Programming a Base Configuration for the ThunderSWITCH II (TNETX4090/4080)

APPLICATION REPORT: SPWA008A

Worldwide Broadband Access Group
LAN Products
December 1998

TEXAS INSTRUMENTS

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.

TRADEMARKS

TI and ThunderSWITCH are trademarks of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

PIC Telephone	(972) 644-5580
PIC FAX	(972) 480-7800
Email	networks@ti.com

Contents

Abstract **7**
Introduction **8**
ThunderSWITCH-to-EEPROM Interface **9**
Example 24C02 EEPROM Configurations for the TNETX4090 11
Example 24C02 EEPROM Configurations for the TNETX4080 13
Example 24C08 EEPROM Configurations for the TNETX4090 15
Example 24C08 EEPROM Configurations for the TNETX4080 19
Conclusion **23**
Appendix A. EEPROM Protocol and Handshaking Sequences **24**
Valid Data 24
Start and Stop States 25
Data Transfer Acknowledge 25
EEPROM Read States 26
EEPROM Write States 28
Appendix B. CRC Subroutine **31**
Appendix C. Related Documentation **33**
World Wide Web Support 33
Email Support 33

Figures

Figure 1. ThunderSWITCH-to-EEPROM Interface 10
Figure 2. EEPROM Valid Data Timing Diagram 24
Figure 3. Start-up and Stop Bits Timing Diagram 25
Figure 4. EEPROM Acknowledge Response Timing Diagram 25
Figure 5. EEPROM Data Read Timing Diagram 26
Figure 6. EEPROM Data Write Timing Diagram 28

Tables

Table 1. ThunderSWITCH-to-EEPROM/DIO Mapping 9
Table 2. TNETX4090 EEPROM Memory Map (24C02) 12
Table 3. TNETX4080 EEPROM Memory Map (24C02) 14
Table 4. TNETX4090 EEPROM Memory Map (24C08) 16
Table 5. TNETX4080 EEPROM Memory Map (24C08) 20
Table 6. TNETX4090/4080 Related Documentation 33

Examples

Example 1. Code for EEPROM Data Read Sequence 26
Example 2. Code for EEPROM Data Write Sequence 28
Example 3. Software Listing to Calculate CRC 31

Programming a Base Configuration for the ThunderSWITCH II (TNETX4090/4080)

Abstract

This application report describes how to program Texas Instruments' (TI™) ThunderSWITCH II™ (TNETX4090/4080) also referred to in this document as the ThunderSWITCH device. This report includes a description of the Electrically Erasable Programmable Read Only Memory (EEPROM) interface and handshaking protocol, configuration examples, description of the built-in data security protection, and product support information. Also included are CRC and EEPROM data read and write source code listings along with a CRC calculation program.

Introduction

The EEPROM interface is provided so the system level manufacturer can produce an unmanaged pre-configured system to their customers. Customers may also wish to change or reconfigure their system and retain their preferences. The EEPROM contains configuration, initialization, and CRC (Cyclic Redundancy Check) information accessed infrequently, typically at power up and after a reset.

EEPROMs sometimes corrupt the configuration and initialization information. Corruption is caused by erroneous or false data writes. Noise spikes, power supply glitches, or bus contention problems may cause a false write or erase cycle. Once the EEPROM data is corrupted, it cannot be cleared by simply removing the power; the data is permanently corrupted and the false data must be detected.

Texas Instruments provides a detection mechanism. If the ThunderSWITCH device detects the presence of an EEPROM, the contents are read and a CRC calculation is performed on the EEPROM data. The calculated CRC value is then compared to a previously written CRC (within the EEPROM). If the CRC values match, the ThunderSWITCH device is properly configured and processing begins. If the CRC values do not match, the device remains in a non-operational state and the FAULT LED is turned on. For EEPROM based system, a valid CRC is always required; any time new data is written to the EEPROM, a new CRC value must be calculated (via software) and saved within the EEPROM. See Appendix B for the CRC code listing.

The ThunderSWITCH device contains the logic to perform a CRC. The EEPROM CRC algorithm is based on the IEEE802.3u specification for frame check sequence (FCS) field. The following description of the CRC algorithm is adapted from the IEEE802.3u specification.

In the TNETX4090/4080 the four-byte CRC value is stored in addresses 0xFC-0xFF. The encoding is defined by the following generating polynomial:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

ThunderSWITCH-to-EEPROM Interface

Either the 24C02 or 24C08 serial EEPROM device is used in a CPU-less environment. Both EEPROMs use a two wire serial interface for communication and are available in a small footprint package.

The 24C02 provides 2048 bits organized as 256 x 8. The 24C02 provides port and system configuration information to the ThunderSWITCH device (see Table 1 for details).

The 24C08 provides 8192 bits organized as 1024 x 8. The 24C08 provides port, system, and VLAN configuration information to the ThunderSWITCH device (see Table 1 for details).

Table 1. ThunderSWITCH-to-EEPROM/DIO Mapping

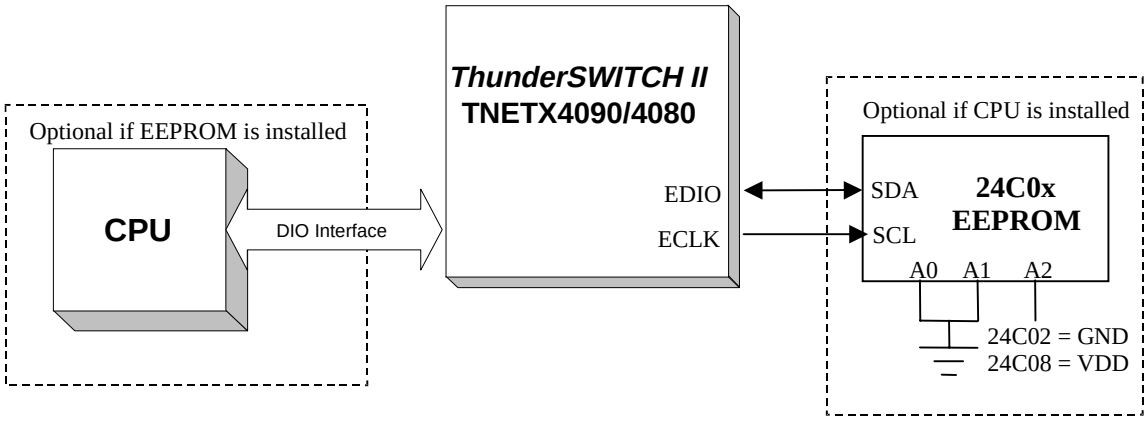
REGISTERS	**Loadable Using 24C02 EEPROM**	**Loadable Using 24C08 EEPROM**	**DIO Address Range (HEX)**
Port Control Registers	Yes	Yes	0x0000-0x003F
Address lookup configuration registers	Yes	Yes	0x0040-0x009F
System registers	Yes	Yes	0x00A0-0x00FF
VLAN registers	No	Yes	0x0100-0x03FF
Port Status registers	No	No	0x0400-0x043F
Address lookup registers	No	No	0x0440-0x05FF
PCS registers	No	No	0x0600-0x07FF
System registers	No	No	0x0800-0x0FFF
Manufacturing test registers	No	No	0x1000-0x11FF

The EEPROM size is detected automatically according to the address assigned to the EEPROM (shown in Figure 4):

A 24C02 EEPROM should have its **A0**, **A1**, and **A2** pins tied low.

A 24C08 EEPROM should have **A0** and **A1** pins tied low and **A2** pin tied high.

Figure 1. ThunderSWITCH-to-EEPROM Interface



After the initial start condition, a slave address containing a device address of 000 is output on **EDIO**. Then **EDIO** is observed for an acknowledge from the EEPROM. If one is received, operation will continue for the 24C02 EEPROM. If none is received, another start condition will be issued, this time with device address 101. If this too receives no acknowledge, it is deduced that no EEPROM is present and device operation continues using the current register settings (that is, those following a hardware reset or those previously entered by software). When this device is driving **EDIO** it will only drive out a strong logical-0. When a logical-1 is intended to be driven out the pin must be resistively pulled high; an on-chip 50uA current-source pull-up 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.

Once a 24C02 EEPROM is detected, (after power-up or after a reset) the ThunderSWITCH device reads the contents of the EEPROM (in ascending order from 0x00 to 0xFF) and initializes DIO registers 0x000 – 0x0FB. As the TNETX4090/4080 reads the EEPROM data, it calculates a CRC; if the calculated CRC value matches the value stored at EEPROM addresses 0xFC – 0xFF, theTNETX4090/4080 begins operation.

If a 24C08 EEPROM is detected (after power-up or after a reset), the ThunderSWITCH device reads the contents of the EEPROM (in ascending order from 0x100 to 0x3FF, followed by address 0x00 continuing through to 0xFF) and initializes DIO registers 0x000 – 0x3FF. As the ThunderSWITCH device reads the EEPROM data, it calculates a CRC; if the calculated CRC value matches the value stored at EEPROM addresses 0xFC – 0xFF, the ThunderSWITCH device begins operation.

ThunderSWITCH II TNETX4090/4080 will have a Device ID (0x03). The EEPROM Device ID serves two purposes:

- It identifies the EEPROM for the given device.
- It uniquely identifies the ThunderSWITCH device.

Example 24C02 EEPROM Configurations for the TNETX4090

Table 2 shows an example of the 24C02 EEPROM configuration. Notice that all values are in hexadecimal notation. The following list briefly describes the EEPROM settings (for more detail, refer to the TNETX4090 data sheet or programmers guide listed in Appendix C, *Related Documentation*):

- Status – All ports are enabled.
- Speed – All 100Mbps ports are configured to support autonegotiation.
- Port 8 is configured for Gigabit in GMII mode. Ring Topology is enabled and the ring ID = 0x3. Note all TNETX4090 devices in a Ring Topology must have a unique ring ID (for example if there were 3 TNETX4090 devices connected in Ring Mode, the other 2 devices could have switch ID = 0x1 and 0x2).
- Transmit Pacing – This feature is disabled and should be disabled under normal operating conditions.
- Duplex – All 100Mbps ports (ports 0-7) are configured to support autonegotiation. Port 8 is forced into full-duplex mode.
- TX/RX Access – All 100Mbps ports (ports 0-7) are access ports. Port 9 is configured as a non-access port.
- Maxlength – All 100Mbps ports (ports 0-7) can receive a maximum frame size of 1518 bytes. Hence the **maxlen** bit is set to 1.
- Pause – All 100Mbps ports (ports 0-7) are configured to support autonegotiation of pause based flow control; all ports support half duplex (collision based) flow control.
- Re-negotiation – For normal operation the **neg** bit within the EEPROM must be a 0. Since the auto-engotiation must be performed by external hardware or CPU.
- Port 09 (Network Management port) is the only port included in the mask for the uplink routing register. Hence all mirrored frames will be sent to the NM Port.
- Port Trunking is not enabled.

- Threshold aging is performed and the age time is set for 80 seconds.
- PauseTime100 & 1000 values are 0x010 which is equivalent to 8K bit times
- Device Node = 0x080028000001. The Device Node is the source address for all pause frames transmitted by this device. (The Device Node shown above is only an example and must be changed for a production platform.)
- The following Device IDs are used so that software can uniquely identify the corresponding ThunderSWITCH device:
  - TNETX4090 Device ID = 03
- New addresses are added to the lookup table without regard for the validity of the CRC; this option can be changed if the user wants to only add addresses from packets that contain a valid CRC.
- 2 Megabytes of memory is used.
- Flow control threshold = 324 buffers. This limit allows each port to continue receiving a maximum length frame.

Table 2. TNETX4090 EEPROM Memory Map (24C02)

BASE	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000:	30	1C	30	1C	30	1C	30	1C	30	1C	30	1C	30	1C	30	1C
0010:	00	0C	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0020:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0030:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0040:	00	00	00	00	0A	00	00	00	00	00	00	00	00	00	00	00
0050:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0060:	FF	01	00	00	FF	01	00	00	00	00	00	00	00	00	00	00
0070:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0080:	FF	FF	FF	FF	FF	FF	FF	FF	00	00	00	00	**11**	00	00	00
0090:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00A0:	00	80	00	**03**	00	00	00	00	00	00	00	00	00	00	00	00
00B0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00C0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00D0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00E0:	00	00	14	02	00	00	00	00	10	00	10	00	00	00	10	00
00F0:	28	00	00	00	00	40	00	00	CF	00	03	20	**90**	**AD**	**05**	**BE**

Notes: 1) CRC values are in **bold text** @ 0xFC – 0xFF.
2) Device ID is in **bold text** @ address A3
3) Ring Topology enable and ring ID are in **bold text** @ 8C.

Example 24C02 EEPROM Configurations for the TNETX4080

Table 3 shows an example of the 24C02 EEPROM configuration. Notice that all values are in hexadecimal notation. The following list briefly describes the EEPROM settings (for more detail, refer to the TNETX4080 data sheet or programmers guide listed in Appendix C, *Related Documentation*):

- Status – All 10/100Mbps ports are enabled.
- Speed – All 10/100Mbps ports are configured to support autonegotiation.
- There is no Port 8 on this device – **always disable this port on the TNETX4080.**
- Transmit Pacing – This feature is disabled and should be disabled under normal operating conditions.
- Duplex – All 10/100Mbps ports (ports 0-7) are configured to support autonegotiation.
- TX/RX Access – All 10/100Mbps ports (ports 0-7) are access ports.
- Maxlength – All 10/100Mbps ports (ports 0-7) can receive a maximum frame size of 1518 bytes. Hence the **maxlen** bit is set to 1.
- Pause – All 10/100Mbps ports (ports 0-7) are configured to support autonegotiation of pause based flow control; all ports support half duplex (collision based) flow control.
- Re-negotiation – For normal operation the **neg** bit within the EEPROM must be a 0. Since the auto-engotiation must be performed by external hardware or CPU.
- Port 09 (Network Management port) is the only port included in the mask for the uplink routing register. Hence all mirrored frames will be sent to the NM Port.
- Port Trunking is not enabled.
- Time Threshold aging is performed and the age time is set for 80 seconds.
- PauseTime100 & 1000 values are 0x010 which is equivalent to 8K bit times
- Device Node = 0x080028000001. The Device Node is the source address for all pause frames transmitted by this device.

(The Device Node shown above is only an example and must be changed for a production platform.)

- The following Device IDs are used so that software can uniquely identify the corresponding ThunderSWITCH device:
  - TNETX4080 Silicon Device ID = 03
  - The Device ID programmed into the EEPROM to identify the TNETX4080 correctly is 0x09. This allows the TSMAN software to correctly identify the TNETX4080 device.
- New addresses are added to the lookup table without regard for the validity of the CRC; this option can be changed if the user wants to only add addresses from packets that contain a valid CRC.
- 2 Megabytes of memory is used.
- Flow control threshold = 324 buffers. This limit allows each port to continue receiving a maximum length frame.

Table 3. TNETX4080 EEPROM Memory Map (24C02)

BASE	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000:	30	1C	30	1C	30	1C	30	1C	30	1C	30	1C	30	1C	30	1C
0010:	80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0020:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0030:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0040:	00	00	00	00	50	00	00	00	00	00	00	00	00	00	00	00
0050:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0060:	FF	01	00	00	FF	01	00	00	00	00	00	00	00	00	00	00
0070:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0080:	FF	FF	FF	FF	FF	FF	FF	FF	00	00	00	00	00	00	00	00
0090:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00A0:	00	80	00	**09**	00	00	00	00	00	00	00	00	00	00	00	00
00B0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00C0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00D0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00E0:	00	00	14	02	00	00	00	00	10	00	10	00	00	00	10	00
00F0:	28	00	00	00	00	40	00	00	CF	00	03	20	**C0**	**7E**	**ED**	**55**

Notes: 1) CRC values are in **bold text** @ 0xFC – 0xFF.
2) Device ID is in **bold text** @ address A3

Example 24C08 EEPROM Configurations for the TNETX4090

Table 4 lists an example of the 24C08 EEPROM configuration. The following list briefly describes the EEPROM settings (for more detail, see the appropriate data sheet or programmers guide listed in Appendix A, *Related Documentation*):

- Status
- All ports are enabled.
- Speed
- Ports 0-7 are configured to support autonegotiation. Port 8 is configured in Gigabit GMII mode.
- Transmit Pacing
- This feature is disabled and should be disabled under normal operating conditions.
- Duplex
- Ports 0-7 are configured to support autonegotiation. Port 8 is forced into full-duplex mode to provide maximum bandwidth on the cascade ring.
- All ports are configured as non-access ports (can accept VLAN tagged frames).
- VLAN configuration (example shown for TNETX4090 only). Port 9 (network management port) is a member of all VLANs, Port 8 (gigabit port) is also a member of all VLANs (since in ring mode it has to forward frames to ports on other TNETX4090 devices), and its default VID (Port8QTAG) = FF.
  - Ports 0 – 2 & 8,9 are assigned to VLAN3 (0x3)
  - Ports 3 – 5 & 8,9 are assigned to VLAN71 (0x47)
  - Ports 6 – 7 & 8,9 are assigned to VLAN33 (0x21)
  - Default VID for Ports 0-2 = 0x4
  - Default VID for Ports 3-5 = 0x5
  - Default VID for Ports 6-7 = 0x6

  The Default VID make sure that un-tagged frames that are received are only flooded to the ports in the same VLAN.
- Maxlength

  Since all ports are configured as non-access ports they can all receive a maximum frame size of 1535 bytes.

- Pause

  Ports 0-7 are configured to support autonegotiation of pause based flow control; all ports support half duplex (collision based) flow control. Port 8 will use the hardware flow control on the cascade ring.

- Re-negotiation – For normal operation the **neg** bit within the EEPROM must be a 0. Since the auto-engotiation must be performed by external hardware or CPU.

- Port 09 (Network Management port) is the only port included in the mask for the uplink routing register. Hence all mirrored frames will be sent to the NM Port.

- Port Trunking is not enabled.

- Threshold aging is performed and the age time is set for 80 seconds.

- PauseTime100 & 1000 values are 0x010, which is equivalent to 8K bit times

- Device Node = 0x080028000001.

  The Device Node is the source address for all pause frames transmitted by this device. (The Device Node shown above is only an example and must be changed for a production platform.)

- The following Device IDs are used so that software can identify the unique ThunderSWITCH device:

  TNETX4090 Device ID = 03

- New addresses are added to the lookup table without regard for the validity of the CRC; this option can be changed if the user wants to only add addresses from packets that contain a valid CRC.

- 2 Megabyte of memory is used.

- Flow control threshold = 324 buffers. This limit allows each port to continue receiving a maximum length frame.

Table 4. TNETX4090 EEPROM Memory Map (24C08)

```
Base  +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f
0000: 30 0C 30 0C 30 0C 30 0C 30 0C 30 0C 30 0C 30 0C
0010: 48 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 0A 00 00 00 00 00 00 00 00 00 00 00
```

```
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: FF 01 00 00 FF 01 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: FF FF FF FF FF FF FF FF 00 00 00 00 11 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 80 00 03 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 14 02 00 00 00 00 10 00 10 00 00 00 10 00
00F0: 28 00 00 00 40 00 00 00 CF 00 03 20 7C D3 D7 CC
0100: 07 03 00 00 38 03 00 00 B0 03 00 00 07 03 00 00
0110: 38 03 00 00 B0 03 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0300: 03 00 47 00 21 00 04 00 05 00 06 00 00 00 00 00
```

```
0310: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0320: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0370: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0380: 04 00 04 00 04 00 05 00 05 00 05 00 06 00 06 00
0390: FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Notes: 1) CRC values are in **bold text** @ 0xFC – 0xFF.
2) Device ID is in **bold text** @ address A3
3) Ring Topology enable and ring ID are in **bold text** @ 8C.

Example 24C08 EEPROM Configurations for the TNETX4080

Table 5 lists an example of the 24C08 EEPROM configuration. The following list briefly describes the EEPROM settings (for more detail, see the appropriate data sheet or programmers guide listed in Appendix A, *Related Documentation*):

- Status – All 10/100Mbps ports are enabled.
- Speed – All 10/100Mbps ports are configured to support autonegotiation.
- There is no Port 8 on this device – **always disable this port on the TNETX4080.**
- Transmit Pacing – This feature is disabled and should be disabled under normal operating conditions.
- Duplex – All 10/100Mbps ports (ports 0-7) are configured to support autonegotiation.
- All ports are configured as non-access ports (can accept VLAN tagged frames).
- VLAN configuration (example shown for TNETX4080 only). Port 9 (network management port) is a member of all VLANs,
  - Ports 0 – 2 & 9 are assigned to VLAN3 (0x3)
  - Ports 3 – 5 & 9 are assigned to VLAN71 (0x47)
  - Ports 6 – 7 & 9 are assigned to VLAN33 (0x21)
  - Default VID for Ports 0-2 = 0x4
  - Default VID for Ports 3-5 = 0x5
  - Default VID for Ports 6-7 = 0x6
- The Default VID make sure that un-tagged frames that are received are only flooded to the ports in the same VLAN.
- Maxlength

  Since all ports are configured as non-access ports they can all receive a maximum frame size of 1535 bytes.
- Pause

  Ports 0-7 are configured to support autonegotiation of pause based flow control; all ports support half duplex (collision based) flow control. Port 8 will use the hardware flow control on the cascade ring.

- Re-negotiation – For normal operation the **neg** bit within the EEPROM must be a 0. Since the auto-engotiation must be performed by external hardware or CPU.
- Port 09 (Network Management port) is the only port included in the mask for the uplink routing register. Hence all mirrored frames will be sent to the NM Port.
- Port Trunking is not enabled.
- Time Threshold aging is performed and the age time is set for 80 seconds.
- PauseTime100 & 1000 values are 0x010 which is equivalent to 8K bit times
- Device Node = 0x080028000001. The Device Node is the source address for all pause frames transmitted by this device. (The Device Node shown above is only an example and must be changed for a production platform.)
- The following Device IDs are used so that software can uniquely identify the corresponding ThunderSWITCH device:
  - TNETX4080 Silicon Device ID =03
  - The Device ID programmed into the EEPROM to identify the TNETX4080 correctly is 0x09. This allows the TSMAN software to correctly identify the TNETX4080 device.
- New addresses are added to the lookup table without regard for the validity of the CRC; this option can be changed if the user wants to only add addresses from packets that contain a valid CRC.
- 2 Megabytes of memory is used.
- Flow control threshold = 324 buffers. This limit allows each port to continue receiving a maximum length frame.

Table 5. TNETX4080 EEPROM Memory Map (24C08)

```
Base  +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +a +b +c +d +e +f
0000: 30 00 30 00 30 00 30 00 30 00 30 00 30 00 30 00
0010: 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 50 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: FF 01 00 00 FF 01 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: FF FF FF FF FF FF FF FF 00 00 00 00 00 00 00 00
```

```
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 80 00 03 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 14 02 00 00 00 00 10 00 10 00 00 00 10 00
00F0: 28 00 00 00 40 00 00 00 CF 00 03 20 09 43 B3 65
0100: 07 02 00 00 38 02 00 00 B0 02 00 00 07 02 00 00
0110: 38 02 00 00 B0 02 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
01F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0300: 03 00 47 00 21 00 04 00 05 00 06 00 00 00 00 00
0310: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0320: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0370: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0380: 04 00 04 00 04 00 05 00 05 00 05 00 06 00 06 00
0390: FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
03F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Notes: 1) CRC values are in **bold text** @ 0xFC – 0xFF.
2) Device ID is in **bold text** @ address A3

Conclusion

For unmanaged systems, the TNETX4090/4080 EEPROM interface allows the designer to build a pre-configured system. Although EEPROMs are not error free, Texas Instruments provides an error detection mechanism on all ThunderSWITCH products. Remember any time new data is written to the EEPROM, a new CRC value must also be calculated (via software) and written to the EEPROM.

Appendix A. EEPROM Protocol and Handshaking Sequences

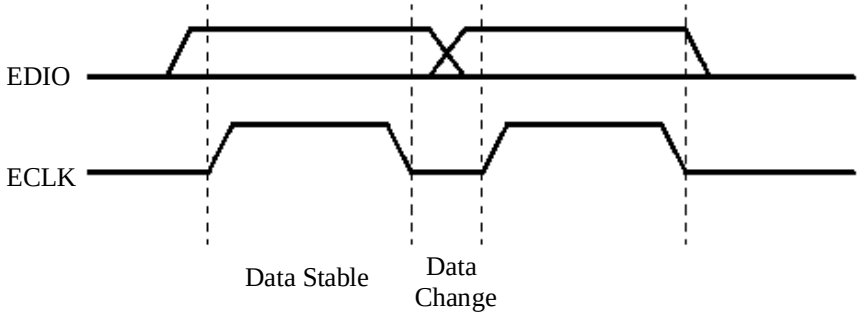
The basic protocol and handshaking sequences for reading and writing to the EEPROM via the DIO interface are shown in Figure 2 through Figure 6.

NOTE:
EDIO and ECLK pins are controlled by the ThunderSWITCH device EDATA, ETXEN, and ECLK Serial Interface I/O Register, SIO.

Valid Data

Valid data (EDIO) can change only when ECLK is low (see Figure 2). Changing the EDIO signal when ECLK is high indicates a start or stop condition.

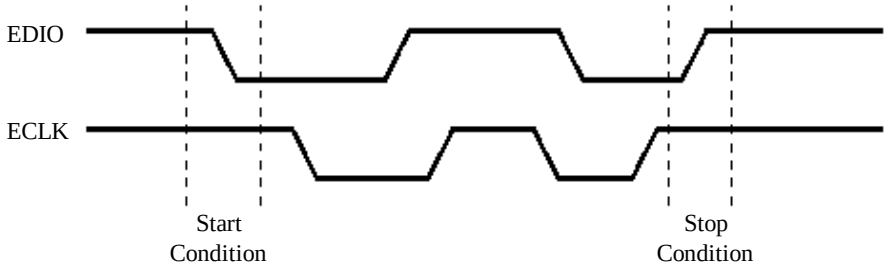
Figure 2. EEPROM Valid Data Timing Diagram



Start and Stop States

A high-to-low transition of EDIO while ECLK is high represents a start-up condition. This is followed by a 4-bit device code and a 3-bit device address (000b). A stop condition is indicated by a low-to-high transition of EDIO while ECLK is high; this is preceded by a data read or write. Figure 3 illustrates the start and stop conditions.

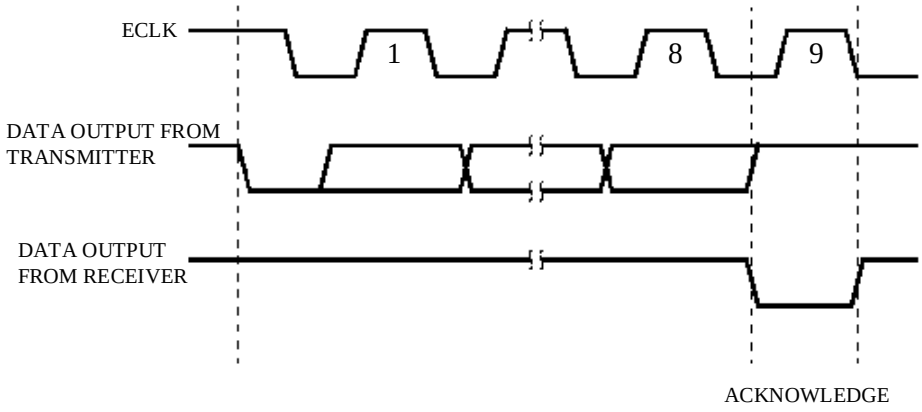
Figure 3. Start-up and Stop Bits Timing Diagram



Data Transfer Acknowledge

Another important handshake signal is the acknowledge signal, which indicates a successful data transfer. The TNETX4090/4080 releases the bus after transmitting eight bits. During the ninth clock cycle, the EEPROM pulls EDIO low. Figure 4 illustrates the acknowledge sequence.

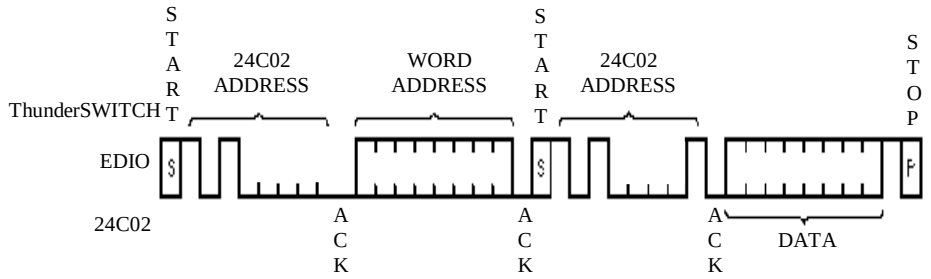
Figure 4. EEPROM Acknowledge Response Timing Diagram



EEPROM Read States

Figure 5 illustrates an EEPROM data read sequence. Example 1 shows the code providing this sequence.

Figure 5. EEPROM Data Read Timing Diagram



Example 1. Code for EEPROM Data Read Sequence

```
//---------------------------------------------------------------------
//  TS_EeRdByte() - read byte of data from eeprom (see 24C02
//                  device specification)
//
//  Parameters:
//    IN
//       addr           WORD  Address on EEPROM to read from
//       len            WORD  Number of bytes to read
//       pdata          *unsigned char  Data read
//
//  Return val:
//       success        int     1
//       failure        int     0
//---------------------------------------------------------------------
int TS_EeRdByte( WORD addr, WORD len, unsigned char *pdata )
{
    fnDioRd = &TS_DioRdByte;
    fnDioWr = &TS_DioWrByte;
    fnDioWrIm = &TS_DioWrByteIm;
    AddrSio = SIO_XCTRL;

    while( len-- )
        if( !My_EeRdByte( addr++, pdata++) )
            return(0);
    return(1);
}
//---------------------------------------------------------------------
//  My_EeRdByte() - read byte of data from eeprom (see Exel XL24C02
//                  device specification)
//
//  Parameters:
//    IN
//       addr           WORD  Address on EEPROM to read from
```

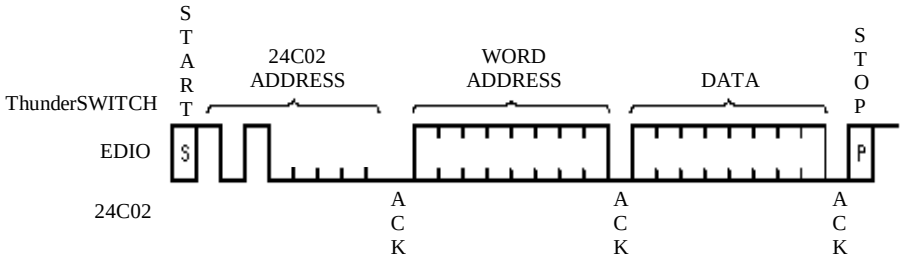
```
//       data           *BYTE Data read
//
//  Return val:
//    int               1 = read successful, 0 otherwise
//---------------------------------------------------------------------
static int My_EeRdByte( WORD addr, BYTE *data )
{
   BYTE  sio, i, temp = 0x00;

   (*fnDioRd)(AddrSio, 1, &sio);

   // send eeprom start sequence

   set(ECLOK)
   set(EDATA)
   set(ETXEN)
   clr(EDATA)
   clr(ECLOK)

   // select EEPROM device to use and set up to write address to read
   // from out to device
   TS_EeSel(WRITE);

   // if the EEPROM doesn't ACK, something's wrong
   if (!TS_EeAck())
      return (0);

   // Send address in EEPROM to read from
   TS_EeAddr(addr);

   // if the EEPROM doesn't ACK, something's wrong
   if (!TS_EeAck())
      return (0);

        // The Sel, Ack and Addr routines have changed what's in SIO_XCTRL
   (*fnDioRd)(AddrSio, 1, &sio);

   // send start access sequence for the read
   set(ETXEN)
   set(ECLOK)
   set(EDATA)
   set(ETXEN)
   clr(EDATA)
   clr(ECLOK)

   // send EEPROM device selection sequence and set up for read
   TS_EeSel(READ);

   if (!TS_EeAck())
      return(0);

   *data = 0;
   (*fnDioRd)(AddrSio, 1, &sio);

   for ( i = 0x80; i; i >>= 1 )
```

```
                                                                    {
      set(ECLOK)

      (*fnDioRd)(AddrSio, 1, &temp);
      if (temp & EDATA)
         *data |= i;

      clr(ECLOK)
   }

   tog(ECLOK)

   // send stop access sequence
   clr(EDATA)
   set(ECLOK)
   set(EDATA)
   clr(ETXEN)

   return(1);
}
```

EEPROM Write States

Figure 6 shows an EEPROM data write sequence. Example 2 shows the code providing this sequence.

Figure 6. EEPROM Data Write Timing Diagram



Example 2. Code for EEPROM Data Write Sequence

```
//---------------------------------------------------------------------
//  TS_EeWrByte() -  write a byte to the EEPROM
//
//  Parameters
//       addr           WORD   address to write to on EEPROM
//       len            WORD   number of bytes to write
//       data           *unsigned char  data to write
//
//  Return Value
//       success        int     1
//       failure        int     0
//---------------------------------------------------------------------
int TS_EeWrByte( WORD addr, WORD len, unsigned char *pdata )
{
    fnDioRd = &TS_DioRdByte;
```

```
    fnDioWr = &TS_DioWrByte;
    fnDioWrIm = &TS_DioWrByteIm;
    AddrSio = SIO_XCTRL;
    while( len-- )
        if( !My_EeWrByte( addr++, *pdata++ ) )
            return(0);
    return(1);
}

//---------------------------------------------------------------------
//  My_EeWrByte() -  write a byte to the EEPROM
//
//  Parameters
//     addr          WORD  address to write to on EEPROM
//     data          BYTE  data to write
//
//  Return Value
//     int           1 on success, 0 otherwise
//---------------------------------------------------------------------
static int My_EeWrByte( WORD addr, BYTE data)
{
   BYTE  sio, i;

   (*fnDioRd)(AddrSio, 1, &sio);

   // send eeprom start sequence

   set(ECLOK)
   set(EDATA)
   set(ETXEN)
   clr(EDATA)
   clr(ECLOK)

   // select EEPROM device to use and set up to write address to read
   // from out to device
   TS_EeSel(WRITE);

   // if the EEPROM doesn't ACK, something's wrong
   if (!TS_EeAck())
      return (0);

   // Send address in EEPROM to read from
   TS_EeAddr(addr);

   // if the EEPROM doesn't ACK, something's wrong
   if (!TS_EeAck())
      return (0);

                  // The Ack, Sel and Addr routines have changed SIO_XCTRL
   (*fnDioRd)(AddrSio, 1, &sio);
   set(ETXEN)

   for (i = 0x80; i; i >>= 1)
   {
      if ( i & data )
```

```
      set(EDATA)
      else
         clr(EDATA)

      tog(ECLOK)
   }

   if ( !TS_EeAck() )
      return (0);

   (*fnDioRd)(AddrSio, 1, &sio);

   // send stop access sequence
   set(ETXEN)
   clr(EDATA)
   set(ECLOK)
   set(EDATA)
   clr(ETXEN)

   // wait until eeprom writes the data
   do
   {
      (*fnDioRd)(AddrSio, 1, &sio);
// start sequence
      set(ECLOK)
      set(EDATA)
      set(ETXEN)
      clr(EDATA)
      clr(ECLOK)
      TS_EeSel(WRITE);
   } while ( !TS_EeAck() );
}
```

Appendix B. CRC Subroutine

The following subroutine calculates the CRC value that will be stored in the TNETX4090/4080 EEPROM. The CRC is calculated bit-wise, starting with the MSB and ending with the LSB. The most significant CRC byte should be placed at address 0xFC and the least significant CRC byte should be placed at address 0xFF. Once calculated, the CRC value is written to the EEPROM as shown in Example 3.

Example 3. Software Listing to Calculate CRC

```
//---------------------------------------------------------------------
//  CalcCRC - Calculate CRC for the specified range of data
//---------------------------------------------------------------------
//
//  Parameters:
//   start          IN  int   byte offset within buffer of where to start
//   nbytes         IN  int   number of bytes to compute
//   buf            IN  BYTE* pointer to data buffer
//   pcrc           OUT long* pointer where to put the CRC result
//
//  Return val:
//   void
//---------------------------------------------------------------------
void CalcCRC(
    int     start,
    int     nbytes,
    BYTE    *buf,
    long    *pcrc)
{
    long    crc, tmp = 0;
    int     i;

    crc  = 0xffffffffl;
    for (i = start;i < nbytes;i++)
    {
        crcbyt(buf[i],&crc);
    }
    crc ^= 0xffffffffl;
    tmp = (crc &  0x000000FF) << 24;  //0 -> 3
    tmp |= (crc & 0x0000FF00) << 8;   //1 -> 2
    tmp |= (crc & 0x00FF0000) >> 8;   //2 -> 1
    tmp |= (crc & 0xFF000000) >> 24;  //3 -> 0
    *pcrc = tmp;
}

static void crcbyt(int dat,long *crc)
{
    int i;

    for (i = 0;i < 8;i++)
    {
        crcbit(dat>>7,crc);
        dat = dat << 1;
    }
}

static void crcbit(int dat,long *crc)
```

```
{
    if ( (((*crc>>31) & 1l)^((long)dat & 1l)) ==1)
    {
        *crc ^= 0x02608edbl;
        *crc = *crc << 1;
        *crc |= 0x00000001l;
    }
    else
    {
        *crc = *crc << 1;
        *crc &= 0xfffffffel;
    }
}
```

Appendix C. Related Documentation

Table 6 lists specific product names, part numbers, and their documentation's literature number.

Table 6. TNETX4090/4080 Related Documentation

Product Name	Part Number	Literature Number
Desktop TNETX4090/4080 Reference Design Kit	TNETX4090/4080 EVM	N/A
TNETX4090/4080 Data Sheet	N/A	SPWS044C/SPWS050
Considerations Using the ThunderSWITCH Port Trunking and Load Sharing Features	N/A	SPWA016
TNETX4090 Programmers Guide	N/A	SPAU003

World Wide Web Support

Our World Wide Web site at **www.ti.com** contains the most up to date product information, revisions, and additions. New users must register with TI&ME before they can access the data sheet archive. TI&ME allows users to build custom information pages and receive automatic new product updates via email.

Email Support

For technical issues or clarification on switching products, please send a detailed email to **networks@ti.com**. Questions receive prompt attention and are usually answered within one business day.