: – If Mxx_LINK = 0, there is no link. – If Mxx_LINK = 1, the link is OK. An internal pullup resistor is provided.
M00_RCLK M01_RCLK M02_RCLK M03_RCLK M04_RCLK M05_RCLK M06_RCLK M07_RCLK	A21 B16 A11 C6 J3 P2 W2 AE6	I	Pullup	Receive clock. Receive clock source from the attached PHY device.
$\overline{\text{M00\_RENEG}}$ $\overline{\text{M01\_RENEG}}$ $\overline{\text{M02\_RENEG}}$ $\overline{\text{M03\_RENEG}}$ $\overline{\text{M04\_RENEG}}$ $\overline{\text{M05\_RENEG}}$ $\overline{\text{M06\_RENEG}}$ $\overline{\text{M07\_RENEG}}$	C26 D26 D1 C1 F3 F2 AA3 AD1	I	None	Renegotiate. Indicates to the attached PHY that this port wishes to renegotiate a new configuration.
M00_RXDV M01_RXDV M02_RXDV M03_RXDV M04_RXDV M05_RXDV M06_RXDV M07_RXDV	D19 A16 C9 C4 J2 T4 W1 AC8	I	Pulldown	Receive data valid. Indicates data on Mxx_RXD7–Mxx_RxD0. is valid. This signal is synchronous to Mxx_RCLK.

PRODUCT PREVIEW

Terminal Functions (Continued)

10-/100-Mbit/s MAC interface (MII mode) (ports 0–7) (continued)

TERMINAL NAME	TERMINAL NO.	I/O	INTERNAL RESISTOR	DESCRIPTION
M00_RXD3 M00_RXD2 M00_RXD1 M00_RXD0 M01_RXD3 M01_RXD2 M01_RXD1 M01_RXD0 M02_RXD3 M02_RXD2 M02_RXD1 M02_RXD0 M03_RXD3 M03_RXD2 M03_RXD1 M03_RXD0 M04_RXD3 M04_RXD2 M04_RXD1 M04_RXD0 M05_RXD3 M05_RXD2 M05_RXD1 M05_RXD0 M06_RXD3 M06_RXD2 M06_RXD1 M06_RXD0 M07_RXD3 M07_RXD2 M07_RXD1 M07_RXD0	D20 C20 B20 A20 D15 C15 B15 A15 D10 C10 B10 A10 D5 C5 B5 A5 K3 K2 K1 J1 R4 R3 R2 R1 Y4 Y3 Y2 Y1 AC7 AD7 AE7 AF7	I	Pullup	Receive data. Nibble receive data from the attached PHY device. Data on these signals is synchronous to Mxx_RCLK. When Mxx_RXDV and Mxx_RXER are low, these pins are sampled the cycle before Mxx_LINK goes high to configure the port, based on capabilities negotiated by the attached PHY device as follows: – Mxx_RXD0 indicates full-duplex mode when high; half duplex when low, and sets duplex in PortxStatus. – Mxx_RXD1 indicates IEEE Std 802.3 pause frame support when high; no pause when low, and sets pause in PortxStatus. – Mxx_RXD2 indicates 100 Mbit/s when high; 10 Mbit/s when low, and sets speed in PortxStatus. – Mxx_RXD3 is unused and is ignored.
M00_RXER M01_RXER M02_RXER M03_RXER M04_RXER M05_RXER M06_RXER M07_RXER	C19 C14 B9 B4 L4 T3 AA2 AD8	I	Pulldown	Receive error. Indicates reception of a coding error on received data.
M00_TCLK M01_TCLK M02_TCLK M03_TCLK M04_TCLK M05_TCLK M06_TCLK M07_TCLK	B23 C17 C13 A8 F1 L2 T1 AD4	I	Pullup	Transmit clock. Transmit clock source from the attached PHY or PMI device.

Terminal Functions (Continued)

10-/100-Mbit/s MAC interface (MII mode) (ports 0–7) (continued)

TERMINAL		I/O	INTERNAL RESISTOR	DESCRIPTION
NAME	NO.			
M00_TXD3 M00_TXD2 M00_TXD1 M00_TXD0 M01_TXD3 M01_TXD2 M01_TXD1 M01_TXD0 M02_TXD3 M02_TXD2 M02_TXD1 M02_TXD0 M03_TXD3 M03_TXD2 M03_TXD1 M03_TXD0 M04_TXD3 M04_TXD2 M04_TXD1 M04_TXD0 M05_TXD3 M05_TXD2 M05_TXD1 M05_TXD0 M06_TXD3 M06_TXD2 M06_TXD1 M06_TXD0 M07_TXD3 M07_TXD2 M07_TXD1 M07_TXD0	C22 B22 A22 A23 C18 B18 A18 A19 D12 C12 B12 A12 D7 C7 B7 A7 G4 G3 G2 G1 M4 M3 M2 M1 U4 U3 U2 U1 AC5 AD5 AE5 AF5	O	None	Transmit data. Byte transmit data. When Mxx_TXEN is asserted, these signals carry transmit data. Data on these signals is synchronous to Mxx_RFCLK. When Mxx_TXEN, Mxx_TXER, and Mxx_LINK are all low, these pins indicate the desired capabilities for autonegotiation as follows: – Mxx_TXD0 indicates full-duplex capability when high; half duplex when low, as determined by reqhd in PortxControl. – Mxx_TXD1 indicates IEEE Std 802.3 pause frame support when high; no pause when low, as determined by reqnp in PortxControl. – Mxx_TXD2 indicates 100 Mbit/s when high; 10 Mbit/s when low, as determined by req10 in PortxControl. – Mxx_TXD3 is unused and is 0.
M00_TXEN M01_TXEN M02_TXEN M03_TXEN M04_TXEN M05_TXEN M06_TXEN M07_TXEN	D22 B17 B13 B8 H4 L1 V3 AE4	O	None	Transmit enable. Indicates valid transmit data on Mxx_TXDn. This signal is synchronous to Mxx_RFCLK.
M00_TXER M01_TXER M02_TXER M03_TXER M04_TXER M05_TXER M06_TXER M07_TXER	D21 A17 D11 C8 H3 N3 V2 AF4	O	None	Transmit error. Allows coding errors to be propagated across the MII. Mxx_TXER is asserted at the end of an under-running frame, enabling the TNETX4080 to force a coding error.

PRODUCT PREVIEW

Terminal Functions (Continued)

MII management interface

TERMINAL NAME	TERMINAL NO.	I/O	INTERNAL RESISTOR	DESCRIPTION
MDCLK	K26	O	Pullup	Serial MII management data clock. Disabled [high-impedance (Z) state] through the use of the serial input/output (SIO) register. An internal pullup resistor is provided.
MDIO	K25	I/O	Pullup	Serial MII management data input/output. Disabled [high-impedance (Z) state] through the use of the SIO register. An internal pullup resistor is provided.
$\overline{\text{MRESET}}$	K24	O	Pullup	Serial MII management reset. Disabled [high-impedance (Z) state] through the use of the SIO register. An internal pullup resistor is provided.

RDRAM interface

TERMINAL NAME	TERMINAL NO.	I/O	INTERNAL RESISTOR	DESCRIPTION
DBUS_CTL	Y26	O	None	Bus control. Controls signal-to-frame packets, transmits part of the operation code, initiates data transfers, and terminates data transfers. This is a rambus signal logic (RSL) signal (see Note 1).
DBUS_DATA0 DBUS_DATA1 DBUS_DATA2 DBUS_DATA3 DBUS_DATA4 DBUS_DATA5 DBUS_DATA6 DBUS_DATA7 DBUS_DATA8	AC26 AA24 AB26 Y24 V24 U25 U26 T26 R25	I/O	None	Bus data. Signal lines for request, write-data, and read-data packets. The request packet contains the address, operation codes, and other control information. These are RSL signals (see Note 1).
DBUS_EN	T25	O	None	Bus enable. Controls signal-to-transfer column addresses for random-access (nonsequential) transactions. This is an RSL signal (see Note 1).
DCCTRL	P24	I	None	Current control program. Connected to the current control resistor whose other terminal is connected to the termination voltage.
DRX_CLK	V26	O	None	Receive clock. This signal is derived from DTX_CLK. This is an RSL signal (see Note 1). It is connected directly to DTX_CLK in the TNETX4080.
DTX_CLK	V25	I	None	Transmit clock. This is an RSL signal (see Note 1). The primary internal clock is derived from this signal.
DVREF	AA26	I	None	Reference voltage. Logic threshold reference voltage for RSL signals.

NOTE 1: RSL is a low-voltage swing, active-low signaling technology.

PRODUCT PREVIEW

Terminal Functions (Continued)

DIO interface

TERMINAL		I/O	INTERNAL RESISTOR	DESCRIPTION
NAME	NO.			
SAD0 SAD1	AF22 AE22	I	Pullup	DIO address bus. Selects the internal host registers provided $\overline{\text{SDMA}}$ is high. Internal pullup resistors are provided.
$\overline{\text{SCS}}$	AD22	I	Pullup	DIO chip select. When low, $\overline{\text{SCS}}$ indicates a DIO port access is valid. An internal pullup resistor is provided.
SDATA0 SDATA1 SDATA2 SDATA3 SDATA4 SDATA5 SDATA6 SDATA7	AF20 AE20 AD20 AC20 AF21 AE21 AD21 AC21	I/O	Pullup	DIO data bus. Byte-wide bidirectional DIO port. Internal pullup resistors are required.
$\overline{\text{SDMA}}$	AF24	I	Pullup	DIO DMA select. When low, $\overline{\text{SDMA}}$ modifies the behavior of the DIO interface to allow it to operate efficiently with an external direct memory access (DMA) controller. SAD0 and SAD1 are not used to select the internal host register for the access. Instead, the DIO address to access internal registers is provided by the DMAAddress register, and one of two host register addresses is selected according to dmainc in SysControl. An internal pullup resistor is provided.
SINT	AF19	O	None	Interrupt. Interrupt to the attached microprocessor. The interrupt type can be found in the Int register.
$\overline{\text{SRDY}}$	AF23	O	Pullup	DIO ready. When low during reads, $\overline{\text{SRDY}}$ indicates to the host when data is valid to be read. When low during writes, $\overline{\text{SRDY}}$ indicates when data has been received. $\overline{\text{SRDY}}$ is driven high for one clock cycle before placing the output in high impedance after $\overline{\text{SCS}}$ is taken high. An internal pullup resistor is provided.
SRNW	AC22	I	Pullup	DIO read not write – When high, read operation is selected. – When low, write operation is selected. An internal pullup resistor is provided.
SRXRDY	AE23	O	None	Network management (NM) port, receive ready. When high, SRXRDY indicates that the NM port's receive buffers are completely empty and the NM port is able to receive a frame of any size up to 1535 bytes in length.
STXRDY	AD23	O	None	Network management (NM) port, transmit ready. When high, STXRDY indicates that at least one buffer of frame data is available to be read by the management CPU. It outputs a 1 if any of the end-of-frame (eof), start-of-frame (sof), or interior-of-frame (iof) bits in NMTxControl is set to 1, otherwise, it outputs 0.

EEPROM interface

TERMINAL		I/O	INTERNAL RESISTOR	DESCRIPTION
NAME	NO.			
ECLK	L26	O	None	EEPROM data clock. An internal pullup resistor is provided.
EDIO	M26	I/O	Pullup	EEPROM data input/output. An internal pullup resistor is provided.

LED interface

TERMINAL		I/O	INTERNAL RESISTOR	DESCRIPTION
NAME	NO.			
LED_CLK	AD19	O	None	LED clock. Serial shift clock for the LED status data.
$\overline{\text{LED\_DATA}}$	AE19	O	None	LED data. Serial LED status data.

PRODUCT PREVIEW

Terminal Functions (Continued)

power supply

TERMINAL		DESCRIPTION
NAME	**NO.**	
GND	A1, A2, A13, A14, A25, A26, AF13, AF14, B1, B3, B24, B26, C2, C25, N1, N26, P1, P25, P26, R23, R24, R26, T24, U23, W23, W24, W25, W26, Y23, Y25, AA23, AA25, AB25, AD2, AD25, AE1, AE3, AE24, AE26, AF1, AF2, AF25, AF26	Ground. The 0-V reference for the TNETX4080.
GNDa	U24	Ground. The 0-V reference for the analog functions within the rambus ASIC cell (RAC).
$V_{DD(2.5)}$	B2, B25, C3, C24, D4, D9, D14, D18, D23, J4, J23, N4, P23, V4, V23, AC4, AC9, AC13, AC18, AC23, AD3, AD24, AE2, AE25	2.5-V supply voltage. Power for the core.
$V_{DD(3.3)}$	D8, D13, D17, H23, K4, P4, W4, AC10, AC14, AC19	3.3-V supply voltage. Power for the I/Os.
$V_{DD}a_{(2.5)}$	T23	2.5-V supply voltage. Power for the analog functions within the RAC.

DIO interface description

The DIO is a general-purpose interface that is used with a range of microprocessor or computer system interfaces. The interface is backward compatible with the existing TI ThunderSWITCH™ products. The DIO provides new signals to support external DMA controllers for improved performance.

This interface configures the switch using the attached CPU, and to access statistics registers (see Table 2). DIO accesses the NM port to allow frame data to be transferred between the CPU and the switch to support spanning tree, SNMP, and RMON. The CPU reads and writes packets directly under software control or an external DMA controller can be used to improve performance. See *TNETX4090 Programmer’s Reference Guide*, literature number SPAU003, for description of registers.

Table 2. DIO Internal Register Address Map

BYTE 3	BYTE 2	BYTE 1	BYTE 0	DIO ADDRESS
Port1Control		Port0Control		0x0000
Port3Control		Port2Control		0x0004
Port5Control		Port4Control		0x0008
Port7Control		Port6Control		0x000C
Reserved		Reserved		0x0010
Reserved				0x0014–0x003C
Reserved	UnkVLANPort	MirrorPort	UplinkPort	0x0040
Reserved		AgingThreshold		0x0044
Reserved				0x0048–0x004C
NLearnPorts				0x0050
TxBlockPorts				0x0054
RxUniBlockPorts				0x0058
RxMultiBlockPorts				0x005C
UnkUniPorts				0x0060
UnkMultiPorts				0x0064
UnkSrcPorts				0x0068
NewVLANIntPorts				0x006C
Reserved				0x0070–0x007C
TrunkMap3	TrunkMap2	TrunkMap1	TrunkMap0	0x0080
TrunkMap7	TrunkMap6	TrunkMap5	TrunkMap4	0x0084
Trunk3Ports	Trunk2Ports	Trunk1Ports	Trunk0Ports	0x0088
Reserved			Reserved	0x008C
Reserved				0x0090–0x009C
DevCode	Reserved	SIO	Revision	0x00A0
DevNode[23:16]	DevNode[31:24]	DevNode[39:32]	DevNode[47:40]	0x00A4
Reserved		DevNode[7:0]	DevNode[15:8]	0x00A8
MCastLimit				0x00DC
RamStatus	RamControl	Reserved		0x00E0
Reserved				0x00E4
PauseTime100		PauseTime10		0x00E8
Reserved		Reserved		0x00EC
Reserved	FlowThreshold			0x00F0
Reserved		LEDControl		0x00F4

PRODUCT PREVIEW

Table 2. DIO Internal Register Address Map (Continued)

BYTE 3	BYTE 2	BYTE 1	BYTE 0	DIO ADDRESS
SysControl		StatControl		0x00F8
Reserved (for EEPROM CRC)				0x00FC
VLAN0Ports				0x0100
VLAN1Ports				0x0104
VLAN2Ports				0x0108
VLAN3Ports				0x010C
VLAN4Ports				0x0110
VLAN5Ports				0x0114
VLAN6Ports				0x0118
VLAN7Ports				0x011C
VLAN8Ports				0x0120
VLAN9Ports				0x0124
VLAN10Ports				0x0128
VLAN11Ports				0x012C
VLAN12Ports				0x0130
VLAN13Ports				0x0134
VLAN14Ports				0x0138
VLAN15Ports				0x013C
Reserved				0x0140
Reserved				0x0144
Reserved				0x0148
Reserved				0x014C
Reserved				0x0150
Reserved				0x0154
Reserved				0x0158
Reserved				0x015C
Reserved				0x0160
Reserved				0x0164
Reserved				0x0168
Reserved				0x016C
Reserved				0x0170
Reserved				0x0174
Reserved				0x0178
Reserved				0x017C
Reserved				0x0180
Reserved				0x0184
Reserved				0x0188
Reserved				0x018C
Reserved				0x0190
Reserved				0x0194
Reserved				0x0198
Reserved				0x019C

Table 2. DIO Internal Register Address Map (Continued)

BYTE 3	BYTE 2	BYTE 1	BYTE 0	DIO ADDRESS
Reserved				0x01A0
Reserved				0x01A4
Reserved				0x01A8
Reserved				0x01AC
Reserved				0x01B0
Reserved				0x01B4
Reserved				0x01B8
Reserved				0x01BC
Reserved				0x01C0
Reserved				0x01C4
Reserved				0x01C8
Reserved				0x01CC
Reserved				0x01D0
Reserved				0x01D4
Reserved				0x01D8
Reserved				0x01DC
Reserved				0x01E0
Reserved				0x01E4
Reserved				0x01E8
Reserved				0x01EC
Reserved				0x01F0
Reserved				0x01F4
Reserved				0x01F8
Reserved				0x01FC
Reserved				0x0200–0x02FC
VLAN1QID		VLAN0QID		0x0300
VLAN3QID		VLAN2QID		0x0304
VLAN5QID		VLAN4QID		0x0308
VLAN7QID		VLAN6QID		0x030C
VLAN9QID		VLAN8QID		0x0310
VLAN11QID		VLAN10QID		0x0314
VLAN13QID		VLAN12QID		0x0318
VLAN15QID		VLAN14QID		0x031C
Reserved				0x0320
Reserved				0x0324
Reserved				0x0328
Reserved				0x032C
Reserved				0x0330
Reserved				0x0334
Reserved				0x0338
Reserved				0x033C
Reserved				0x0340
Reserved				0x0344

PRODUCT PREVIEW

Table 2. DIO Internal Register Address Map (Continued)

BYTE 3	BYTE 2	BYTE 1	BYTE 0	DIO ADDRESS
Reserved				0x0348
Reserved				0x034C
Reserved				0x0350
Reserved				0x0354
Reserved				0x0358
Reserved				0x035C
Reserved				0x0360
Reserved				0x0364
Reserved				0x0368
Reserved				0x036C
Reserved				0x0370
Reserved				0x0374
Reserved				0x0378
Reserved				0x037C
Port1QTag		Port0QTag		0x0380
Port3QTag		Port2QTag		0x0384
Port5QTag		Port4QTag		0x0388
Port7QTag		Port6QTag		0x038C
Reserved		Reserved		0x0390
Reserved				0x0394–0x03FC
Port1Status		Port0Status		0x0400
Port3Status		Port2Status		0x0404
Port5Status		Port4Status		0x0408
Port7Status		Port6Status		0x040C
Reserved		Reserved		0x0410
Reserved				0x0414–0x043C
FindNode[23:16]	FindNode[31:24]	FindNode[39:32]	FindNode[47:40]	0x0440
FindVLAN	FindControl	FindNode[7:0]	FindNode[15:8]	0x0444
FindPort				0x0448
NewNode[23:16]	NewNode[31:24]	NewNode[39:32]	NewNode[47:40]	0x044C
Reserved		NewNode[7:0]	NewNode[15:8]	0x0450
NewVLAN		NewPort		0x0454
AddNode[23:16]	AddNode[31:24]	AddNode[39:32]	AddNode[47:40]	0x0458
AddVLAN	AddDelControl	AddNode[7:0]	AddNode[15:8]	0x045C
AddPort				0x0460
AgedNode[23:16]	AgedNode[31:24]	AgedNode[39:32]	AgedNode[47:40]	0x0464
AgedVLAN	AgedPort	AgedNode[7:0]	AgedNode[15:8]	0x0468
DelNode[23:16]	DelNode[31:24]	DelNode[39:32]	DelNode[47:40]	0x046C
DelVLAN	DelPort	DelNode[7:0]	DelNode[15:8]	0x0470
AgingCounter		NumNodes		0x0474
Reserved				0x0478–0x0540
Reserved				0x0544
Reserved				0x0548

Table 2. DIO Internal Register Address Map (Continued)

BYTE 3	BYTE 2	BYTE 1	BYTE 0	DIO ADDRESS
Reserved				0x054C
Reserved				0x0550
Reserved				0x0554
Reserved				0x0558
Reserved				0x055C
Reserved				0x0560
Reserved				0x0564
Reserved				0x0568
Reserved				0x056C
Reserved				0x0570
Reserved				0x0574
Reserved				0x0578
Reserved				0x057C
Reserved				0x0580
Reserved				0x0584
Reserved				0x0588
Reserved				0x058C
Reserved				0x0590
Reserved				0x0594
Reserved				0x0598
Reserved				0x059C
Reserved				0x05A0
Reserved				0x05A4
Reserved				0x05A8
Reserved				0x05AC
Reserved				0x05B0
Reserved				0x05B4
Reserved				0x05B8
Reserved				0x05BC
Reserved				0x05C0
Reserved				0x05C4
Reserved				0x05C8
Reserved				0x05CC
Reserved				0x05D0
Reserved				0x05D4
Reserved				0x05D8
Reserved				0x05DC
Reserved				0x05E0
Reserved				0x05E4
Reserved				0x05E8
Reserved				0x05EC
Reserved				0x05F0

Table 2. DIO Internal Register Address Map (Continued)

BYTE 3	BYTE 2	BYTE 1	BYTE 0	DIO ADDRESS
Reserved				0x05F4
Reserved				0x05F8
Reserved				0x05FC
Reserved				0x0600–0x060C
Reserved				0x0700
Reserved				0x0704
Reserved				0x0708
Reserved				0x070C
Reserved				0x0710
Reserved				0x0714–0x0718
Reserved				0x071C
Reserved				0x0720–0x07FC
Reserved		DMAAddress		0x0800
Reserved	Int			0x0804
Reserved	IntEnable			0x0808
SysTest	FreeStackLength			0x080C
RAMAddress				0x0810
Reserved			RAMData	0x0814
Reserved	NMRxControl			0x0818
Reserved	NMTxControl			0x081C
Reserved			NMData	0x0820
Reserved				0x0824–0x0FFC
Manufacturing test registers (internal use only)				0x1000–0x10FF
Reserved				0x1900–0x3FFC
Hardware reset				0x4000–0x5FFC

DIO interface description (continued)

Table 3 and Table 4 list the least significant byte address for the port-specific statistics. Table 5 lists address-lookup statistics. Each statistic is four bytes long. To determine the address of a particular statistic, replace the xx in the head column with the characters from the tail address. Table 3 has two tail columns: one for even-numbered ports and the other for odd-numbered ports. See the *TNETX4090 Programmer's Reference Guide,* literature number SPAU003, for a detailed description of the statistic registers.

Example:

Port 7 head	= 0x83xx
64-octet frames tail	= A8 (odd-numbered port)
Port 7 octet frames statistic	= 0x83A8

Table 3. Port Statistics 1

PORT NO.	HEAD	STATISTIC	TAIL	
			EVEN PORTS	ODD PORTS
0	0x80xx	Receive octet	00	80
1	0x80xx	Good receive frames	04	84
2	0x81xx	Broadcast receive frames	08	88
3	0x81xx	Multicast receive frames	0C	8C
4	0x82xx	Receive CRC errors	10	90
5	0x82xx	Receive align/code errors†	14	84
6	0x83xx	Oversized receive frames	18	98
7	0x83xx	Receive jabbers	1C	9C
Reserved	0x84xx	Undersized receive frames	20	A0
NM	0x84xx	Receive fragments	24	A4
Reserved	0x85xx	64-octet frames	28	A8
	0x85xx	65–127 octet frames	2C	AC
	0x86xx	128–255 octet frames	30	B0
	0x86xx	256–511 octet frames	34	B4
	0x87xx	512–1023 octet frames	38	B8
	0x87xx	1024–1518 octet frames	3C	BC
	0x88xx	Net octets	40	C0
	0x88xx	SQE test errors†	44	C4
	0x89xx	Tx octets	48	C8
	0x89xx	Good transmit frames	4C	CC
	0x8Axx	Single-collision transmit frames†	50	D0
	0x8Axx	Multiple-collision transmit frames†	54	D4
	0x8Bxx	Carrier sense errors†	58	D8
	0x8Bxx	Deferred transmit frames†	5C	DC
	0x8Cxx	Late collisions†	60	E0
	0x8Cxx	Excessive collisions†	64	E4
	0x8Dxx	Broadcast transmit frames	68	E8
	0x8Dxx	Filtered receive frames	6C	EC
	0x8Exx	Filtered receive frames	70	F0
	0x8Exx	Transmit data errors	74	F4
	0x8Fxx	Collisions†	78	F8
	0x8Fxx	Receive overruns	7C	FC

† The NM port does not have this statistic. This address is reserved on the NM port.

Table 4. Port Statistics 2

PORT NO.	HEAD	STATISTIC	TAIL (ALL PORTS)
0	0x900x	Pause transmit frames†	0
1	0x901x	Pause receive frames†	4
2	0x902x	Security violations	8
3	0x903x	Reserved	C
4	0x904x		
5	0x905x		
6	0x906x		
7	0x907x		
Reserved	0x908x		
NM	0x909x		
Reserved	0x90Ax		
	0x90Bx		
	0x90Cx		
	0x90Dx		
	0x90Ex		
	0x90Fx		
	0x910x		
	0x911x		
	0x912x		
	0x913x		
	0x914x		
	0x915x		
	0x916x		
	0x917x		
	0x918x		
	0x919x		
	0x91Ax		
	0x91Bx		
	0x91Cx		
	0x91Dx		
	0x91Ex		
	0x91Fx		

† The NM port does not have this statistic. This address is reserved on the NM port.

PRODUCT PREVIEW

DIO interface description (continued)

Table 5. Address-Lookup Statistics

PORT NO.	HEAD	STATISTIC
N/A	0x9200–0x9FFC	Reserved
N/A	0xA000	Unknown unicast destination addresses
N/A	0xA004	Unknown multicast destination addresses
N/A	0xA008	Unknown source addresses
N/A	0xA00C–0xFFFC	Reserved

When accessing the statistics values from the DIO port, it is necessary to perform four 1-byte DIO reads to obtain the full 32-bit counter. Counters always should be read in ascending byte-address order (0, 1, 2, 3). To prevent the counter being updated while reading the four bytes, the entire 32-bit counter value is transferred to a holding register when byte 0 is read.

To provide ease of use with both big- and little-endian CPUs, two alternative byte-ordering schemes are supported. The mode of operations can be selected through the StatControl register.

receiving/transmitting management frames

Frames originating within the host are written to the NM port via the NMRxControl and NMData registers. Once a frame has been fully written, it is then received by the switch and routed to the destination port(s).

Frames that were routed to this port from any of the switch ports are placed in a queue until the host is ready to read them via the NMTxControl and NMData registers. They then are effectively transmitted out of the switch.

$\overline{\text{SDMA}}$ can be used to transmit or r$\overline{\text{eceive}}$ management frames (the SAD1–SAD0 pins are ignored when $\overline{\text{SDMA}}$ is asserted) (see Table 6). When $\overline{\text{SDMA}}$ is asserted, the switch uses the value in the DMAAddress register instead of the DIO address registers to access frame data (this also can be used to access the switch statistics). STXRDY and SRXRDY, the interrupts, freebuffs, eof, sof, and iof mechanisms can be used, as desired, to prevent unwanted stalls on the DIO bus during busy periods.

Table 6. DMA Interface Signals

SIGNAL	DESCRIPTION
$\overline{\text{SDMA}}$	Automatically sets up DIO address using the DMAAddress register
STXRDY	Indicates that at least one data frame buffer can be read by the management CPU
SRXRDY	Indicates that the management CPU can write a frame of any size up to 1535 bytes

state of DIO signal pins during hardware reset

The CPU can perform a hardware reset by writing to an address in the range of 0x40–0x5F (writes to a DMA address in this range have no effect on reset); this is equivalent to asserting the hardware $\overline{\text{RESET}}$ pin with the following exceptions. During hardware reset, the output and bidirectional DIO pins behave as shown in Table 7.

- DIO interface continues to operate. The reset condition remains active until $\overline{\text{SCS}}$ is driven high. $\overline{\text{SRDY}}$ does not become high impedance or resistively pulled high (unlike a true hardware reset), so it still can be used as a normal acknowledge in this case.
- Following the reset, no EEPROM autoload is performed.

Table 7. DIO Interface During Hardware Reset

DIO INTERFACE	STATE DURING HARDWARE RESET
$\overline{\text{SRDY}}$	High impedance – resistively pulled up
SDATA7–SDATA0	High impedance – resistively pulled up
STXRDY	Driven low
SRXRDY	Driven high

IEEE Std 802.1Q VLAN tags on the NM port

Frames received from the host via the NM port are required to contain a valid IEEE Std 802.1Q header (frames that do not contain a valid IEEE Std 802.1Q header are incorrectly routed). They also can be corrupted at the transmission port(s) as the tag-stripping process does not check that the four bytes after the source address actually are a valid tag. The four bytes are a valid tag under all other circumstances.

When a frame is transmitted by the NM port (received by the host), no tag-stripping occurs, so the frame may contain one or two tags, depending on how the frame originally was received.

frame format on the NM port

The frame format on the NM port differs slightly from a standard Ethernet frame format. The key differences are: the frame always contains an IEEE Std 802.1Q header in the four bytes following the source address (see Figure 2). The tag protocol identifier (TPID) or ethertype field, however, is used in the switch for other purposes, so a frame transmitted out of the switch on the NM port does not have the IEEE Std 802.1Q TPID of 81–00 (ethertype constant) value in these two bytes.

The first TPID byte output contains:

- The frame source port number in the least significant bits. This allows the frame source port number to be carried within the frame, which is useful for processing BPDUs, for example.
- A cyclic redundancy check (CRC) type indicator (crctype) in the most significant bit (bit 7).
  - If crctype = 1, then the CRC word in the frame excludes the IEEE Std 802.1Q header.
  - If crctype = 0, then the CRC word in the frame includes the IEEE Std 802.1Q header. This CRC word is for a regular IEEE Std 802.1Q frame format with the value in the IEEE Std 802.1Q TPID of 81–00 (ethertype constant) in the TPID field. Because the internal frame format uses the TPID field for other purposes in the manner being described, it is necessary to insert the IEEE Std 802.1Q TPID of 81–00 (ethertype constant) value into the TPID field if the frame needs to be restored to a normal IEEE Std 802.1Q frame format, which passes a CRC check.

frame format on the NM port (continued)

To provide a CRC word, which includes the header, the NM port generates a new CRC word as the frame is being read out. It simultaneously checks the existing CRC in the frame and, if an error is found, ensures that the final byte of the newly generated CRC is corrupted to contain an error, too. The CRC word is deliberately corrupted if the header parity protection (described in the following) indicates an error in the header. In either case, the pfe bit also is set to 1 after the final byte of the frame has been read from NMData.

If the frame was received on a port other than the NM port, then the crctype bit is set, depending on whether or not an IEEE Std 802.1Q tag header was inserted into the frame during ingress.

– If crctype = 1, a header was inserted.

– If crctype = 0, a header was not inserted (crctype also is 0 if the frame VLAN ID was 0x000 and was replaced by the port VLANID (PVID) from the PortxQTag register).

In an IEEE Std 802.1D-compliant application, the header simply can be removed from the frame to produce a headerless frame with a correct CRC word.

– All other bits in the byte are reserved and are 0.

The second TPID byte output contains:

- Odd-parity protection bits for the other three bytes in the tag header
- Bit 5 protects the first byte of the TPID field (i.e., the one containing crctype and source port number).
- Bit 6 protects the first byte of the VLAN ID field.
- Bit 7 protects the second byte of the VLAN ID field.
- All other bits in the byte are reserved and are 0.

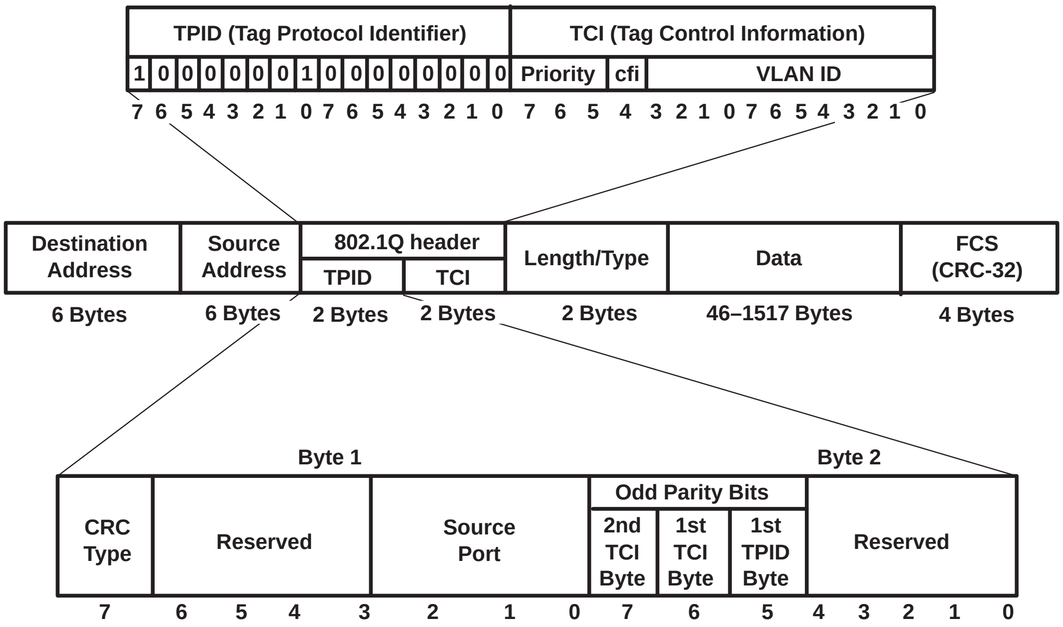


Figure 2. NM Frame Format

frame format on the NM port (continued)

Any device reading frames out of the NM port must expect frames to be in the format shown in Figure 2.

Frames received into the switch on the NM port also must conform to this format, with the following caveats:

- crc = 0 in NMRxControl

  When the host is providing a frame containing valid CRC it also must provide in the TPID field valid header parity protection and indicate via the crctype bit which type of CRC the frame contains (i.e., including the header (crctype = 0), or excluding the header (crctype = 1)). If crctype indicates that the header is included, as for NM port transmissions, this mimics the presence of IEEE Std 802.1Q TPID of 81–00 (ethertype constant) in the TPID field. If a CRC error or parity error is detected, the frame is discarded.

  When crctype indicates that the header is included, the NM port regenerates CRC to exclude the header during the reception process (this converts the frame into the required internal frame format).

- crc = 1 in NMRxControl

  If the switch is being asked to generate a CRC word for the frame, the values in the TPID field are ignored by the NM port. The switch inserts header parity protection. It replaces the final four bytes of the frame with the calculated CRC (the values in the final four bytes provided are don't care).

  In either case, the NM port inserts its own port number into the source port field in the least significant bits of the first TPID byte, sets the crctype bit to 0, and also sets the reserved bits to 0.

Frames received from the host via the NM port are required to contain a valid IEEE Std 802.1Q VLAN ID in the third and fourth bytes, following the source address (the NM port does not have a PortxQTag register for inserting a VLAN tag if none is provided and does not have an rxacc bit). Frames that do not contain a VLAN tag are incorrectly routed. They also can be corrupted at the transmission port(s). The header-stripping process does not check that the two bytes after the source address are a valid IEEE Std 802.1Q TPID because there is a valid header under all other circumstances.

When a frame is transmitted on the NM port, no header stripping occurs (again, because the NM port does not have a PortxQTag register or txacc bit), so the frame read by the host software contains one header (or possibly more, depending on how the frame was received).

In either case, the NM port inserts its own port number into the source port field in the least significant bits of the first TPID byte and sets the reserved bits to 0. Frames received from the host via the NM port are required to contain a valid IEEE Std 802.1Q VLAN ID (VID) in the third and fourth bytes following the source address. (The NM port does not have a default VLAN ID register for inserting a VLAN tag if none is provided. It cannot also be configured as an access port.) Frames that do not contain a valid tag are incorrectly routed. They also can be corrupted at the transmission port(s) as the tag-stripping process does not check that the four bytes after the source address are a valid tag because they are valid tags under all other circumstances.

When a frame is transmitted on (read from) the NM port, no tag stripping occurs (because the NM port does not have the default VLAN ID register or access configuration control), so the frame read by the host software can contain one or more header tags, depending on how the frame was received.

PRODUCT PREVIEW

full-duplex NM port

The NM port can intermix reception and transmission as desired. It is the direction of the NMData access (i.e., read or write) that determines whether a byte is removed from the transmit queue or added to the receive queue. The DIO interface, however, is only half duplex since it cannot do a read and write at the same time.

NM bandwidth and priority

The NM port is capable of transferring a byte to or from NMData once every 80 ns, or 100 Mbit/s. This can be sustained between the DIO port and the NM ports dedicated receive or receive buffers.

However, the NM port is prioritized lower than the other ports between its receive buffers and the external memory system so that, during periods of high activity, the NM port does not cause frames to be dropped on the other ports.

interrupt processing

The SRXRDY signal and the nmrx interrupt are set when the receive FIFO is completely empty. This indicates that the NM port is ready to accept a frame of any length (up to 1535 bytes).

If the host wished to download a sequence of frames, it could use the freebuffs field to determine space availability.

PHY management interface

This interface gives the user an easy way to implement a software-controlled bit serial MII PHY management interface.

MII devices that implement the management interface consisting of MDIO and MDCLK can be accessed through an internal register (see the *TNETX4090 Programmer's Reference Guide*, literature number SPAU003, for details on controlling this interface). A third signal, $\overline{\text{MRESET}}$, is provided to allow hardware reset of PHYs that support it.

All three pins have internal pullup resistors since they can be placed in a high-impedance state to allow another bus master.

The interface does not implement any timing or MII frame formatting. The timing and frame format must be ensured by the management software setting or clearing the bits within the internal registers in an appropriate manner. Refer to IEEE Std 802.3u and the MII device data sheets for the appropriate protocol requirements.

MAC interface

receive control

Data received from the PHYs is interpreted and assembled into the TNETX4080 buffer memory. Interpretation involves detection and removal of the preamble, extraction of the address and frame length, extraction of the IEEE Std 802.1Q header (if present), and data handling and CRC. Also included is a jabber-detection timer to detect frames that exceed the maximum length being received on the network.

giant (long) frames

The maxlen bit within each port's PortxControl register controls the maximum received frame size on that port.

- If maxlen = 0, the maximum received frame length Is 1535 bytes if no VLAN header is inserted, or 1531 bytes if a VLAN header is inserted. (When stored within the switch, a frame never can be longer than 1535 bytes).
- If maxlen = 1, the maximum received frame length is 1518 bytes as specified by IEEE Std 802.3. This is the maximum length on the wire. If a VLAN header is inserted into a 1518-byte frame within the MAC, the frame is stored as a 1522-byte frame within the switch.

All received frames that exceed the maximum length are discarded by the switch.

The long option bit in StatControl indicates how the statistics for long frames should be recorded.

short frames

All received frames less than 64 bytes in length are discarded upon reception and are not stored in memory or transmitted.

receive filtering of frames

Received frames that contain an error (e.g., CRC, alignment, jabber, etc.) are discarded before transmission and the relevant statistics counter is updated.

data transmission

The MAC takes data from the TNETX4080 internal buffer memory and passes it to the PHY. The data also is synchronized to the transmit clock rate.

A CRC block checks that the outgoing frame has not been corrupted within the switch by verifying that it still has a valid CRC as the frame is being transmitted. If a CRC error is detected, it is counted in the transmit data errors counter.

transmit control

The frame control block handles the output of data to the PHYs. Several error states are handled. If a collision is detected, the state machine jams the output. If the collision was late (after the first 64-byte buffer has been transmitted), the frame is lost. If it is an early collision, the controller backs off before retrying. While operating in full duplex, both carrier-sense (CRS) mode and collision-sensing modes are disabled (the switch does not start transmitting a new frame if collision is active in full-duplex mode).

Internally, frame data only is removed from buffer memory once it has been successfully transmitted without collision (for the half-duplex ports). Transmission recovery also is handled in this state machine. If a collision is detected, frame recovery and retransmission are initiated.

PRODUCT PREVIEW

adaptive performance optimization (APO)

Each Ethernet MAC incorporates APO logic. This can be enabled on an individual port basis. When enabled, the MAC uses transmission pacing to enhance performance (when connected on networks using other transmit pacing-capable MACs). Adaptive performance pacing introduces delays into the normal transmission of frames, delaying transmission attempts between stations, and reducing the probability of collisions occurring during heavy traffic (as indicated by frame deferrals and collisions), thereby, increasing the chance of successful transmission.

When a frame is deferred, suffers a single collision, multiple collisions, or excessive collisions, the pacing counter is loaded with an initial value of 31. When a frame is transmitted successfully (without a deferral, single collision, multiple collision, or excessive collision), the pacing counter is decremented by 1, down to 0.

With pacing enabled, a new frame is permitted to immediately [after one inter-packet gap (IPG)] attempt transmission only if the pacing counter is 0. If the pacing counter is not 0, the frame is delayed by the pacing delay (a delay of approximately four interframe gap delays).

NOTE:
APO affects only the IPG preceding the first attempt at transmitting a frame. It does not affect the backoff algorithm for retransmitted frames.

interframe gap enforcement

The measurement reference for the interpacket gap of 96-bit times is changed, depending on frame traffic conditions. If a frame is successfully transmitted (without collision), 96-bit times is measured from Mxx_TXEN. If the frame suffered a collision, 96-bit times is measured from Mxx_CRS.

backoff

The device implements the IEEE Std 802.3 binary exponential backoff algorithm.

receive versus transmit priority

The queue manager prioritizes receive and transmit traffic as follows:

- Highest priority is given to frames that currently are being transmitted. This ensures that transmitting frames do not underrun.
- Next priority is given to frames that are received if the free-buffer stack is not empty. This ensures that received frames are not dropped unless it is impossible to receive them.
- Lowest priority is given to frames that are queued for transmission but have not yet started to transmit. These frames are promoted to the highest priority only when there is spare capacity on the memory bus.
- The NM port receives the lowest priority to prevent frame loss during busy periods.

The memory bus has enough bandwidth to support the two highest priorities. The untransmitted frame queues grow when frames received on different ports require transmission on the same port(s) and when frames are repeatedly received on ports that are at a higher speed than the ports on which they are transmitted. This is likely to be exacerbated by the reception of multicast frames, which typically require transmission on several ports. When the backlog grows to such an extent that the free buffer stack is nearly empty, flow control is initiated (if it has been enabled) to limit further frame reception.

10-/100-Mbit/s MII (ports 0–7)

speed, duplex, and flow-control negotiation

Each individual port can operate at 10 Mbit/s or 100 Mbit/s in half or full duplex, and can indicate (or not) support of IEEE Std 802.3 flow control. The operating modes for each port can be negotiated between the MACs and the PHYs after power up, by setting neg in PortxControl. This provides for unmanaged operation, when using PHYs that support this signaling scheme.

If neg = 1, negotiation is initiated via the Mxx_LINK signal being driven low by the PHY. As long as Mxx_LINK is low, the MAC indicates the capabilities it wishes the PHY to negotiate with. It outputs on:

- Mxx_TXD0 is the desired duplex (0 = half, 1 = full). This signal reflects reqhd in the appropriate PortxControl register.
- Mxx_TXD1 is the desired IEEE Std 802.3 flow-control mode (0 = no pause, 1 = pause required). This signal reflects the inverse of the value of reqnp in the appropriate PortxControl register.
- Mxx_TXD2 is the desired speed (0 = 10 Mbit/s, 1 = 100 Mbit/s required). This signal reflects the inverse of the value of req10 in the appropriate PortxControl register.

Mxx_TD3 does not take part in the negotiation process and outputs as 0 while Mxx_LINK is low.

As long as Mxx_LINK is low, the PHY outputs on:

- Mxx_RXD0 is the result of duplex negotiation (0 = half, 1 = full) that is recorded in the duplex bit of the appropriate PortxStatus register.
- Mxx_RXD1 is the result of flow-control negotiation (0 = no pause, 1 = pause supported) that is recorded in the pause bit of the appropriate PortxStatus register.
- Mxx_RXD2 is the result of speed negotiation (0 = 10 Mbit/s, 1 = 100 Mbit/s supported) that is recorded in the speed bit of the appropriate PortxStatus register.

Mxx_RXD3 is ignored by the switch when link is low.

If the switch is autobooted via an EEPROM, this negotiation is automatic (if the neg bit of the appropriate PortxControl register is set to 1 by the EEPROM load). The switch is active and outputs valid requests on Mxx_TXD0, Mxx_TXD1, and Mxx_TXD2 before Mxx_LINK is taken high by the PHY (see Figure 3).

If, however, a switch requires software initiation, or at a later time, software desires a change in the mode of a port, it must request the PHY to drive Mxx_LINK low to begin renegotiation. This is achieved by writing to the control registers within the PHY via the serial MII interface.

PRODUCT PREVIEW

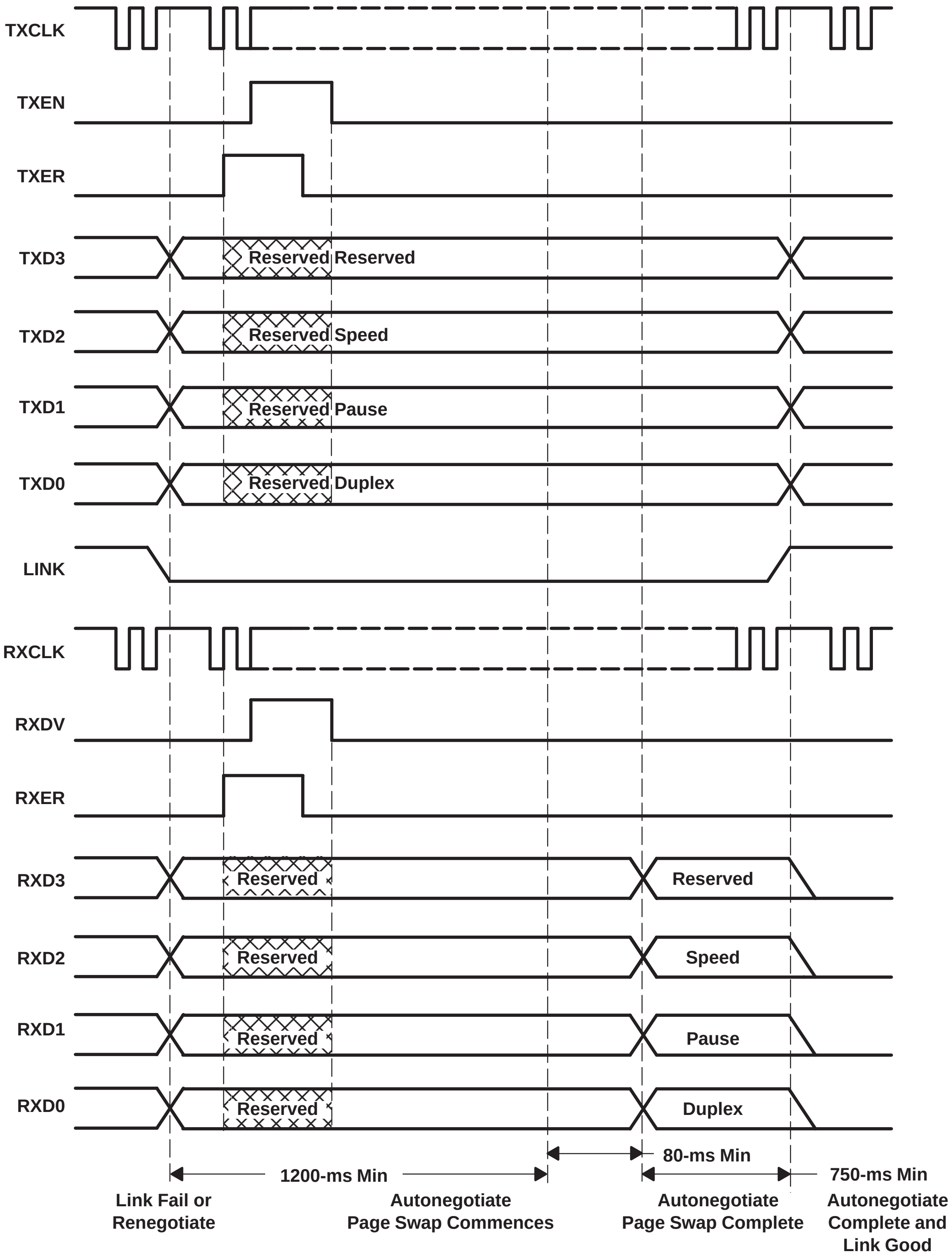


Figure 3. 10-/100-Mbit/s Port Negotiation With the TNETE2104

EEPROM interface

The EEPROM interface is provided so the system-level manufacturer can produce a CPU-less preconfigured system. This also can be used to change or reconfigure the system and retain the preferences between system power downs.

The EEPROM contains configuration and initialization information that is accessed infrequently, typically at power up and after a reset. The organization of the EEPROM data is shown in the DIO address map. Downloads are initiated in one of two ways:

- At the end of hard reset (rising edge on $\overline{\text{RESET}}$)
- Writing a 1 to load in SysControl register. This bit is cleared automatically when the download completes. It cannot be set during the download by the EEPROM data, thereby preventing a download loop.

During the download, no DIO writes are permitted. (If a DIO write is attempted, $\overline{\text{SRDY}}$ is held high until the download has completed.)

Either a 24C02 or 24C08 serial EEPROM device can be used. Both use a two-wire serial interface for communication and are available in a small-footprint package.

- The 24C02 provides 2048 bits, organized as 256 × 8. Downloading data from the EEPROM initializes DIO addresses 0x0000 through 0x00FB. These registers control all initializable functions except VLANs. The downloading sequence starts with DIO address 0x0000, continuing in ascending order to 0x00FF.
- The 24C08 provides 8192 bits, organized as 1024 × 8. Downloading data from the EEPROM initializes DIO addresses 0x0000 through 0x03FF. These registers control all initializable functions, including VLANs. The downloading sequence starts with DIO address 0x0100, continuing in ascending order to 0x03FF, followed by address 0x0000, and continuing in ascending order to 0x00FF. This ensures that SysControl is the last register loaded.

The EEPROM size is detected automatically according to the address assigned to the EEPROM:

- 2048 bits, organized as a 256 × 8 EEPROM, should have A0, A1, and A2 pins tied low.
- 8192 bits, organized as a 1024 × 8 EEPROM, should have A0 and A1 pins tied low and A2 pin tied high (see Figure 4).

PRODUCT PREVIEW

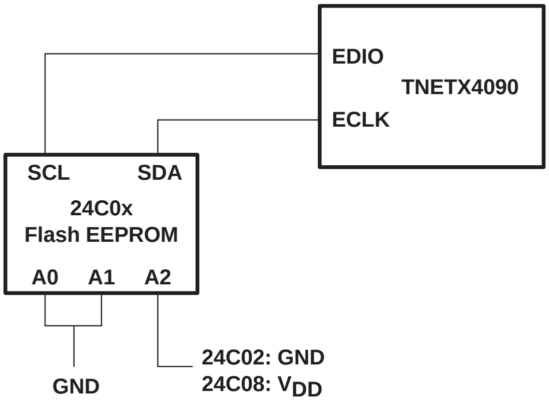


Figure 4. Flash EEPROM Configuration

EEPROM interface (continued)

After the initial start condition, a slave address containing a device address of 000 is output on EDIO, and then EDIO is observed for an acknowledge from the EEPROM. If an acknowledge is received, operation continues for the 24C02 EEPROM. If none is received, a stop condition is generated, followed by another start condition and slave address, this time containing a device address of 100. If this receives no acknowledge, no EEPROM is present, and device operation continues, using the current register settings (i.e., those following a hardware reset, or those previously entered by software).

When this device is driving EDIO, it drives out only a strong logical 0. When a logical 1 is intended to be driven out, the pin must be resistively pulled high. An on-chip 50-µA current-source pullup device is provided on this pin. The system designer must decide if this is sufficient to achieve a logical-1 level in a timely manner or if an external supplementary resistor is required.

Multiple bus masters are not supported on the EEPROM interface because the ECLK pin always is driven by the device with a strong 0/strong 1 (i.e., not a strong 1/resistively pulled-up 1).

An Ethernet CRC check is used to ensure the EEPROM data is valid. The 4-byte CRC should be placed within the EEPROM in four data bytes immediately following the last byte to be loaded (equivalent to locations 0x00FC–0x00FF, just above SysControl). As each byte is loaded from the EEPROM, the bits within that byte are entered into the CRC checker bit-wise, most significant bit first.

A valid CRC always must be provided by the EEPROM. The EEPROM data for the most significant bit of SysControl is withheld until the CRC computed by the device has been checked against the one read from the EEPROM. If the CRC is invalid:

- The reset bit is set to 1 in SysControl, load and initd are both 0, and the TNETX4080 does not begin operation.
- The fault LED is illuminated and remains in that state until the TNETX4080 is hardware reset or until load in SysControl is set to 1.

interaction of EEPROM load with the SIO register

The EDIO pin is shared with the SIO register edata bit. The edata and etxen bits must not both be set to 1 when the load bit is set or the EDIO pin is held at resistive 1 and the EEPROM load fails.

The value of the eclk bit in SIO is don’t care when load is set, but to ensure the EEPROM does not see a glitch on its clock signal, the load bit should not be set until the minimum clock high or low time required by the EEPROM on its clock signal has expired since the eclk bit was last changed.

The SIO register is not loaded during the EEPROM download.

summary of EEPROM load outcomes

Table 8 summarizes the various states of register bits and the fault LED for each possible outcome following an EEPROM load attempt.

Table 8. Summary of EEPROM Load Outcomes

OUTCOME	STOP	LOAD	INITD†	FAULT LED	ECLK
Successful load	0	0	1	0‡	Not locked
No EEPROM present	0	0	0	0‡	Locked
CRC error detected	1	0	0	1	Not locked

† Assuming the start bit was set to 1 by the EEPROM load
‡ Assuming the fault bit in LEDControl = 0 and no memory system parity error is detected

compatibility with future device revisions

All EEPROM locations that correspond to reserved addresses in the memory map, register bits that are read only, and register bits that are marked as reserved should be set to 0 to ensure compatibility with future versions of the device. Failure to do so may result in the unintentional activation of features in future devices. All such bits are included in the CRC calculation.

LED interface

This interface allows a visual status for each port to be displayed. In addition, the state of the internal flow control and fault functions are displayed along with 12 software-controllable LEDs.

Each port has a single LED, which can convey three states, as shown in Table 9.

Table 9. Port LED States

STATE	DISPLAY
No link	Off
Link, but no activity	On
Activity	Flashing at 8 Hz

The interface is intended for use in conjunction with external octal shift registers, clocked with LED_CLK. Every 1/16th of a second, the status bits are shifted out via $\overline{\text{LED\_DATA}}$.

The status bits are shifted out in one of two possible orders, as determined by slast in LEDControl, to ensure that systems that do not require all the LED status can be implemented with the minimum number of octal shift registers (see Table 10).

- If slast = 0, the software-controlled status bits are shifted out before the port status bits.
- If slast = 1, the software-controlled status bits are shifted out after the port status bits.

The fault status bit is shifted out last, enabling a minimal system that displays only the fault status to be implemented without any shift registers.

Table 10. LED Status Bit Definitions and Shift Order

ORDER		NAME	FUNCTION
slast = 0	slast = 1		
1st–12th	11th–22nd	SW0–SW11	Software LEDs 0–11. These allow additional software-controlled status to be displayed. These 12 LEDs reflect the values of bits 0–11 of the swied field in LEDControl at the moment that the LED interface samples them. If this occurs between writes to the most significant and least significant bytes of LEDControl, these values appear on the LEDs, separated by 1/16th of a second.
13th–21st	1st–9th	P00–P08	Port status LEDs 0–8. These nine LEDs indicate the status of ports 0–8, in this order (port 0 is output first). Note that port 9 (management port) does not have an LED. The transmit multicast content of these bits can be controlled by the txais bit in LEDControl. Note that IEEE Std 802.3x pause frames never appear on the LEDs as port activity. The port's LED toggles each 1/16th of a second if there was any frame traffic (other than pause frames) on the port during the previous 1/16th of a second.
22nd	10th	C08	Port 8 collision LED. LED is extinguished if the port 8 is not in PMA mode. It indicates the collisions on port 8 and toggles each 1/16th of a second if there is a collision on the port during the previous 1/16th of a second.
23rd	23rd	FLOW	Flow control. LED is on when the internal flow control is enabled and active. Active means that flow control was asserted during the previous 1/16th of a second.
24th	24th	FAULT	Fault. LED indicates: – the EEPROM CRC was invalid. – an external DRAM parity error has occurred. – the FITLED in LEDControl has been set. The CRC and parity error indications are cleared by hardware reset (pin or DIO). The CRC error indication also is cleared by setting load to 1. The parity error indication also is cleared by setting start to 1.

lamp test

When the device is in the hardware reset state, $\overline{\text{LED\_DATA}}$ is driven low and LED_CLK runs continuously. This causes all LEDs to be illuminated and serves as a lamp test function.

multi-LED display

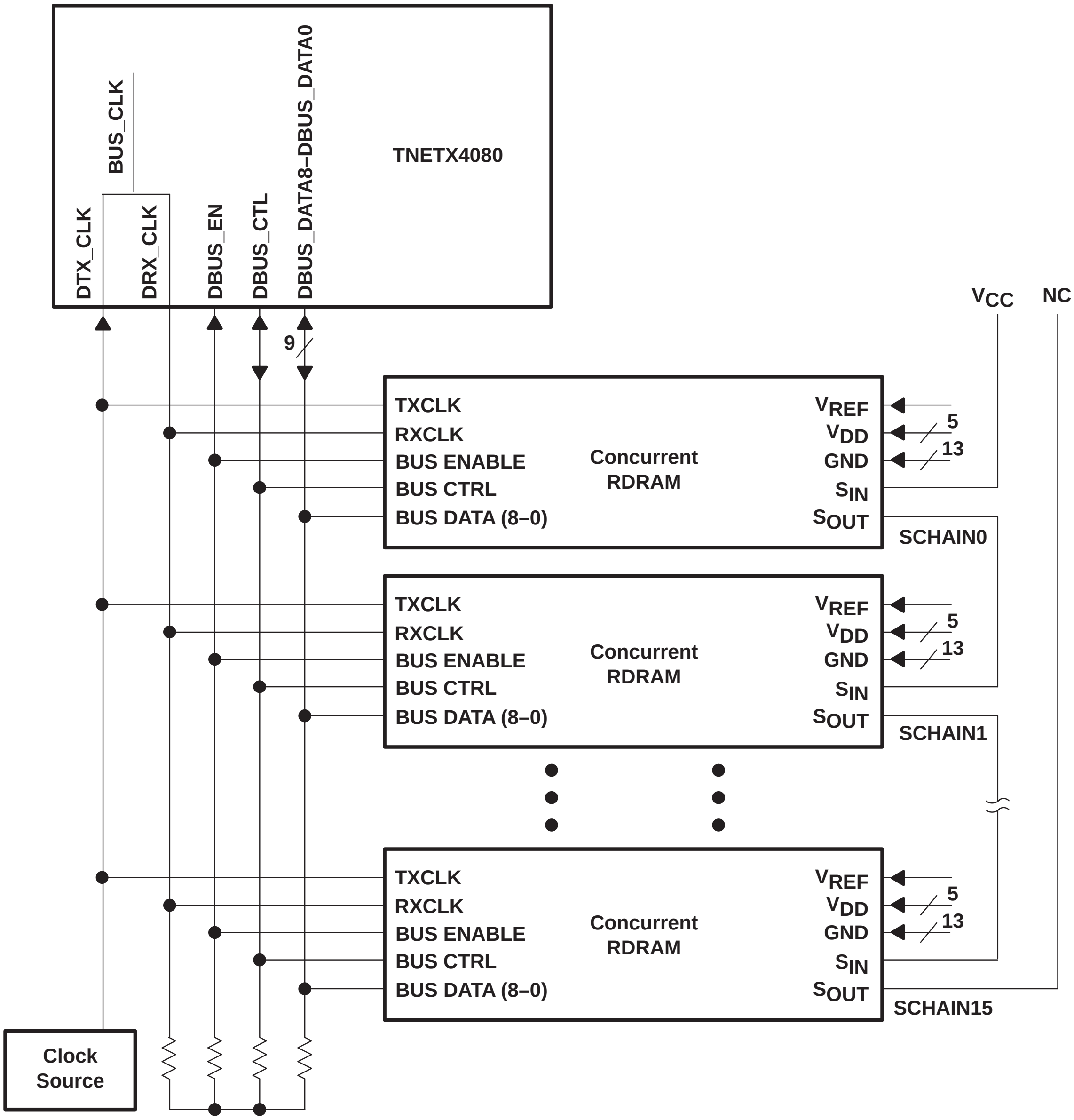
The LED interface is intended to provide the lowest-cost display with a single multifunction LED per port. In systems requiring a full-feature display using multiple LEDs per port, this is achieved by driving the LEDs directly from the PHY signals.

RDRAM interface

The TNETX4080 requires the use of external memory devices to retain frame data during switching operations. The high bandwidth requirements of gigabit-per-second Ethernet switching are met using a concurrent RDRAM interface (see *Rambus Layout Guide,* literature number DL0033).

Each RDRAM interface operates at 600-Mbit/pin/s and is intended for use with 16-/18-/64-/72-Mbit/s concurrent RDRAMs with access times of 50 ns. The TNETX4080 automatically determines the word length of the RDRAMs during initialization and performs parity checks if 9-bit memories are in use.

A maximum of 16 RDRAM devices of differing organizations can be attached to any one RDRAM interface. Multiple devices must be daisy-chained together via their SIN and SOUT pins during initialization (see Figure 5). All RDRAMs in a given system must be of the same type.



NC – No internal connection

Figure 5. Multiple RDRAM Module Connections

PRODUCT PREVIEW

IEEE Std 1149.1 (JTAG) interface

The TNETX4080 is fully IEEE Std 1149.1 compliant. It also includes on-chip pullup resistors on the five IEEE Std 1149.1 (JTAG) pins to eliminate the need for external ones. The instructions that TI supports are:

- Mandatory (EXTEST, BYPASS, and SAMPLE/PRELOAD)
- Optional public (HIGHZ, IDCODE, and BIST)
- Private (various private instructions are used by TI for test purposes)

The opcodes for the various instructions (6-bit instruction register) are shown in Table 11.

Table 11. IEEE Std 1149.1 (JTAG) Instruction Opcodes

INSTRUCTION TYPE	INSTRUCTION NAME	IEEE Std 1149.1 (JTAG) OPCODE
Mandatory	EXTEST	000000
Mandatory	SAMPLE/PRELOAD	000001
Optional	IDCODE	000100
Optional	HIGHZ	000101
Optional	RACBIST	000110
Private	TI testing	Others
Mandatory	BYPASS	111111

HIGHZ instruction

When selected, the HIGHZ instruction causes all outputs and bidirectional pins to become high impedance. All pullup and pulldown resistors are disabled.

RACBIST instruction

The RACBIST instruction invokes a built-in self test of the RAC and the rambus channel. This tests the integrity of the connection between the TNETX4080 and the external RDRAMs. When selected, the value of the test can be read via JTAG DR SCAN. A 2-bit status value is reported (see Table 12).

Table 12. IEEE Std 1149.1 (JTAG) BIST Status

PASS (BIT 1)	COMPLETE (BIT 0)
When bit 0 = 1 *0 = fail *1 = pass	*0 = BIST running
	*1 = BIST complete

The IDCODE for the TNETX4080 is shown in Table 13.

Table 13. IEEE Std 1149.1 (JTAG) ID Code

VARIANT	PART NUMBER	MANUFACTURE	LEAST SIGNIFICANT BIT
BIT 31 BIT 28	BIT 27 BIT 12	BIT 11 BIT 1	BIT 0
0000	1011000111110111	00000010111	1

frame routing

VLAN support

The internal routing engine supports the IEEE Std 802.1Q VLANs as shown in Figure 6 and described in the following paragraphs.

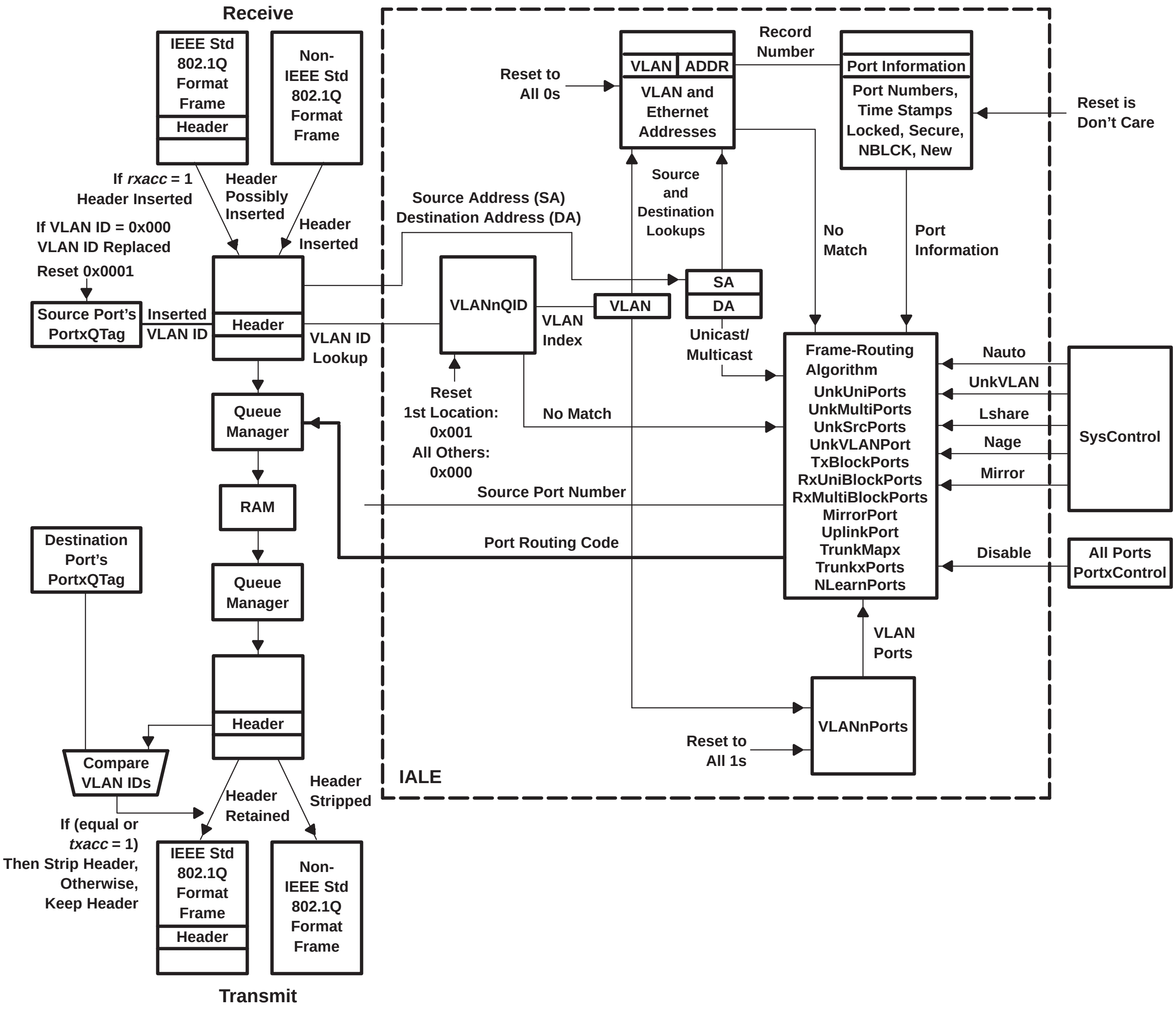


Figure 6. VLAN Overview

IEEE Std 802.1Q tags – reception

By the time the IALE examines the received frame, it contains an IEEE Std 802.1Q tag header (after the source address). The tag used depends on the port configuration. If the port is configured as an access port, IALE always uses the default VID programmed for this port and assumes that all received frames on this port are untagged. If the frame already contained a tag, it is tagged again. If the port is programmed as a non-access port, the tag added depends on the received frame. If the frame is not tagged or the value of the tag field is 0x000, the default port VID is used to internally tag the frame. Otherwise, the VID contained in the frame is used by the IALE.

The IALE supports 16 of the possible 4096 VIDs that can be encoded within the IEEE Std 802.1Q tag. The VID in the received frame is compared with these 16 VLAN IDs to see which (if any) of them it matches.

unknown VLAN

If there is no match, the rest of the address-lookup process is abandoned. A new VLAN interrupt is provided to the attached CPU. The source address, VLAN ID, and port information is provided in internal registers so that the CPU can determine if it wants to add this VID to the lookup table. If the destination address is unicast, the frame is discarded. If the destination address is a multicast/broadcast, the frame is forwarded based on a programmable port mask.

known VLAN

If there is a match, the VLAN index associated with this VID, together with the destination and source address, is forwarded to the address lookup and subsequent routing process. (Only one of the VIDs matches if they have been programmed correctly. If more than one matches, the hardware chooses one of them.)

new VLAN member

The IALE checks to see if the source port already has been declared as a member of this VLAN. If not, an interrupt is provided to allow the attached CPU to add this port as a new member of the VLAN.

IEEE Std 802.1Q header – transmission

The IEEE Std 802.1Q header is carried within the frame to the transmitting MAC port, where the decision to strip out the header before transmission is made, based on the port configuration. If the port is configured as an access port, the tag is stripped before transmission. If the frame is only 64 bytes long, four bytes of pad (0s) are inserted between the end of the data and the start of the CRC word (a new CRC value is calculated and inserted in the frame). Three, two, and one byte(s) are inserted for 65-, 66-, and 67-byte frames, respectively. If the port is configured as a nonaccess port, the VID is compared with the default port VID. If they match, the header is stripped; otherwise, the header is retained.

If the frame is transmitted to the NM port, no header stripping occurs; the frame is transmitted unaltered. It may contain one or two IEEE Std 802.1Q headers, depending on how the frame is received.

address maintenance

The addresses within the IALE can be maintained automatically by the TNETX4080, where addresses are learned/updated from the wire and deleted, using one of two aging algorithms looked up during frame-routing determination. Multicast addresses are not automatically learned or aged. The attached CPU can add/update, find, or delete address records (see *TNETX4090 Programmer's Reference Guide*, literature number SPAU003, for details) via the DIO interface.

The learning and aging processes are completely independent. This allows addresses to be automatically learned from the wire, but allows the CPU to manage the aging process under software control.

spanning-tree support

Each port provides independent controls to block reception or transmission of frames, learning of addresses, or disable the port on a per-port basis. Blocking can be overridden to allow reception or transmission of spanning-tree frames.

PRODUCT PREVIEW

aging algorithms

time-threshold aging

When learning addresses, the IALE adds the address to the table and tags it with a time stamp. If another frame is received with this address, the time stamp is refreshed. If the aging counter expires before another frame is received from this source address, the address is deleted from the table. If the table is full, the oldest address is deleted to make room for a new address, even if the age for this address has not expired.

table-full aging

In table-full aging, the oldest address (or one of the oldest addresses if there is more than one) automatically is deleted from the IALE records only if the table is full, and a new address must be added to the table. In this mode, the age stamp for the addresses is not refreshed.

frame-routing determination

When a frame is received, its 48-bit destination and source addresses are extracted and the VLAN index is determined as described in *VLAN Support*. The destination address and VLAN index are then looked up in the IALE records to determine if they exist. If a match is found, the information associated with the record is passed on to the frame-routing algorithm. Figure 7 provides a flow diagram of the routing algorithm. For details of the register information referred to in Figure 7, see the *TNETX4090 Programmer's Reference Guide*, literature number SPAU003.

The source address and VLAN index combination also are looked up in the IALE records to determine if they exist. If a match is found, additional information is provided to the routing process (for details see the *TNETX4090 Programmer's Reference Guide*, literature number SPAU003).

PRODUCT PREVIEW

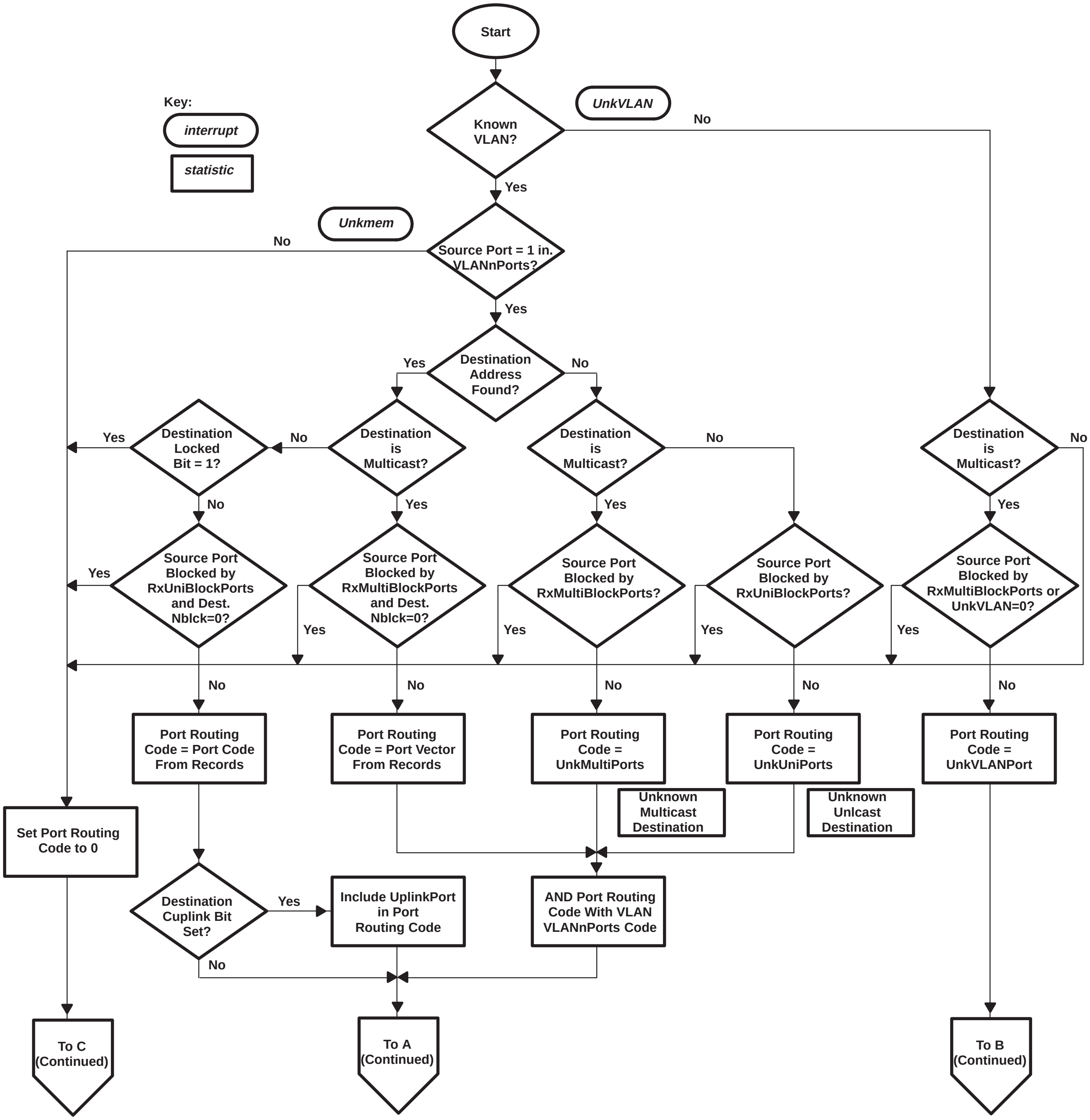

Start
Key:
interrupt
statistic
UnkVLAN
Known VLAN?
No
Yes
Unkmem
No
Source Port = 1 in. VLANnPorts?
Yes
Yes
Destination Address Found?
No
Yes
Destination Locked Bit = 1?
No
Destination is Multicast?
Destination is Multicast?
No
Destination is Multicast?
No
No
Yes
Yes
Yes
Yes
Source Port Blocked by RxUniBlockPorts and Dest. Nblck=0?
Source Port Blocked by RxMultiBlockPorts and Dest. Nblck=0?
Source Port Blocked by RxMultiBlockPorts?
Source Port Blocked by RxUniBlockPorts?
Source Port Blocked by RxMultiBlockPorts or UnkVLAN=0?
Yes
Yes
Yes
Yes
No
No
No
No
No
Port Routing Code = Port Code From Records
Port Routing Code = Port Vector From Records
Port Routing Code = UnkMultiPorts
Port Routing Code = UnkUniPorts
Port Routing Code = UnkVLANPort
Unknown Multicast Destination
Unknown Unlcast Destination
Set Port Routing Code to 0
Destination Cuplink Bit Set?
Yes
Include UplinkPort in Port Routing Code
AND Port Routing Code With VLAN VLANnPorts Code
No
To C (Continued)
To A (Continued)
To B (Continued)

Figure 7. Frame-Routing Algorithm

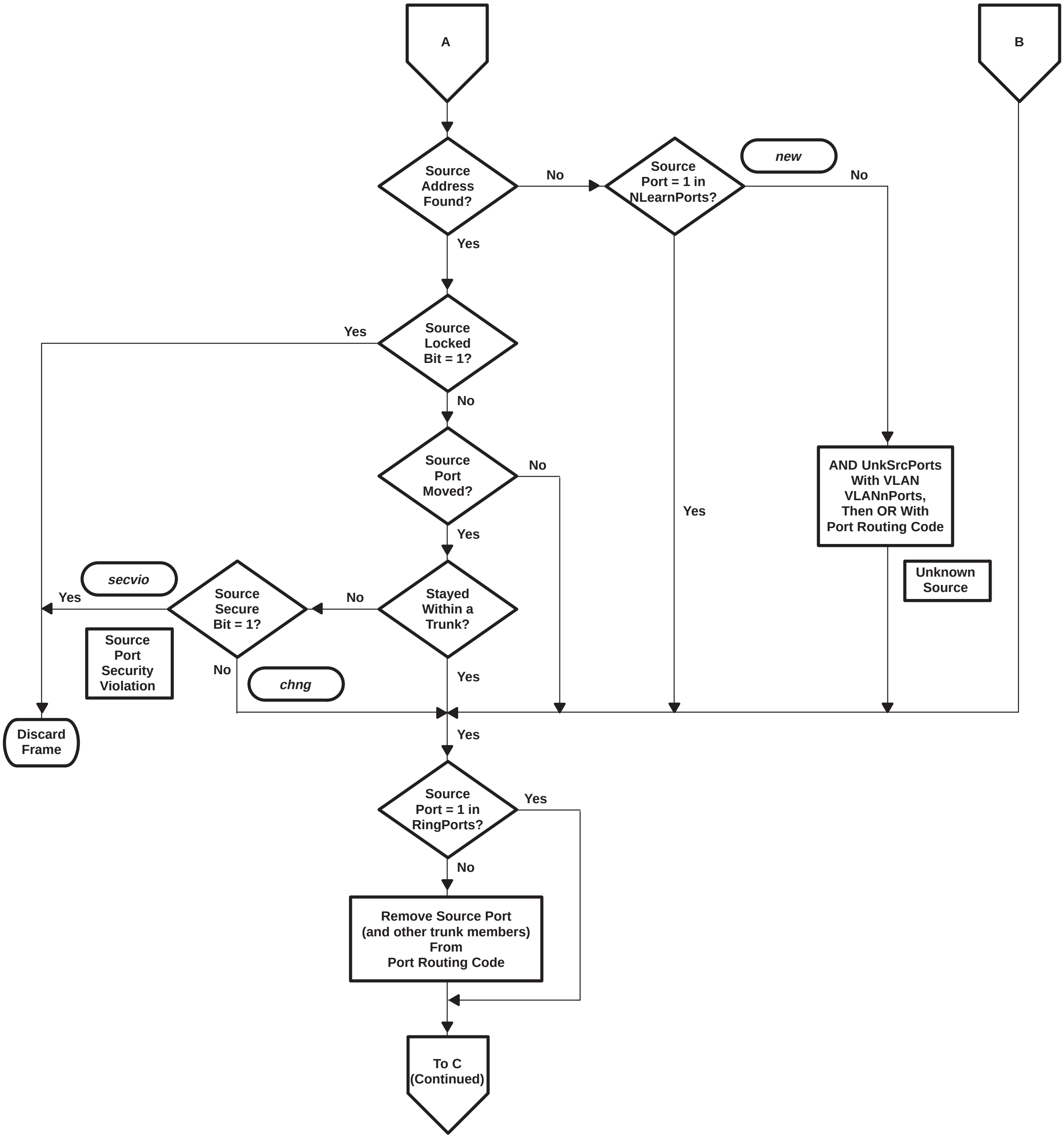
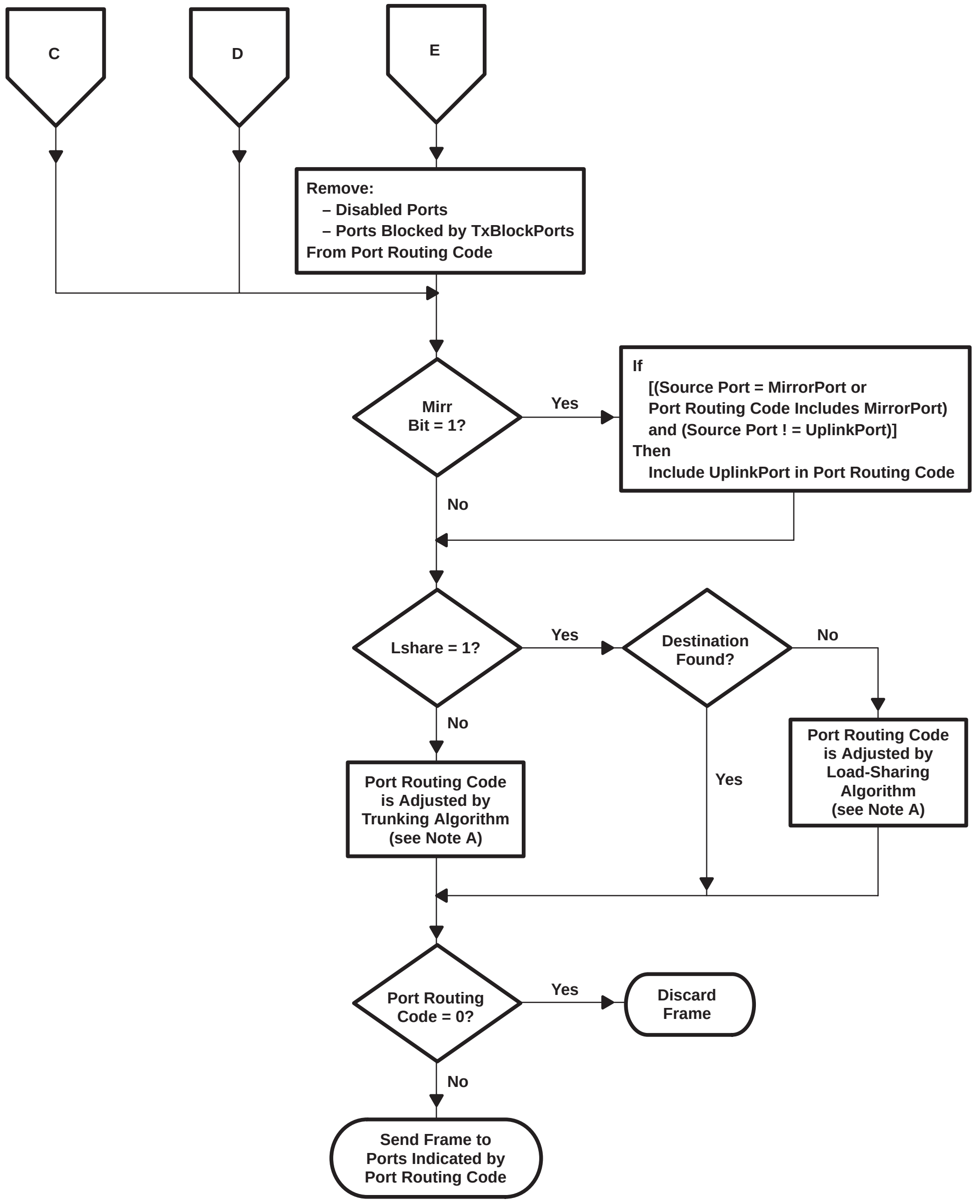

A
B
Source Address Found?
No
Yes
Source Port = 1 in NLearnPorts?
new
No
Source Locked Bit = 1?
Yes
No
Source Port Moved?
No
Yes
Yes
AND UnkSrcPorts With VLAN VLANnPorts, Then OR With Port Routing Code
Unknown Source
secvio
Yes
Source Secure Bit = 1?
No
Stayed Within a Trunk?
Source Port Security Violation
No
chng
Yes
Discard Frame
Yes
Source Port = 1 in RingPorts?
Yes
No
Remove Source Port (and other trunk members) From Port Routing Code
To C (Continued)

Figure 7. Frame-Routing Algorithm (Continued)

PRODUCT PREVIEW


C
D
E
Remove:
– Disabled Ports
– Ports Blocked by TxBlockPorts
From Port Routing Code
Mirr Bit = 1?
Yes
No
If
[(Source Port = MirrorPort or
Port Routing Code Includes MirrorPort)
and (Source Port ! = UplinkPort)]
Then
Include UplinkPort in Port Routing Code
Lshare = 1?
Yes
No
Destination Found?
No
Yes
Port Routing Code is Adjusted by Trunking Algorithm (see Note A)
Port Routing Code is Adjusted by Load-Sharing Algorithm (see Note A)
Port Routing Code = 0?
Yes
No
Discard Frame
Send Frame to Ports Indicated by Port Routing Code

NOTE: See *Port Trunking/Load Sharing*

Figure 7. Frame-Routing Algorithm (Continued)

port routing code

The IALE creates a port routing code in which each bit (marked with a 1) represents a potential destination port for the frame. This code is modified as it proceeds through the frame-routing algorithm. If the final code is all 0s, then the frame is discarded. If it is not, then the frame is transmitted on every port marked by a 1 within the code.

removal of source port

Normally, the IALE does not route a frame to a port on which it was received. The port routing code is examined to see if the source port is included. If so, the port routing code is modified to remove the source port.

If the bit in RingPorts corresponding to the port that received the frame is set, the port routing code is not modified to remove the source port. This is required for connecting the port to other like switches in a ring topology.

If the source port is a member of a trunk (see *Trunking*), then all the other ports that are members of the same trunk also are removed from the port routing code.

port mirroring

It is possible to copy (or mirror) all frames that are received by and transmitted from a port to another designated port, using the mirror port register.

It also is possible to mirror frames destined for a particular MAC address by using the copy-uplink feature. When a frame specifies the destination address with the copy-uplink feature enabled, frames are copied to the specified port.

copy-to-uplink (cuplink)

If destination address is a unicast and the cuplink bit of its address record has been set to a 1 (via a DIO add), and when a frame specifies that destination, a copy of the frame is sent to the port specified in the UplinkPort register.

port trunking/load sharing

Trunking allows two or more ports to be connected in parallel between switches to increase the bandwidth between those devices. The trunking algorithm determines on which of these ports a frame is transmitted, so that the load is spread evenly across these ports.

The TNETX4080 supports a maximum of four trunk groups for the 10-/100-Mbit ports. The port members of a trunk group are software configurable via the DIO interface. Trunk-port determination is the final step in the IALE frame-routing algorithm. Once the destination port(s) for a frame have been determined, the port routing code is examined to see if any of the destination port(s) are members of a trunk. If so, the trunking algorithm is applied to select the port within the trunk that transmits the frame – it may or may not be the one currently in the port routing code. To determine the destination port within a trunk, bits 3–1 of the source and destination address are XORed to produce a map index. This map index is used to index to a group of eight internal registers to determine the destination port (for details see the *TNETX4090 Programmer's Reference Guide*, literature number SPAU003). Port trunking uses the destination/source address pairs to route the traffic to balance the load more evenly across the trunked ports. Since the same destination/source address pair always uses the same port to route the traffic, this also makes it much easier to debug network problems.

Load sharing is similar to trunking , but with two slight differences. It uses the trunking algorithm only once when the destination address is unknown. Once the destination address has been learned, it uses the port routing code associated with the destination address.

If the destination is unknown, the map index is derived from only the source address. If a server is communicating with a large number of different clients, then, since the source address is the same, it is possible to have a very poor distribution of traffic.

- If the destination address is found in the IALE records when it is looked up, the port routing code is not adjusted by the load-sharing algorithm.
- The 3-bit map index is determined only from the source address, as follows:
  - Bits 47–32 are XORed to produce the most significant bit of the map index.
  - Bits 31–16 are XORed to produce the middle of the map index.
  - Bits 15–0 are XORed to produce the least significant bit of the map index.

port trunking example

This example shows how to set up the TNETX4080 to support two port trunks. The first trunk group consists of ports 1, 3, 5, and 7 (see Table 14); the second trunk group consists of ports 0, 2, and 6 (see Table 15).

Table 14. Trunk Group 0 Port Membership (Trunk0Ports Register)

PORT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	0	1	0	1	0	1	0

Table 15. Trunk Group 1 Port Membership (Trunk1Ports Register)

PORT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	0	0	0	1	0	1

The TrunkMapx registers are used to control the distribution of traffic across the ports within a trunk group. In this example, the traffic for trunk group 0 has been equally distributed 25% (this assumes that bits 3–1 of the MAC addresses are random enough to give an even distribution) for each of the four ports in the trunk. For any given source and destination address pair, the traffic always uses the same port within the trunk. This ensures that packets do not get disordered on the trunk ports. Note that since port 4 is not a member of any port trunk group, all the entries for this port have been set to 1. In fact, funtionally, this can be thought of as a single-port trunk (see Table 16).

Table 16. TrunkMapx Register Settings (for Traffic Distribution on Trunk Groups 0 and 1)

MAP INDEX	TRUNK PORT							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	0	1	0	1	0	0	1	0
1	0	0	0	1	1	1	0	0
2	0	0	1	1	0	0	0	1
3	1	1	0	1	0	0	0	0
4	0	0	0	1	0	1	1	0
5	0	0	0	1	1	0	0	1
6	0	1	1	1	0	0	0	0
7	1	0	0	1	0	1	0	0

flow control

The switch incorporates two forms of flow control: collision based, and IEEE Std 802.3 pause frames.

In either case, the switch recognizes when it is becoming congested by monitoring the size of the free-buffer queue. When the number of free buffers drops below the specified threshold, the switch prevents frames from entering the device by issuing the flow control appropriate to each port’s current mode of operation. This prevents reception of any more frames on those ports until the frame backlog is reduced and the number of free buffers has risen above the threshold, at which point, flow control ceases and frames again can be received. The default free-buffer threshold after a hardware reset is chosen to ensure that all ports simultaneously can start reception of a maximum-length frame and ensure complete reception.

The purpose of flow control is to reduce the risk of data loss if a long burst of activity caused the switch to backlog frames to the point where the memory system is full. However, there is no way to prevent frame reception on those ports operating in full-duplex mode that have not negotiated IEEE Std 802.3 flow control. Such ports can exhaust the free-buffer queue, with subsequent data loss.

Each 10-/100-Mbit/s port can request collision or IEEE Std 802.3x flow control through internal registers.

Flow control (both forms) is globally enabled/disabled. Each individual port can request half full-duplex or IEEE Std 802.3 flow to be negotiated by the PHY device.

In full duplex, a port does not start transmitting a new frame if the collision pin is active, although the value of this pin is ignored at other times.

collision-based flow control

Collision-based flow control provides a means of preventing frame reception for ports that are operating in half-duplex mode. While the number of free buffers is fewer than the specified threshold, ports in this state that are not currently transmitting generate collisions when they start to receive a frame. The jam sequence transmitted (55.55.55.55.55.55.55.5D.DD.DD.DD.DD (hex) begins approximately when the source address starts to receive. Port 8 begins jam sequence after approximately eight bytes of payload data (i.e., after the source address) have been received.

These forced collisions are not limited to a maximum of 16 consecutive collisions, and are independent of the normal backoff algorithm.

PRODUCT PREVIEW

IEEE Std 802.3 flow control

The TNETX4080 supports collision-based flow control for ports in half-duplex mode and IEEE Std 802.3x flow control for ports in full-duplex mode. The flow bit in the SysControl register determines the action that will be taken when back pressure is needed, that is, when there are insufficient resources to handle an inbound packet. The holb bit in the SysControl register determines when back pressure is needed.

- If flow = 0, packets are discarded at the ingress port when insufficient resources are available to handle them.
- If flow = 1, ports in half-duplex mode cause collisions to avoid accepting packets, ports in full-duplex mode whose link partners negotiated to accept pause packets will send them, otherwise, packets are dropped.
- With holb = 0, back pressure is applied to all ports when the number of buffers in the global pool is down to the value in the FlowThreshold register (or half of this value if the packet arrives at a port in gigabit mode). This prevents the reception of more frames at any port until the frame backlog is reduced and the number of free buffers has risen above the threshold. When this happens, back pressure is removed from all ports and packets can be received. The value in FlowThreshold should be set so that all ports can complete reception of a maximum-size frame, that is, each port should have enough time to activate the flow mechanisms without dumping a frame for which reception has started.
- If holb = 1, back pressure is applied as when holb = 0, or to an individual port when the number of buffers held in memory for data that arrived on that port is greater than the available pool remaining. This assumes that FlowThreshold is set small enough that this mechanism does not affect the back pressure in this mode. An example is:

When port-A's traffic begins to backlog in memory [no matter to what port(s) it is destined], back pressure will be applied when the amount of data backed up is greater than the available pool (about half the buffers are assigned to data from port A). If A's data stays backlogged and if data arriving at port B also begins to backlog in memory, back pressure will be applied to port B when its data amounts to one-fourth of the buffer pool, or half of the half left after port A had back pressure applied. When port-A's traffic begins to exit the switch, port A stays back pressured until its data is equal to one-third of the total. As buffers become available, port B is allowed to consume up to one-third of the buffer pool (each backlogged total is compared with the buffers available). In this mode, only the stations that have caused their fair share of buffers to be removed from the available pool are back pressured.

Setting holb = 1 activates circuitry that attempts to prevent a backlogged conversation stalling other port traffic by using up all the memory buffers. Because the number of buffers charged to a particular port always is compared with the number of buffers left, there is no threshold register for this mode.

other flow-control mechanisms

hardware flow control

If a port were in MII mode and full duplex, normally, its Mxx_COL would not be needed. Hardware flow control has been added by preventing the start of transmission of a frame if the Mxx_COL is high.

multicast limit

Because buffer resources for multicast (or broadcast) frames are released only when the last port has transmitted the frame, multicast packets to fast and slow ports are released at the slow-port rate. Multicast traffic from a fast port to a slow port could cause back pressure to be applied to the source port, blocking input from that port. This occurs even if at least one of the ports in the multicast could have kept pace with the inbound multicast packets. The MCastLimit register can limit the number of multicast packets allowed to be pending at any output port. When the limit is reached, that port is not added to the routing vector of the next multicast packet for which it was eligible. Eligibility is restored when the number of backlogged packets again is below the limit. The MCastLimit register allows the user to set the ports subject to this restriction, and set the limit in eight binary steps from 2 to 256. The spanning-tree BPDU multicast packet is exempt from this limit.

system test capabilities

RDRAM

The external RDRAM can be read and written using regular DIO accesses following a stop. Individual bytes can be read and written. However, as the RDRAM memory actually is accessed in 128-byte pages, performing 128-byte accesses is the most efficient.

To access the RDRAM, the TNETX4080 must not be operating. The user must perform a reset and not set start in SysControl. Both start and initd in SysControl must be 0. In addition, rdinit in SysTest must be set, indicating that the RDRAM has initialized. This initialization sequence occurs automatically after a hard reset.

Read or write accesses to RDRAM are invoked via rdram and rdwrite in SysTest. Setting rdram to 1 causes a 128-byte transfer between the device and the RDRAM memory to be initiated.

- The transfer direction is determined by rdwrite.
- The external memory byte address for the access is specified by ramaddress in RAMAddress.
- Data to be read from or written to RDRAM is accessed indirectly via RAMData.

writing RDRAM

Writing to RDRAM memory is accomplished as follows:

1. Write the byte address for the access to ramaddress in RAMAddress.
2. Write the data for the access to RAMData. Up to 64 bytes can be written, if all but the six least significant bits of the address are the same for all the data. Inc in RAMAddress can be used to autoincrement the address.
3. Set rdwrite = 0 and rdram = 1 (these can be written simultaneously).
4. If required, poll rdram until it becomes 0. This indicates that the write has completed.

reading RDRAM

Reading from RDRAM memory is accomplished as follows:

1. Write the byte address for the access to ramaddress in RAMAddress.
2. Set rdwrite = 1 and rdram = 1 (these can be written simultaneously).
3. Poll rdram until it becomes 0. This indicates that the read has completed.
4. Read the data for the access to RAMData. Up to 64 bytes can be read, provided that all but the six least significant bits of the address are the same for all the data. Inc in RAMAddress can be used to autoincrement the address.

PRODUCT PREVIEW

internal wrap test

Internal wrap mode causes some or all of the Ethernet MACs to be configured to loop back transmitted data into the receive path. This allows a frame to be sent into a designated source port and then selectively routed successively to and from ports involved in the test, before finally transmitting the frame out of the original port. By varying the number of ports between which the frame is forwarded, the potential fault capture area is expanded or constrained.

Intwrap in SysTest determines which ports loop back. Ports 0 or 7 can be configured to not loop back, allowing them to be used as the start/end port for the test. Alternatively, the NM port (accessed via DIO) can be used for this purpose, with all MII ports configured to loop back.

For a frame to be forwarded from one port to another in this fashion, the switch must be programmed as follows:

- Assign a unique VID to each of the PortxQTag registers, and program these tags into the VLANnQID registers.
- The VLANnPorts register associated with each of the VLANnQID registers should have only one bit set, indicating to which port frames containing that IEEE Std 802.3 tag should be routed.
- Rxacc and Txacc for each port must be 1. This causes the port to add the VID from its PortxQTag to the frame on reception, and strip the tag before transmission.
- The destination address of the frames to be applied is not known, and UnkUniPorts and UnkMultiPorts should be all 1s.

This causes the following:

1. The VID from the source port PortxQTag register is added to the frame upon reception. As the address of the frame is unknown, it is forwarded to the AND of the appropriate VLANnPorts and UnkUniPorts (unicast) or UnkMultiPorts (multicast). As VLANnPorts should contain only a single 1, this should be a single port.
2. The frame is transmitted from the destination port selected in 1. Its VLAN tag is stripped beforehand; the frame loops back to the receive path.
3. Steps 1 and 2 are repeated, but the VID added upon reception is different from the one just stripped off at transmission. This means a different VLANnPorts register is used to determine the destination.

   The port order shown in Figure 8 is sequential, but the actual order depends on how ports are paired in the VCANn Ports registers, and how the PortxQTag registers are assigned.
4. Eventually, the frame is sent to a port that is not configured for loopback and leaves the switch.

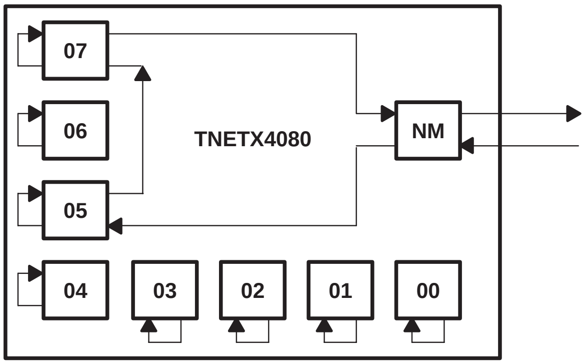


Figure 8. Internal Wrap Example

The operational status of the PHYs or external connections to the device do not have to be considered or assumed good, when in internal loopback mode.

PRODUCT PREVIEW

duplex wrap test

Duplex wrap test is similar to internal wrap mode (see Figure 9). The ports can be set to accept frame data that is wrapped at the PHY. This permits network connections between the device and the PHY to be verified. Any port can be the source port (not just the NM port as shown in Figure 8). By using multicast/broadcast frames, traffic can be routed selectively between ports involved in the test or return the frame directly before retransmission on the uplink. Software control of the external PHYs is required to configure them for loopback.

Duplex frame-wrap test mode is selected by setting dpwrap in SysTest. When selected, the port is forced into full duplex, allowing it to receive frames it transmits.

The switch is configured in the same manner as internal wrap.

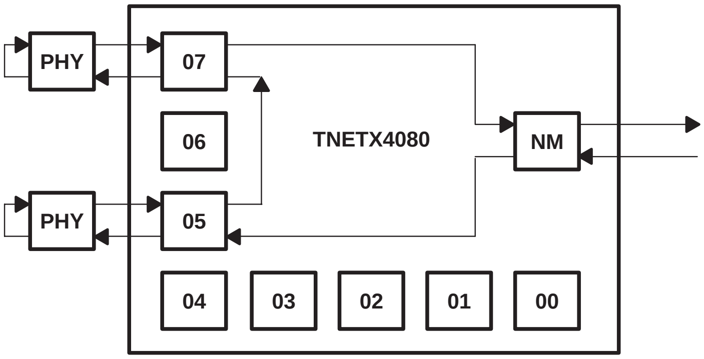


Figure 9. Duplex Wrap Example

PRODUCT PREVIEW

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage range, $V_{DD(2.5)}$ (see Note 1)	–0.5 V to 2.7 V
Supply voltage range, $V_{DD(3.3)}$ (see Note 1)	–0.5 V to 3.6 V
Supply voltage range, V_{DD}a (see Note 1)	–0.5 V to 2.7 V
Input voltage range, V_I	–0.5 V to $V_{DD(3.3)}$ + 0.4 V
Output voltage range, V_O	–0.5 V to $V_{DD(3.3)}$ + 0.5 V
Input voltage range (RSL), V_{IR}	V_{REF} – 0.35 to V_{REF} + 0.8
Output voltage range (RSL), V_{OR}	0.0 to V_{DD}
Thermal impedance, junction-to-ambient package, $Z_{\theta JA}$: Airflow = 0	11.11°C/W
Airflow = 100 ft/min	9.61°C/W
Thermal impedance, junction-to-case package, $Z_{\theta JC}$	0.94°C/W
Operating case temperature range, T_C	0°C to 95°C
Storage temperature range, T_{stg}	–65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" can cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods can affect device reliability.

NOTE 1: All voltage values are with respect to GND.

recommended operating conditions

		MIN	NOM	MAX	UNIT
$V_{DD(2.5)}$	Supply voltage	2.25	2.5	2.7	V
$V_{DD(3.3)}$	Supply voltage	3	3.3	3.6	V
V_{REF}	RSL reference voltage	1.8	2.0	2.2	V
V_I	Input voltage	0		$V_{DD(3.3)}$	V
V_O	Output voltage	0		$V_{DD(3.3)}$	V
V_{IH}	High-level input voltage	2		$V_{DD(3.3)}$	V
V_{IL}	Low-level input voltage (see Note 2)	0		0.8	V
I_{OH}	High-level output current			–2	mA
I_{OL}	Low-level output current (except $\overline{\text{LED\_DATA}}$)			2	mA
I_{OL}	$\overline{\text{LED\_DATA}}$ (pin AE19)	0		8	mA
V_{IHR}	High-level input voltage (RSL)	V_{REF}+0.35		V_{REF}+0.8	V
V_{ILR}	Low-level input voltage (RSL) (see Note 2)	V_{REF}–0.35		V_{REF}–0.8	V
I_{OHR}	High-level output current (RSL)	–10		10	µA
I_{OLR}	Low-level output current (RSL)	0		80	mA

NOTE 2: The algebraic convention, in which the more-negative (less-positive) limit is designated as a minimum, is used for logic-voltage levels only.

electrical characteristics over recommended operating conditions (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
V_{OH}	High-level output voltage	I_{OH} = rated†	$V_{DD(3.3)}$–0.5			V
V_{OL}	Low-level output voltage	I_{OL} = rated†			0.5	V
I_{OZ}	High-impedance-state output current	$V_O = V_{DD}$ or GND			±10	µA
I_{IH}	High-level input current	$V_I = V_{IH}$			1	µA
I_{IL}	Low-level input current	$V_I = V_{IL}$			–1	µA
V_{OHR}	High-level output voltage (RSL)	I_{OHR} = 0.25 mA	2		V_{DD}	V
V_{OLR}	Low-level output voltage (RSL)	I_{OLR} = 1 mA	0		0.4	V
I_{OZR}	High-impedance-state output current (RSL)					µA
I_{IHR}	High-level input current (RSL)					µA
I_{ILR}	Low-level input current (RSL)					µA
C_i	Capacitance, input			6		pF
C_o	Capacitance, output			6		pF

† Values specified in recommended operating conditions

timing requirements over recommended operating conditions

IEEE Std 1149.1 (JTAG) interface

control signals

$\overline{\text{RESET}}$ (see Figure 10)

NO.			MIN	MAX	UNIT
1	$t_{w(RESETP)}$	Pulse duration, $\overline{\text{RESET}}$ low at power up	100		µs
1	$t_{w(RESET)}$	Pulse duration, $\overline{\text{RESET}}$ low at other times	4		t_{cycle}

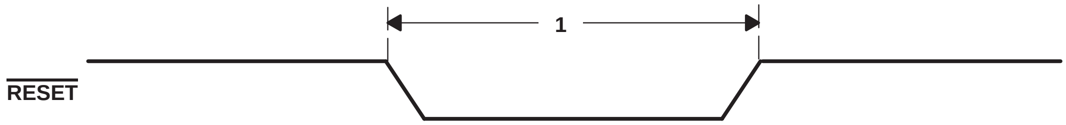


Figure 10. $\overline{\text{RESET}}$

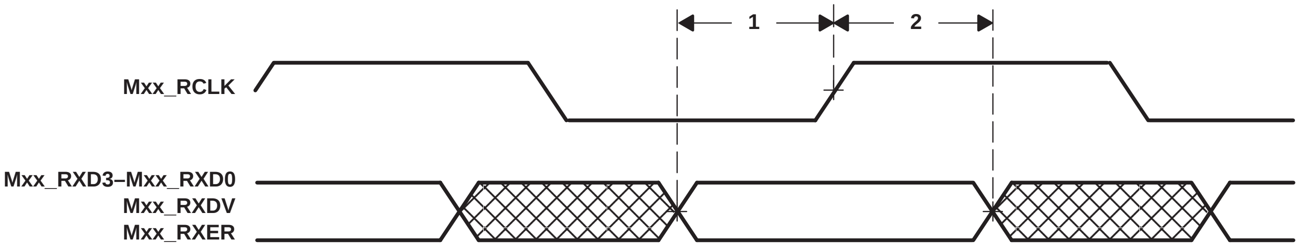
MII (ports 0–7)

Figures 11–13 show the timing for the eight MIIs operating at either 10-Mbit/s or 100-Mbit/s, and the GMII operating at 100-Mbit/s.

Mxx_CRS and Mxx_COL are driven asynchronously by the PHY. Mxx_RXD3–Mxx_RXD0 is driven by the PHY on the falling edge of Mxx_RCLK. Mxx_RXD3–Mxx_RXD0 timing must be met during clock periods in which Mxx_RXDV is asserted. Mxx_RXDV is asserted and deasserted by the PHY on the falling edge of Mxx_RCLK. Mxx_RXER is driven by the PHY on the falling edge of Mxx_RCLK.

MII receive (see Figure 11)

NO.			MIN	MAX	UNIT
1	$t_{su(Mxx\_RXD)}$	Setup time, Mxx_RXD3–Mxx_RXD0 valid before Mxx_RCLK↑	8		ns
1	$t_{su(Mxx\_RXDV)}$	Setup time, Mxx_RXDV valid before Mxx_RCLK↑	8		ns
1	$t_{su(Mxx\_RXER)}$	Setup time, Mxx_RXER valid before Mxx_RCLK↑	8		ns
2	$t_{h(Mxx\_RXD)}$	Hold time, Mxx_RXD3–Mxx_RXD0 valid after Mxx_RCLK↑	8		ns
2	$t_{h(Mxx\_RXDV)}$	Hold time, Mxx_RXDV valid after Mxx_RCLK↑	8		ns
2	$t_{h(Mxx\_RXER)}$	Hold time, Mxx_RXER valid after Mxx_RCLK↑	8		ns



NOTE: For port 8, M08_RFCLK is used for the transmit clock input.

Figure 11. MII Receive

Mxx_CRS and Mxx_COL are driven asynchronously by the PHY. Mxx_TXD3–Mxx_TXD0 is driven by the reconciliation sublayer synchronous to Mxx_TCLK. Mxx_TXEN is asserted and deasserted by the reconciliation sublayer synchronous to the Mxx_TCLK rising edge. Mxx_TXER is driven synchronous to the rising edge of Mxx_TCLK.

MII transmit (see Figure 12)

NO.			MIN	MAX	UNIT
1	$t_{d(Mxx\_TXD)}$	Delay time, from Mxx_TCLK↑ to Mxx_TXD3–MxxTXD0 valid	0	25	ns
1	$t_{d(Mxx\_TXEN)}$	Delay time, from Mxx_TCLK↑ to Mxx_TXEN valid	0	25	ns
1	$t_{d(Mxx\_TXER)}$	Delay time, from Mxx_TCLK↑ to Mxx_TXER valid	0	25	ns

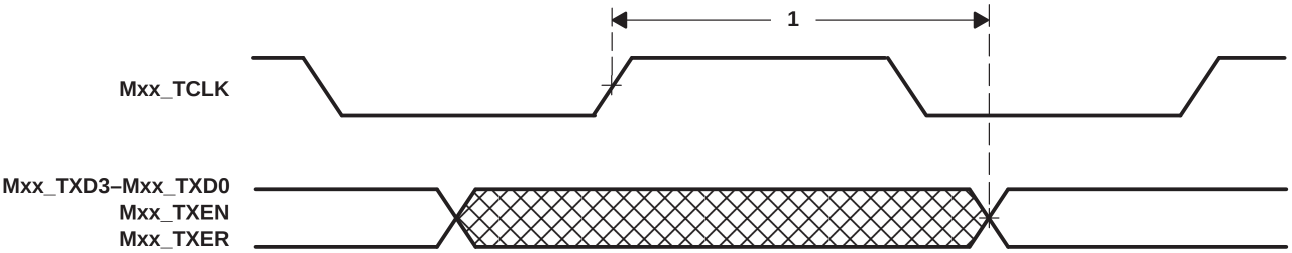


Figure 12. MII Transmit

MII clock (see Figure 13)

NO.			MIN	MAX	UNIT
1	$t_{LOW(Mxx\_CLK)}$	Pulse width low Mxx_RCLK, Mxx_TCLK low	40%	60%	
2	$t_{HIGH(Mxx\_CLK)}$	Pulse width high Mxx_RCLK, Mxx_TCLK high	40%	60%	
3	$t_{C(Mxx\_CLK)}$	Cycle time Mxx_RCLK, Mxx_TCLK	400	40	ns
4	Mxx_CLK frequency		2.5	25	MHz

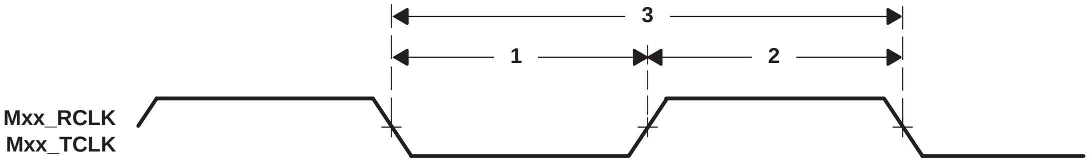


Figure 13. MII Clock

PRODUCT PREVIEW

PRODUCT PREVIEW

RDRAM interface

RDRAM (see Figure 14)

NO.			MIN	MAX	UNIT
1	$t_{c(DX\_CLK)}$	Cycle time DTX_CLK, DRX_CLK	3.33	3.33	ns
2, 3	$t_{w(DX\_CLK)}$	Pulse duration, DTX_CLK, DRX_CLK low or high	45%	55%	$t_{c(DX\_CLK)}$
4, 5	$t_{w(TICK)}$	Pulse duration, tick time	0.5	0.5	t_{cycle}
6, 8	$t_{su(DBUS\_DATA)}$	Setup time, DBUS_DATA before tick	0.35		ns
7, 9	$t_{h(DBUS\_DATA)}$	Hold time, DBUS_DATA after tick	0.35		ns
10, 11	$t_{d(DBUS\_OUT)}$	Delay time, DBUS_DATA, DBUS_CTRL, DBUS_EN from tick	0.635	1.438	$t_{c(DX\_CLK)}$
	t_{cycle}†	Cycle time, internal clock		4	$t_{c(DX\_CLK)}$

† Not shown in Figure 14 due to scale

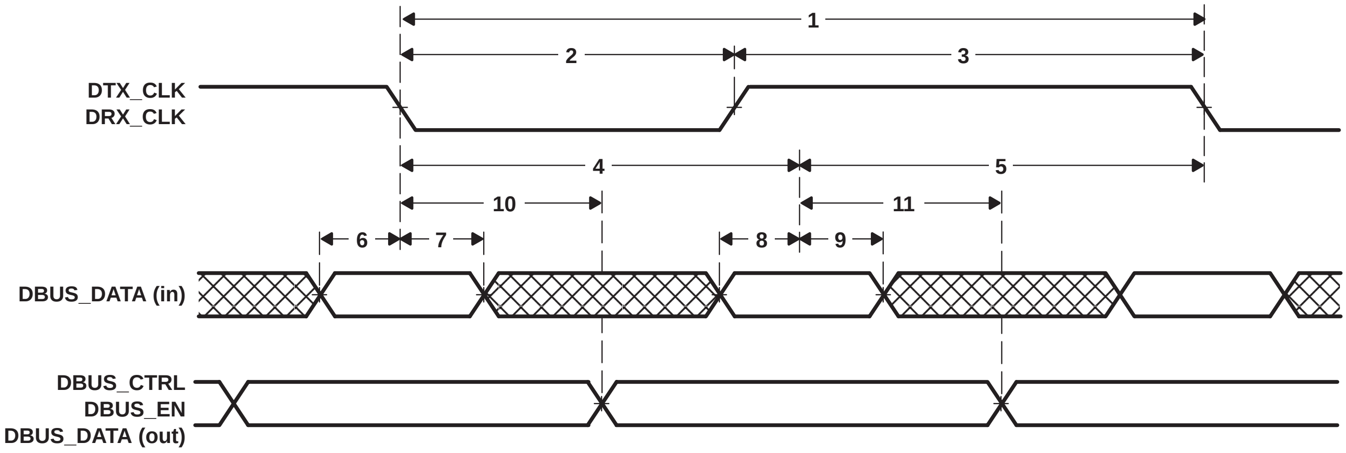


Figure 14. RDRAM

DIO interface

The DIO interface is simple and asynchronous to allow easy adaptation to a range of microprocessor devices and computer system interfaces.

DIO and DMA writes (see Figure 15)

NO.			MIN	MAX	UNIT
1	$t_{w(SCS)}$	Pulse duration, $\overline{\text{SCS}}$↓	$2t_c$		ns
2	$t_{su(SRNW)}$	Setup time, SRNW valid before $\overline{\text{SCS}}$↓	0		ns
3	$t_{su(SAD)}$	Setup time, SAD1–SAD0, $\overline{\text{SDMA}}$ valid before $\overline{\text{SCS}}$↓	0		ns
4	$t_{su(SDATA)}$	Setup time, SAD7–SAD0 valid before $\overline{\text{SCS}}$↓	0		ns
5	$t_{h(SRNW)}$	Hold time, SRNW low after $\overline{\text{SRDY}}$↓	0		ns
6	$t_{h(SAD)}$	Hold time, SAD1–SAD0, $\overline{\text{SDMA}}$ valid after $\overline{\text{SRDY}}$↓	0		ns
7	$t_{h(SDATA)}$	Hold time, SAD7–SAD0 valid after $\overline{\text{SRDY}}$↓	0		ns
8	$t_{h(SCSL)}$	Hold time, $\overline{\text{SCS}}$ low after $\overline{\text{SRDY}}$↓	0		ns
9	$t_{d(SRDYZH)}$	Delay time from $\overline{\text{SCS}}$↓ to $\overline{\text{SRDY}}$↑		10	ns
10	$t_{d(SRDYHL)}$	Delay time from $\overline{\text{SCS}}$↓ to $\overline{\text{SRDY}}$↓	$2t_c$	†	ns
11	$t_{d(SRDYLH)}$	Delay time from $\overline{\text{SCS}}$↑ to $\overline{\text{SRDY}}$↑	t_c	$2t_c$+10	ns
12	$t_{h(SCSH)}$	Hold time, $\overline{\text{SCS}}$ high after $\overline{\text{SRDY}}$↑	0		ns
13	$t_{w(SRDY)}$	Pulse duration, $\overline{\text{SRDY}}$↑		t_c	ns

† When the switch is performing certain internal operations (e.g., EEPROM load), there is a delay of up to 20 ms (24C02) or 800 ms (24C08) between $\overline{\text{SCS}}$ being asserted and $\overline{\text{SRDY}}$ being asserted.

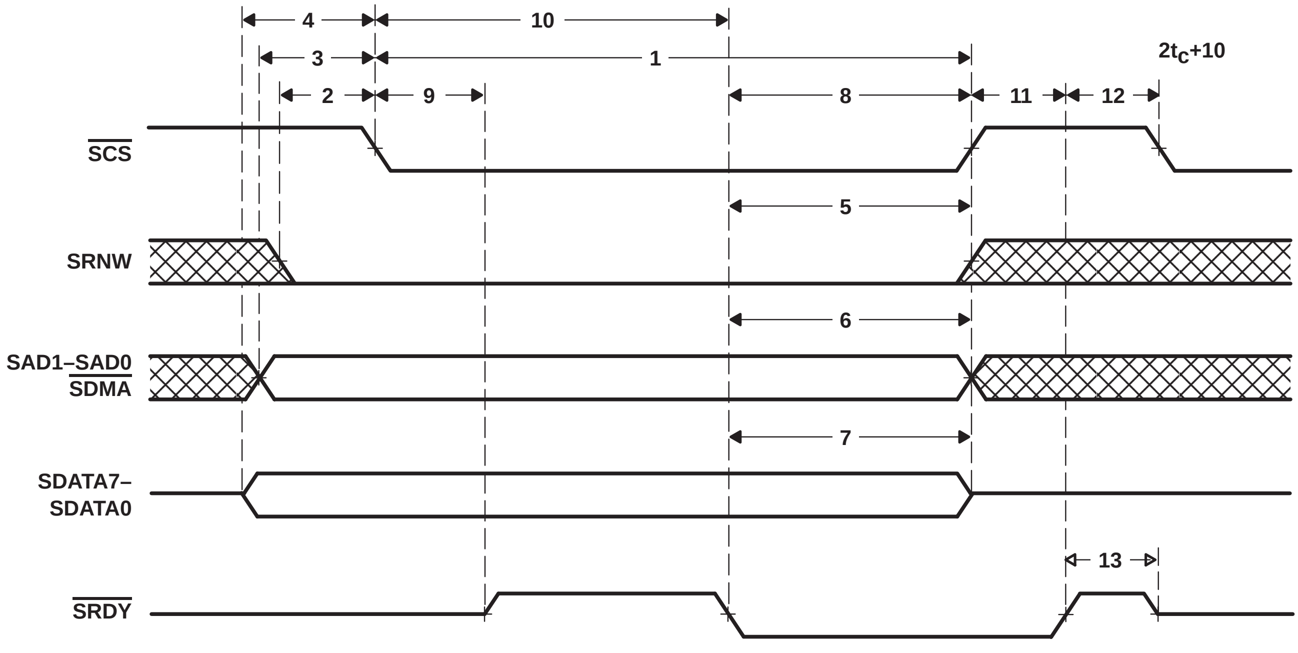


Figure 15. DIO and DMA Writes

DIO and DMA reads (see Figure 16)

NO.			MIN	MAX	UNIT
1	$t_{w(SCSL)}$	Pulse duration, $\overline{\text{SCS}}$ low	$2t_c$		ns
2	$t_{su(SRNW)}$	Setup time, SRNW valid before $\overline{\text{SCS}}\downarrow$	0		ns
3	$t_{su(SAD)}$	Setup time, SAD1–SAD0, $\overline{\text{SDMA}}$ valid before $\overline{\text{SCS}}\downarrow$	0		ns
4	$t_{h(SRNW)}$	Hold time, SRNW low after $\overline{\text{SRDY}}\downarrow$	0		ns
5	$t_{h(SAD)}$	Hold time, SAD1–SAD0, $\overline{\text{SDMA}}$ valid after $\overline{\text{SRDY}}\downarrow$	0		ns
6	$t_{h(SCSL)}$	Hold time, $\overline{\text{SCS}}$ low after $\overline{\text{SRDY}}\downarrow$	0		ns
7	$t_{su(SDATAD)}$	Setup time from $\overline{\text{SRDY}}\downarrow$ to SDATA7–SDATA0 driven	0		ns
8	$t_{d(SRDYZH)}$	Delay time from $\overline{\text{SCS}}\downarrow$ to $\overline{\text{SRDY}}\uparrow$		10	ns
9	$t_{d(SRDYHL)}$	Delay time from $\overline{\text{SCS}}\downarrow$ to $\overline{\text{SRDY}}\downarrow$	0	†	ns
10	$t_{d(SDATAZ)}$	Delay time from $\overline{\text{SCS}}\uparrow$ to SDATA7–SDATA0 3-state	0	10	ns
11	$t_{d(SRDYLH)}$	Delay time from $\overline{\text{SCS}}\uparrow$ to $\overline{\text{SRDY}}\uparrow$	t_c	$2t_c$	ns
12	$t_{h(SCSH)}$	Hold time, $\overline{\text{SCS}}$ high after $\overline{\text{SRDY}}\uparrow$	0		ns
13	$t_{w(SRDY)}$	Pulse duration, $\overline{\text{SRDY}}$ high		t_c	ns

† When the switch is performing certain internal operations (e.g., EEPROM load), there is a delay of up to 20 ms (24C02) or 800 ms (24C08) between $\overline{\text{SCS}}$ being asserted and $\overline{\text{SRDY}}$ being asserted.

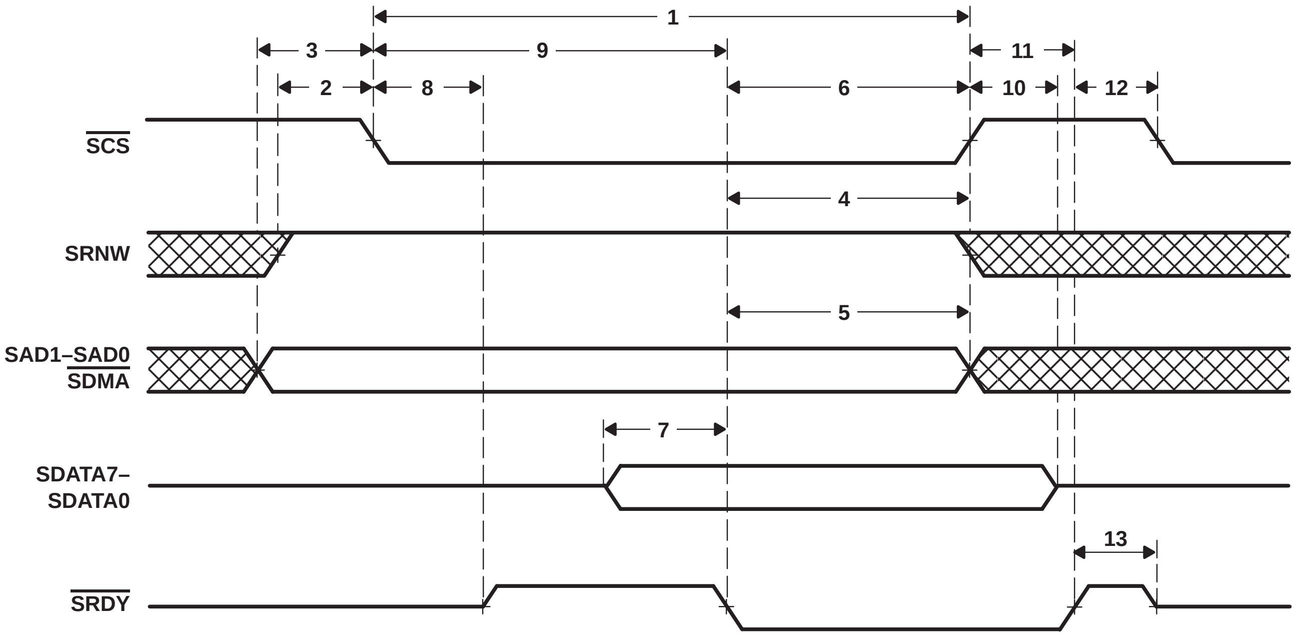


Figure 16. DIO and DMA Reads

EEPROM interface

For further information on EEPROM interface timing, refer to the 24C02 or 24C08 serial EEPROM data sheets.

EEPROM writes (see Figure 17)

NO.			MIN	MAX	UNIT
1	$t_{su(EDIO:Start)}$	Setup time, ECLK high before EDIO↓	383		t_c
2	$t_{h(EDIO:Start)}$	Hold time, ECLK high after EDIO↓	383		t_c
3	$t_{h(EDIO:Data)}$	Hold time, data after ECLK↓	0		t_c
4	$t_{su(EDIO:Data)}$	Setup time, data before ECLK↑	383		t_c
5	$t_{w(ECLK)}$	Pulse duration, ECLK low during start/stop	766		t_c
6	$t_{w(ECLK)}$	Pulse duration, ECLK high during start/stop	766		t_c
7	$t_{w(ECLK:Data)}$	Pulse duration, ECLK high during data	383		t_c
8	$t_{w(ECLK:Data)}$	Pulse duration, ECLK low during data	766		t_c
9	$f_{clock(ECLK)}$	Clock frequency, ECLK		98	kHz

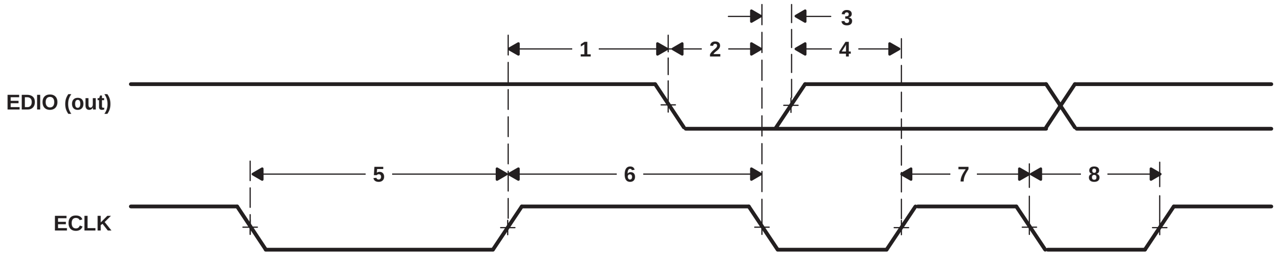


Figure 17. EEPROM Writes

EEPROM reads (see Figure 18)

NO.			MIN	MAX	UNIT
1	$t_{su(EDIO)}$	Setup time, EDIO (in) before ECLK↑	10		ns
2	$t_{h(EDIO)}$	Hold time, EDIO (in) after ECLK↓	0		ns

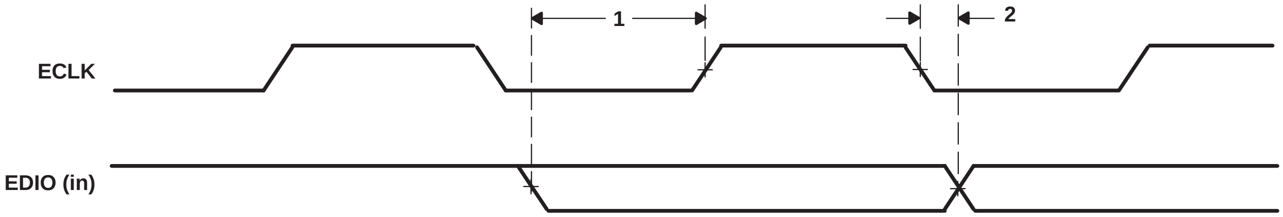


Figure 18. EEPROM Reads

PRODUCT PREVIEW

LED interface

LED (see Figure 19)

NO.			MIN	MAX	UNIT
1	$t_{c(LED\_CLK)}$	Cycle time, LED_CLK		8	t_c
2	$t_{w(LED\_CLK)}$	Pulse duration, LED_CLK high		4	t_c
3	$t_{n(LED\_CLK)}$	Number of LED_CLK pulses in burst		24†	
4	$t_{c(BURST)}$	Cycle time, LED_CLK burst		4687488‡	t_c
5	$t_{su(\overline{LED\_DATA})}$	Setup time, $\overline{\text{LED\_DATA}}$ before LED_CLK↑		4	t_c
6	$t_{h(\overline{LED\_DATA})}$	Hold time, $\overline{\text{LED\_DATA}}$ after LED_CLK↑		4	t_c

† During hard reset, LED_CLK runs continuously.
‡ Does not apply during hard reset

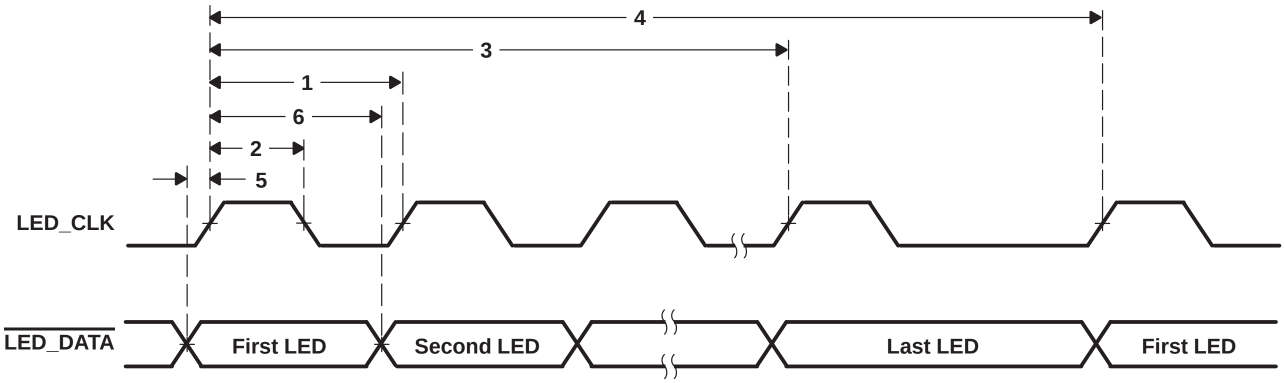


Figure 19. LED

PARAMETER MEASUREMENT INFORMATION

The following load circuits and voltage waveforms show the conditions used for measuring switching characteristics. Test points are illustrated schematically on the load circuits. Reference points are plotted on the voltage waveforms.

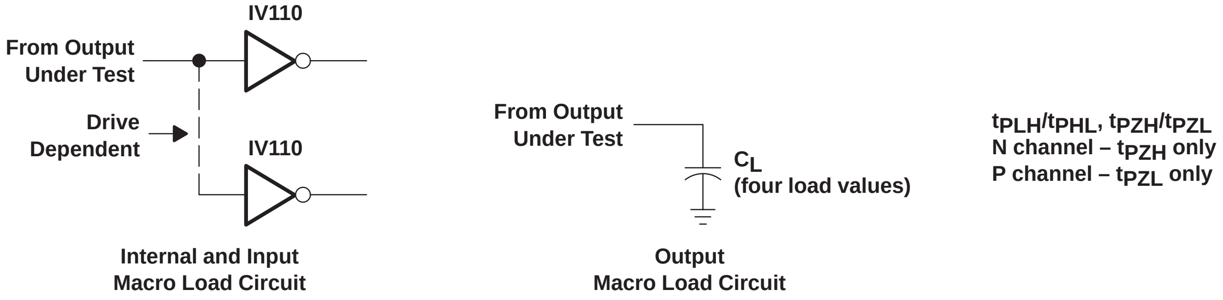

IV110
From Output Under Test
Drive Dependent
IV110
Internal and Input Macro Load Circuit
From Output Under Test
C_L (four load values)
Output Macro Load Circuit
t_{PLH}/t_{PHL}, t_{PZH}/t_{PZL}
N channel – t_{PZH} only
P channel – t_{PZL} only

Figure 20. Loading for Active Transitions

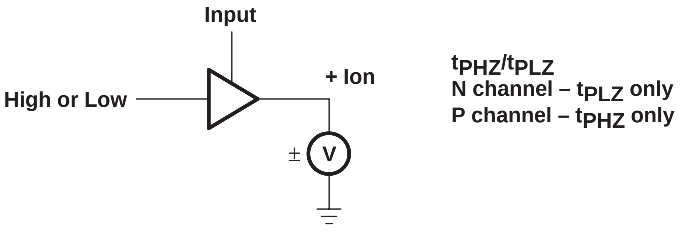


Figure 21. Loading for High-Impedance Transitions

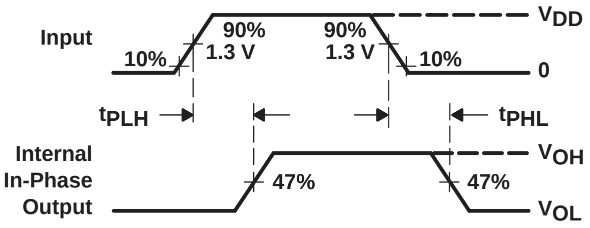


Figure 22. TTL Input Macro Propagation-Delay-Time Voltage Waveforms

PRODUCT PREVIEW

PRODUCT PREVIEW

PARAMETER MEASUREMENT INFORMATION

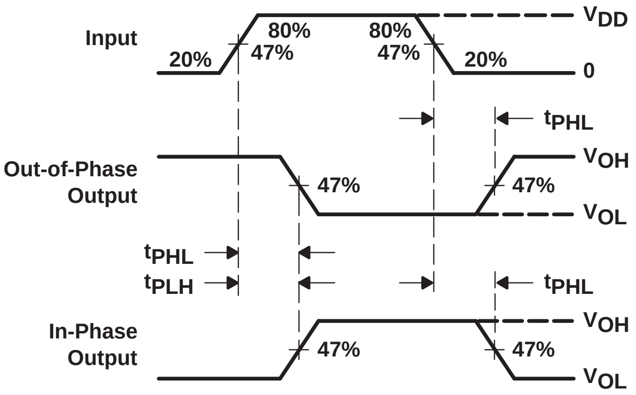


Figure 23. Internal Push/Pull Output Propagation-Delay-Time Voltage Waveforms

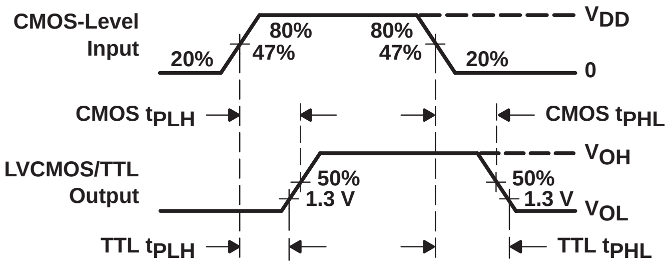


Figure 24. TTL Output Macro Propagation-Delay-Time Voltage Waveforms

PARAMETER MEASUREMENT INFORMATION

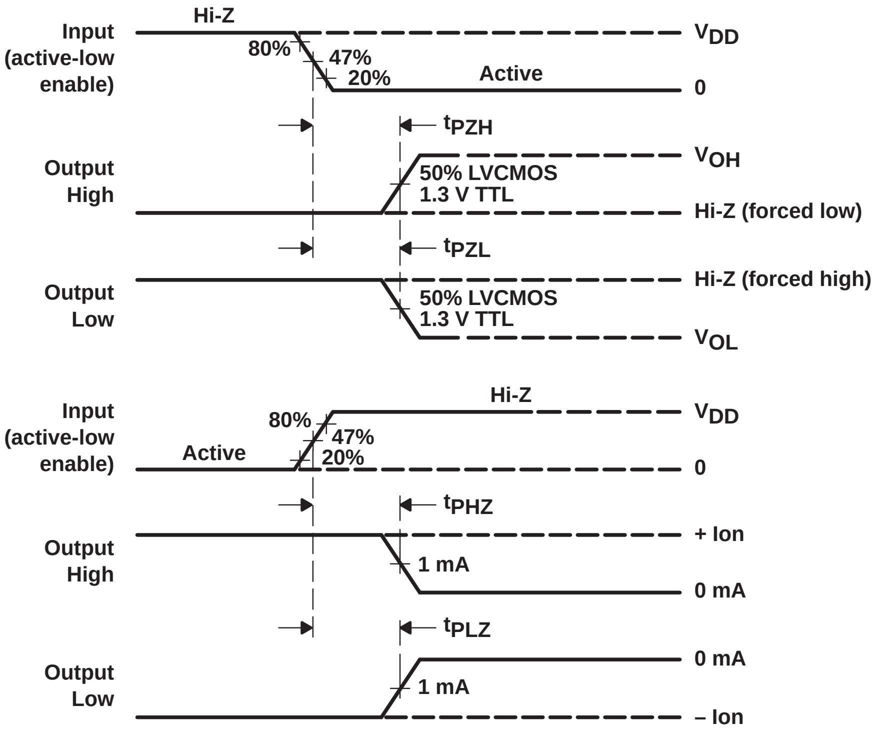
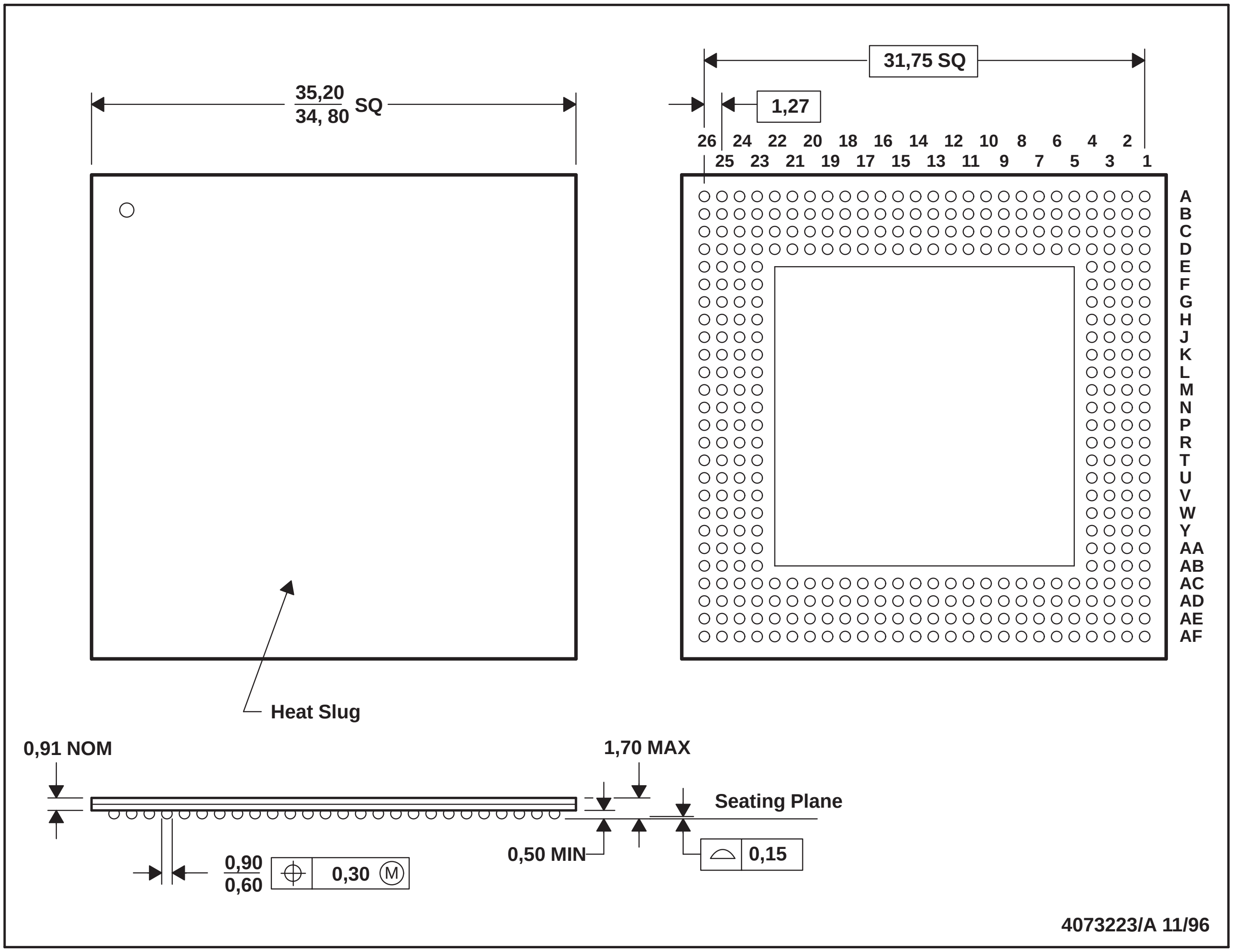


Figure 25. TTL 3-State Output Disable and Enable Voltage Waveforms

PRODUCT PREVIEW

MECHANICAL DATA

GGP (S-PBGA-N352) **PLASTIC BALL GRID ARRAY (CAVITY DOWN) PACKAGE**



NOTES: A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Thermally enhanced die down plastic package with top surface metal heat slug.

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.