Application Note

DP83822 Troubleshooting Guide

Avtar Dhaliwal

ABSTRACT

This application note provides a suggested flow for evaluating a new application and confirming the expected functionality. The step-by-step recommendations will help ease board bring up and initial evaluation of DP83822 designs.

Table of Contents

List of Figures

List of Tables

Trademarks

All trademarks are the property of their respective owners.

1 DP83822 Application Overview

Designed for harsh industrial environments, the DP83822 is an ultra-robust, low-power single-port 10/100 Mbps Ethernet PHY. The DP83822 provides all physical layer functions needed to transmit and receive data over standard twisted-pair cables, or connect to an external fiber optic transceiver. Additionally, the DP83822 provides flexibility to connect to a MAC through a standard MII, RMII, or RGMII interface.

Figure 1-1 is a high-level system block diagram of a typical DP83822 application.

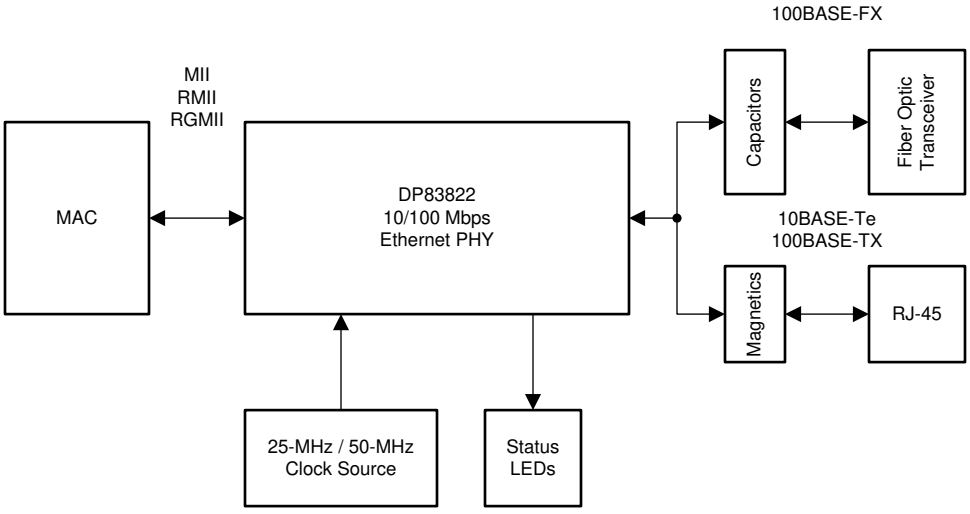

Figure 1-1. DP83822 Block Diagram

The DP83822 connects to an Ethernet MAC and to the MDI. The connection to the MDI is via a transformer and a connector (for copper applications) or capacitors and a transceiver (for fiber applications).

2 Troubleshooting the PHY Application

The following sections approach the debug from a high level, attempting to start with application characteristics that have a broad impact and then zeroing in on more focused aspects of the design.

2.1 Schematic and Layout Checklist

Reference and verify all of the noted schematic and layout recommendations in the DP83822 Product Page in the *Design and Development* section. A strap tool with all the available strap configurations as well as a pin-wise checklist is included in the schematic checklist spreadsheet. The checklist contains special cells with a dropdown menu that are shaded light orange, enabling you to select the desired strap configurations and provides the desired components and schematics in the pin-wise checklist.

2.2 Verify Successful Power-up of PHY

After verifying the schematics with the schematic checklist in the previous section, in order to start testing for any other issues that may present with application of the PHY, the PHY must first be successfully powered up in order to perform any other debugging processes. Probe the voltage rails of the PHY to ensure that the voltages are within limits defined in PHY Supply Voltage Specifications. Verify that the power up voltage parameter timings are within the limits defined in Timing Requirements, Power-Up Timing and Power-Up Timing.

Table 2-1. PHY Supply Voltage Specifications

	Descriptions	Min	Typ	Max	Unit
VDDIO	Supply Voltage 1/O = 1.8V	1.71	1.8	1.89	V
	Supply Voltage I/O = 2.5V	2.375	2.5	2.625	
	Supply Voltage I/O = 3.3V	3.15	3.3	3.45	
AVD	Supply Voltage Analog = 3.3V	3.15	3.3	3.45	V
	Supply Voltage Analog = 1.8V	1.71	1.8	1.89	
Center Tap (CT)	Supply Voltage Center Tap = 3.3V	3.15	3.3	3.45	V
	Supply Voltage Analog = 1.8V	1.71	1.8	1.89	

Table 2-2. Timing Requirements, Power-Up Timing

Parameter		Test Conditions	MIN	TYP	MAX	Unit
T1	AVD (analog supply) ramp delay post VDDIO (digital supply) ramp. AVD and VDDIO potential must not exceed 0.3 V prior to supply ramp.	Time from start of supply ramp	–100		100	ms
	VDDIO ramp time				100	ms
	AVD ramp time				100	ms
T2	Post power-up stabilization time prior to MDC preamble for register accesses. MDC preamble coming in any time after this max wait time will be valid.	MDIO is pulled high for 32-bit serial management initialization			200	ms
T3	Hardware configuration latch-in time for power up				200	ms
T4	Hardware configuration pins transition to output drivers			64		ns
T5	Fast Link Pulse transmission delay post power up			1.5		s

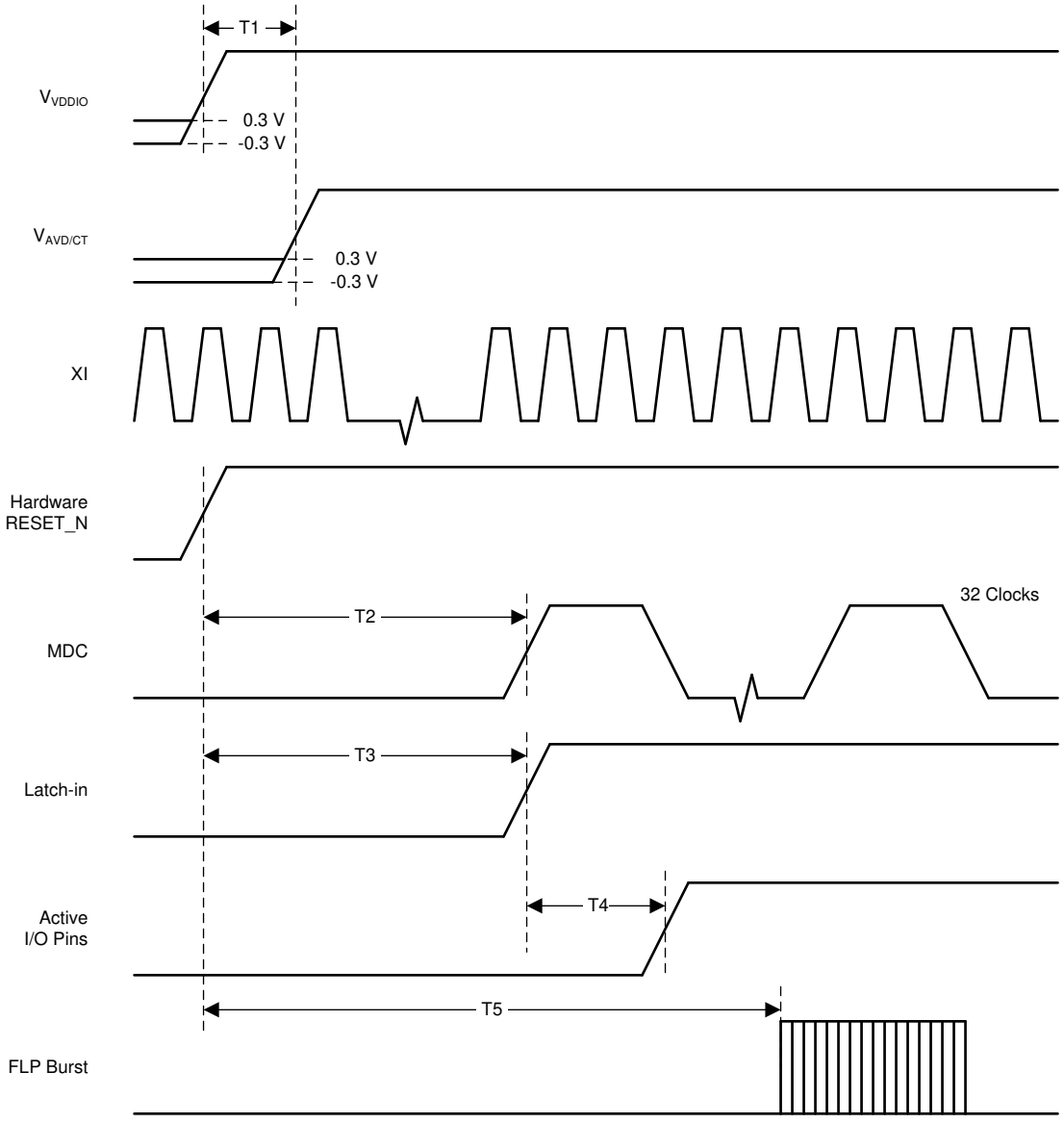

Figure 2-1. Power-Up Timing

Note

If a link up issue is present for DP83822 and the VDDA is operated in 3.3V, check register 0x0421 to see AVDD level and VDDIO level match the desire output. Register 0x0421 bit[2]=1 for 3.3V VDDA. If register 0x0421 does not match with the desire result, write 0x041F register to the desire voltage level. Write register 0x041F bit[12] = 1.

Registers 0x0421 and 0x041F are extended registers, make sure to follow Extended Register Access.

2.3 Read and Check Register Values

Read the registers and verify the default values shown in the device-specific data sheet. Note that the initial values of some registers can vary based on strap options.

Register Address	Register Value With Auto-Neg	
	10 Mbps	100 Mbps
0x0000	3100	3100
0x0001	786D	786D
0x0002	2000	2000
0x0003	A240	A240
0x0004	0061	C1E1
0x0005	C1E1	C1E1
0x0007	2001	2001
0x0009	0000	0000
0x000A	0100	0100
0x000B	1000	1000
0x000F	0000	0000
0x0010	4117	4715
0x0011	0108	0108
0x0012	0000	0000
0x0013	0000	0000
0x0014	0000	0000
0x0015	0000	0000
0x0016	0000	0000
0x0017	0041	0041
0x0018	0400	0400
0x0019	8021	8C21
0x001A	0000	0000
0X001B	007D	007D
0X001C	05EE	05EE
0x001E	0002	0102

With the PHY linked in a given speed, use these values as a reference to identify any variance from the expected operation. Note that not all registers need to be the same, for example .

- The value of Register 0x0005 depends on the link partner's capabilities.
- The '4' or '0' difference in the MSB of Register 0x0010 is due to bit 14 MDI/MDIX Mode, does not affect anything. The significant difference is the '7' or '5' as the LSB, this tells you the Speed Status.

Example: After powering and linking the PHY in 10 Mbps, register 0x0010 is read at value 0x0017 meaning Bits [4, 2, 1, 0] are high. These bits confirm: Auto-Negotiation is complete, Full-Duplex, 10 Mbps Mode, and valid link established.

Repeating this process for any values distinct from the expected values shown in the table will help diagnose the exact state of the PHY for any encountered issues.

2.4 Peripheral Pin Checks

The following section details the expected values of various peripheral output pins of the PHY during operation - measure and compare the noted pin outputs to verify PHY operation.

2.4.1 Probe the RESET_N Signal

The reset input is active low. It is important to confirm that the controller is not driving the RESET_N signal low. Otherwise, the device will be held in reset and will not respond.

2.4.2 Probe the RBIAS pin

The RBIAS resistor is used to develop the internal bias currents and voltages in the PHY. It is specified for 1% tolerance so that the PHY can meet the tightest IEEE 802.3 specifications.

Measure the DC value of the voltage across the RBIAS resistor and confirm that the voltage is 2.7 V.

Power down the board and verify that the RBIAS resistor value is 4.87kΩ ±1%

2.4.3 Probe the Serial Management Interface (MDC, MDIO) Signals

If a register read and write is successful, then this section can be skipped as there's no need to verify correct operation for the Serial Management Interface.

MDIO should pull up to the I/O supply when undriven. Probe MDIO to confirm the default voltage. Before probing make sure a 2.2k **Ω** pullup resistor is connected on the MDIO line, if it is an extended register make sure to follow Extended Register Access. If seeing an abnormal value that is not expected probe MDC and MDIO for further debugging.

Attempt to write and read the registers. Verify the MDIO data sequence with the data sheet to make sure the MDIO read access timing is correct.

Table 2-3. SMI Protocol Structure

SMI Protocol	<idle><start><op code><device addr><reg addr><turnaround><data><idle>
Read Register	<idle><01><10><AAAAA><RRRRR><Z0><XXXX XXXX XXXX XXXX><idle>
Write Register	<idle><01><01><AAAAA><RRRRR><10><XXXX XXXX XXXX XXXX><idle>

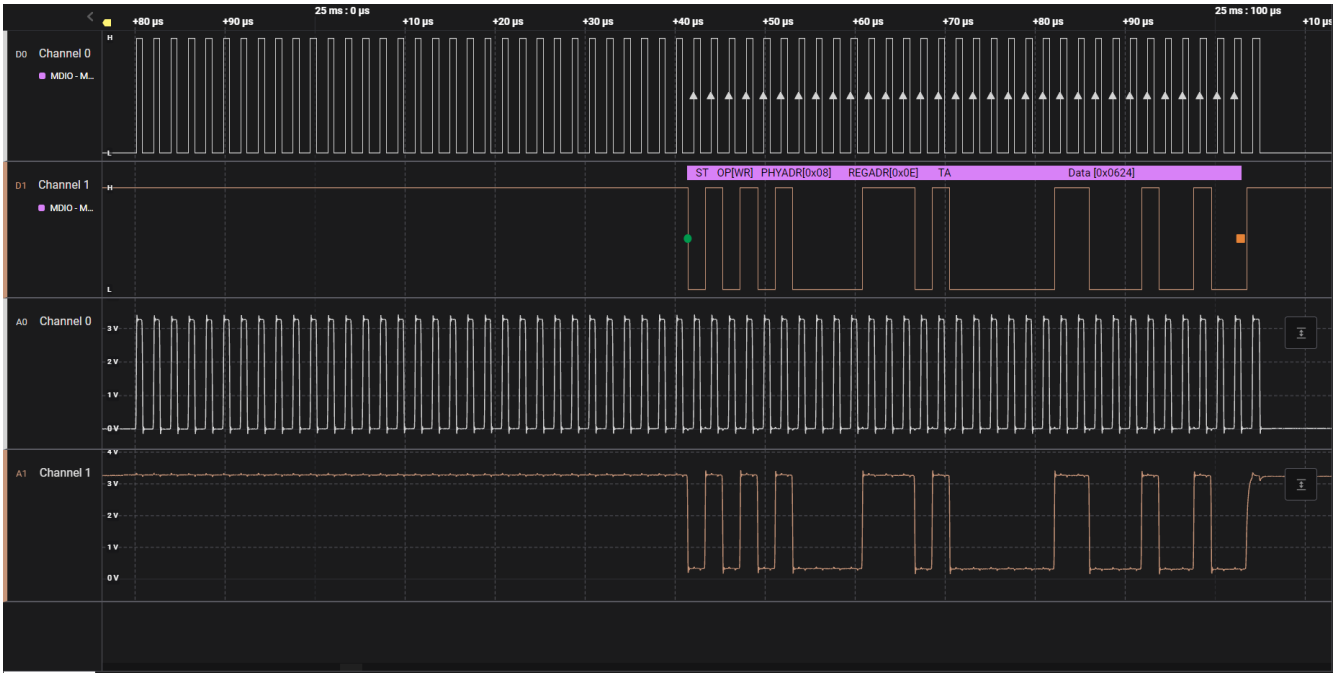

Figure 2-2. MDC/MDIO Write Example

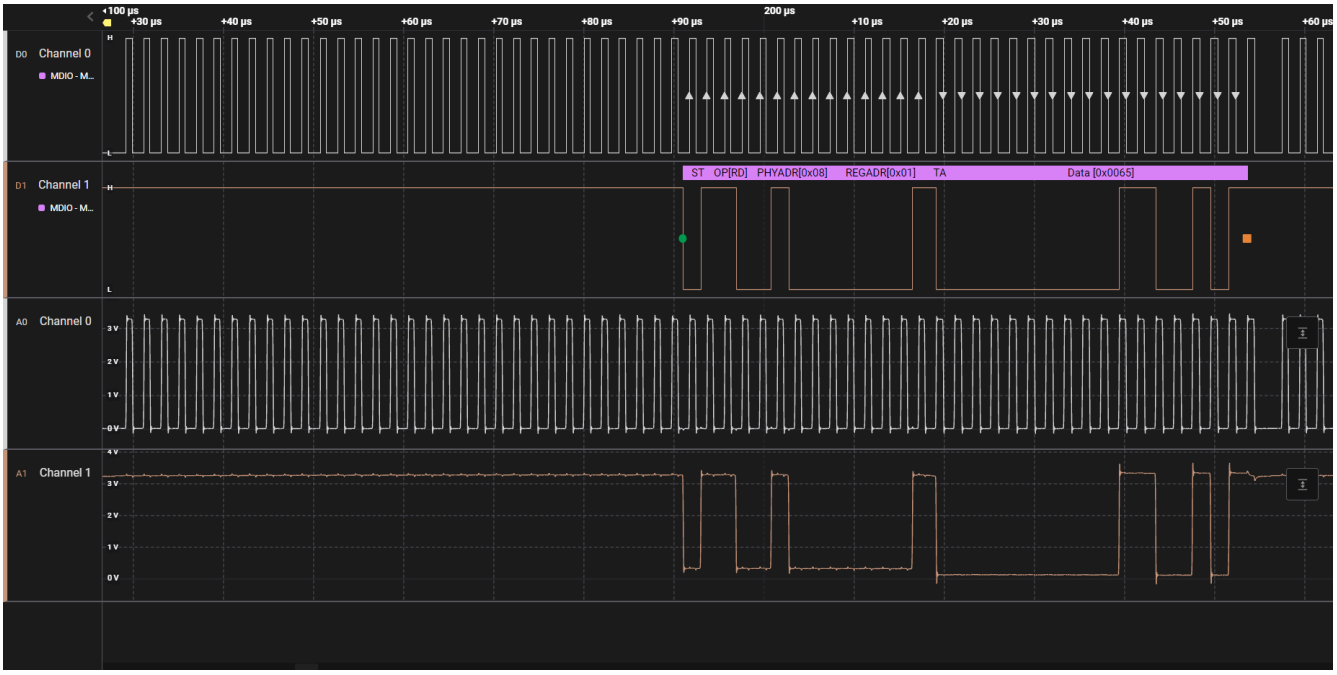

Figure 2-3. MDC/MDIO Read Example

2.4.4 Probe the MDI Signals

A link pulse should be visible on the channel transmit and receive differential pair (TD_P, TD_M).

A short Ethernet cable with 100 Ohm terminations can be used for measuring the MDI signals. A terminated cable is shown in Figure 2-4. A connection diagram for making measurements with the terminated cable is shown in Figure 2-5.

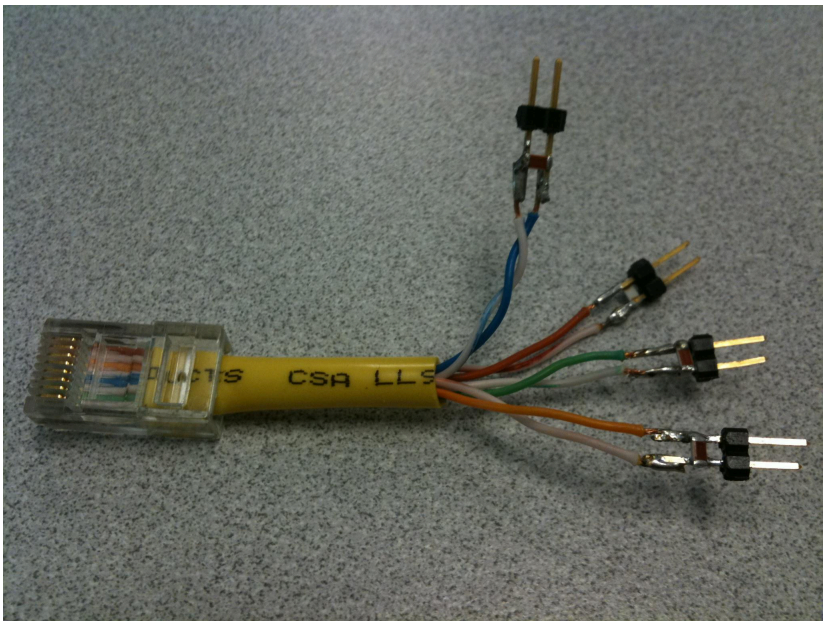

Figure 2-4. 100 Ω Terminated Cable for MDI Signal Measurement

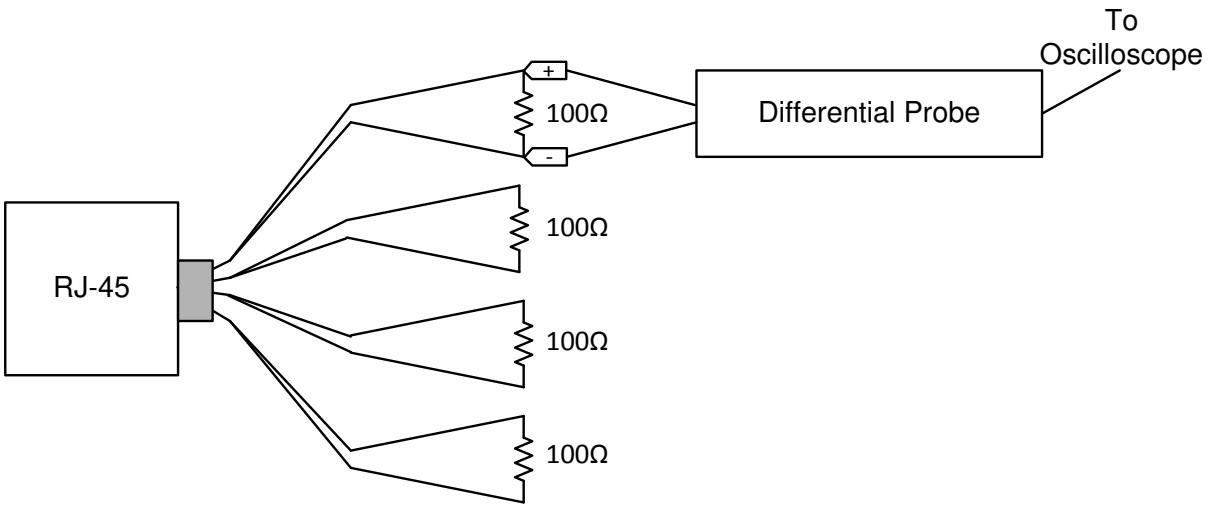

Figure 2-5. Connection Diagram for 100 M Terminated Cable

Link pulses are nominally 100 ns wide and occur every 16 ms. Figure 2-6 shows a correct link pulse.

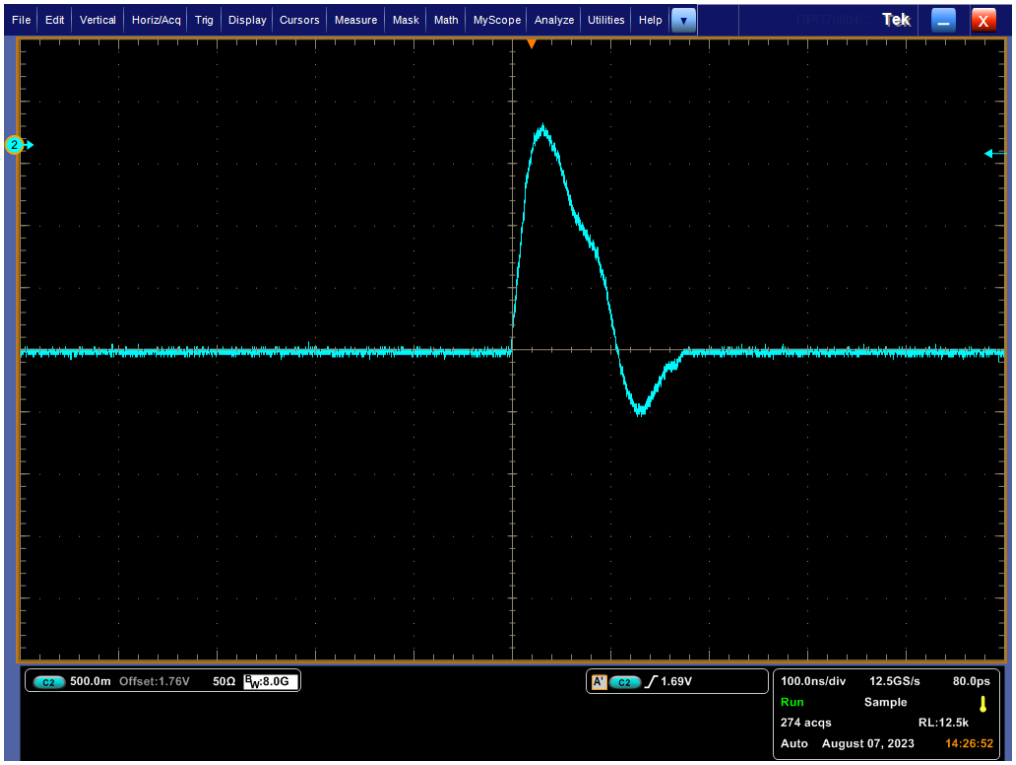

Figure 2-6. DP83822 Link Pulse

2.5 Verifying Strap Configurations During Initialization

The strap values can be read from registers 0x467 (SOR_1) and 0x468 (SOR2), these registers are extended registers and can only be accessed using Extended Register Access. In the cases that the read strap value differs from the desired strap configurations, the voltage levels of the strap pins can be measured. Measurements can be made during power up and after power up when the RESET_N signal is asserted. Strap configurations can also be verified with the strap tool built in the DP83822 Schematic Checklist.

The DP83822 uses the receive path functional pins as bootstrap options to place the device into specific modes of operation. The values of these pins are sampled at power up or hardware reset, through either the RESET pin or bit[15] in the PHY Reset Control Register (PHYRCR, address 0x001F). The DP83822 bootstrap pins are 4-level, which are described in greater detail below. Because bootstrap pins may have alternate functions after reset is de-asserted, they should not be connected directly to VCC or GND. Pullup and pulldown resistors are required for proper operation.

Configuration of the device may be done via 4-level strapping or via serial management interface. A pullup resistor and a pulldown resistor of suggested values should be used to set the voltage ratio of the bootstrap pin input and the supply to select one of the possible modes.

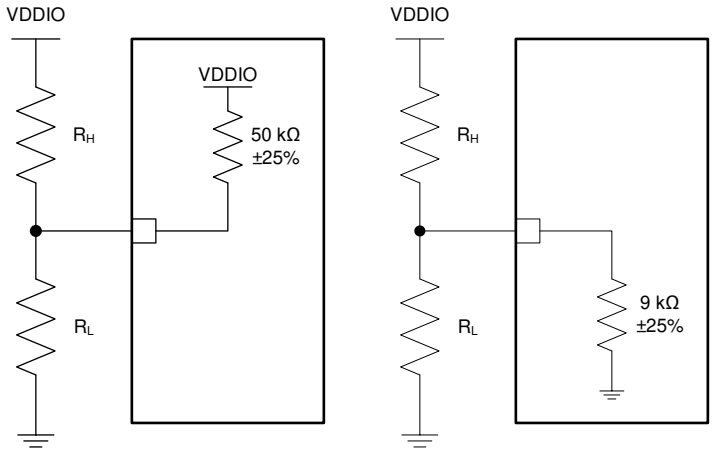

Figure 2-7. Bootstrap Circuits

Table 2-4. Recommended 4-Level Strap Resistor Ratios

Mode (1)	Ideal $R_{H\ (k\Omega)}$	Ideal $R_{L\ (k\Omega)}$
PULLDOWN PINS (9 kΩ)		
1 (Default)	**OPEN**	**OPEN**
2	10	2.49
3	5.76	2.49
4	2.49	OPEN
PULLUP PINS (50 kΩ)		
1	OPEN	1.96
2	13	1.96
3	6.2	1.96
4 (Default)	**OPEN**	**OPEN**

(1) Strap resistors with 1% tolerance are recommended.

Table 2-5. 4-Level Strap Voltage Ratios

Target Voltage (1)	Mode 1	Mode 2	Mode 3	Mode 4
V_{max} (V)	0.098 x VDDIO	0.181 x VDDIO	0.277 x VDDIO	VDDIO
V_{typ} (V)	0	0.165 x VDDIO	0.252 x VDDIO	VDDIO
V_{min} (V)	0	0.148 x VDDIO	0.227 x VDDIO	0.694 x VDDIO

(1) Ensured by production test, characterization or design.

2.6 Debugging Link Quality

There are several possible sources of link problems:

- Cable length and quality
- Clock quality of the 25 MHz reference clock
- MDI signal quality

- Which advertising mode the PHY is in

To verify that a link-up is successful, confirm that Register 0x0001 Bit [2] is read as high [1] and visually inspect if the link LED is lit if applicable. The link quality could cause packet losses and CRC errors despite successful link-up, therefore it is a good practice to always verify the signal quality between the PHY and link partner to ensure signal integrity.

After running through all the previous steps to ensure the PHY functions successfully, the most common link issues happen with the cable or the connector. To find which advertising mode the PHY is in, Register 0x0004 Auto-Negotiation Advertisement Register (ANAR) can be read. For the link partner, Register 0x0005 Auto-Negotiation Link Partner Ability Register (ANLPAR) can be read.

With the PHY powered and connected to a link partner, the following registers can be read from to determine the health of the link:

Table 2-6. Link Quality MSE Registers

Channel	Register Address
A	0x218

For a given channel, read the register value to determine the MSE (Mean Square Error), convert to decimal, and see Table 2-7 to determine link quality.

Table 2-7. MSE Link Quality Ranges

Link Quality	Register Address
Excellent	< 522
Good	522 - 827
Poor	> 827

A Time-Domain Reflectometry (TDR) test can also be performed on the PHY to detect problems within the wire's connections and where the fault occurred. For more details regarding different TDR configurations and test modes as well as how to run a TDR test on the PHY, see *How to use the TDR Feature of DP83822*.

2.7 Built-In Self Test With Various Loopback Modes

There are several options for loopback tests that test and verify various functional blocks within the PHY. Enabling loopback mode allows you to inspect the connections between the MAC and the PHY using xMII/PCS/Digital/AFE loopbacks as well as between the PHY and the MDI using reverse loopback.

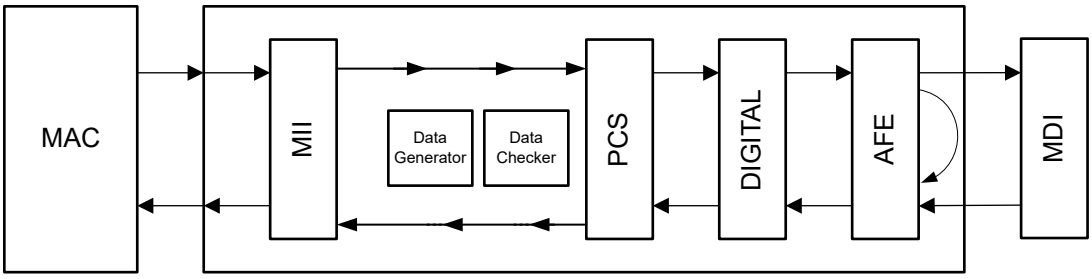

Figure 2-8. MAC-Side Analog Loopback Mode Example

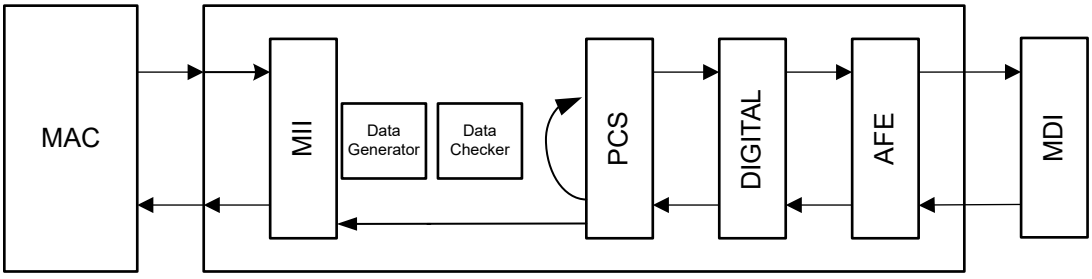

Figure 2-9. Block Diagram, Reverse (Cable-Side) Loopback Mode

The analog loopback is recommended for checking the full data path between the MAC and PHY, while reverse loopback is used with a link partner to verify the data path between the PHY and the MDI(Link Partner).

The device also incorporates an internal PRBS Built-in Self Test (BIST) circuit to accommodate in-circuit testing or diagnostics. BIST can be performed using various loopback modes to isolate any issues to specific parts of the data path by generating packetized data with variable content. Scripts are provided to enable data generator/checker as well as the various loopback modes.

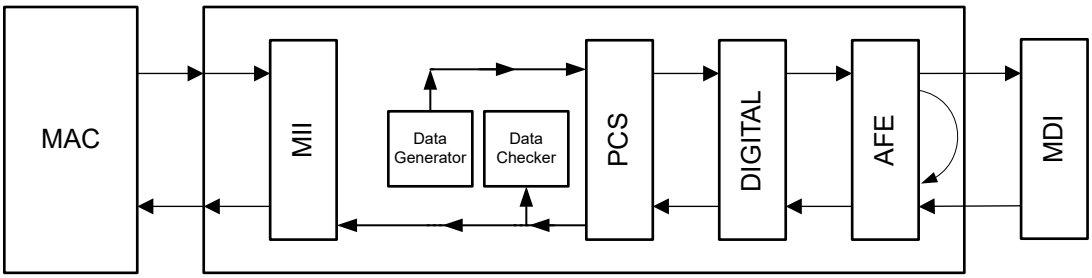

Figure 2-10. Data Generator and Checker With Loopback

Transmitting and Receiving Packets with the MAC:

If generating and checking packets with the MAC is possible, and the PHY has a working link partner with reverse loopback capability, verify the full data path as follows:

1. Power and connect the PHY to the MAC and a working link partner.
2. Enable reverse loopback on the link partner.
3. Transmit test packets from the MAC to the PHY.
4. Verify the MAC receives the same test packets.

If the MAC receives the same test packets transmitted without issue, the full data path through MAC → PHY → Link Partner (MDI) is valid. If this test does not pass, perform analog loopback to isolate the issue along the data path (disconnect cable side connections before running analog loopback):

1. Power and connect the PHY to the MAC.
2. Enable analog loopback on the PHY (write Register 0x0016 = 0x0108).
3. Transmit test packets from the MAC to the PHY.
4. Verify the MAC receives the same test packets.

If the MAC receives the same test packets, the data path through MAC → PHY is valid, and the issue has been isolated to the MDI data path. If this test does not pass, the issue can be on the MAC interface. To check the MAC interface, see Debug the MAC Interface.

Below are example sequence of register reads and writes to perform Analog Loopback:

```
// Analog Loopback

begin

001F 8000 //Hard Reset
0000 2100 //Disables Auto-Neg, Selects 100 Mbps
0016 0108 //Select Analog Loopback
030B 3380 //This helps PRBS LOCK

0016 3108 //Enables PRBS Checker Config & Packet Generation Enable
 //After you write '3108' the register should Read 3b04. (Bit 11 & 9 go high)
001B 807D //Lock Error Counter's Value
001B

//after running this test check register 0010 bit 0 should be 1
end
```

Transmitting and Receiving Packets with BIST:
If generating and checking packets with the MAC is not possible, use an external packet generator or internal PRBS packet generation and check functionalities to verify the data path. Perform reverse loopback with PRBS and a working link partner as follows:

1. Power and connect the PHY to a link partner.
2. Enable PRBS packet generation on the PHY (write 0x16 to 5000).
3. Enable reverse loopback on the link partner
4. Wait at least one second, then check PRBS lock status on the PHY (read register 0x17[11:10]).

If register 0x17[11] is high, the data path through PHY → MDI is valid. If this test does not pass, the issue could be on the PHY's internal data path or the MDI. To verify the internal data path, perform PRBS with analog loopback using the following script. If the internal data path is valid, then the issue is isolated to the MDI (assuming the link partner is working).

Below are example sequences of register reads and writes to perform BIST when using two DP83822 PHY's:

```
// Reverse Loopback on PHY

begin

001F 8000 //Hard Reset
0000 2100 //Disables Auto-Neg, Selects 100 Mbps
0016 5000 //Enables PRBS packet generation
0017 // check PRBS lock status

end

// Reverse Loopback on Link Partner

begin

001F 8000 //Hard Reset
0000 2100 //Disables Auto-Neg, Selects 100 Mbps
0016 0110 //Select Reverse Loopback

end
```

2.8 Debug the Fiber Connection

Fiber Network Circuit shows the recommended circuit for a 100-Mbps fiber network. Variations with PCB and component characteristics require that the application be tested to verify that the circuit meets the requirements of the intended application.

All resistors and capacitors should be placed as close to the fiber transceiver as possible.

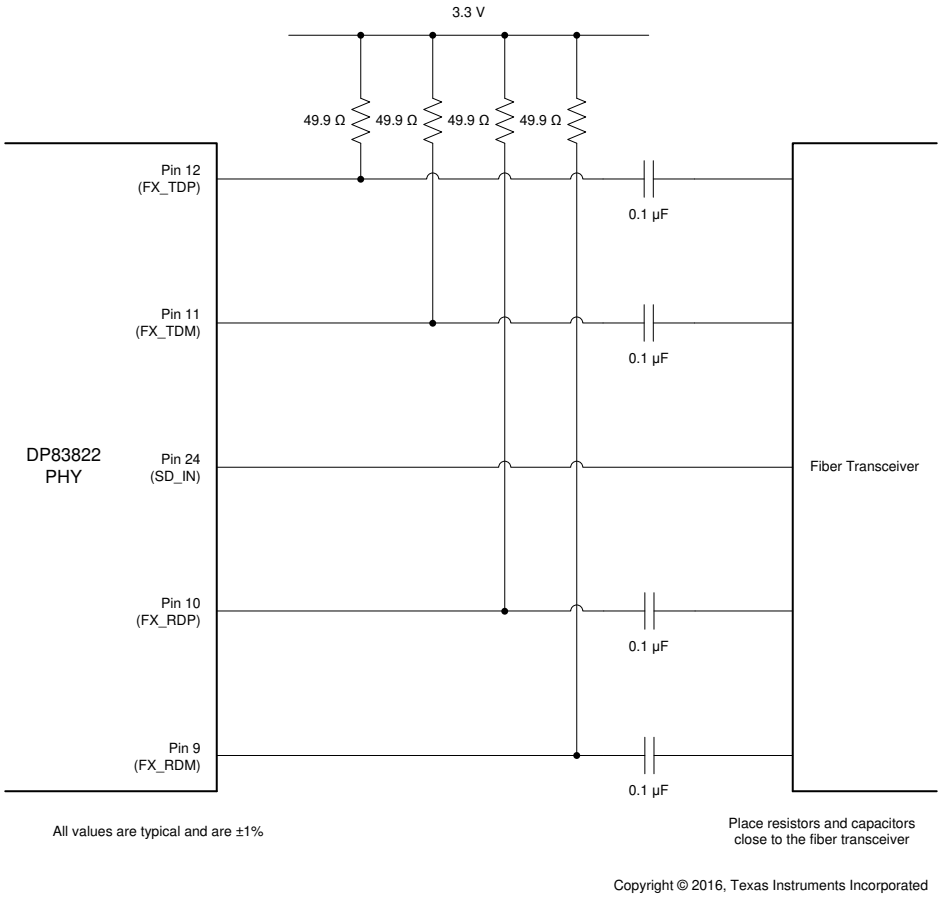

Figure 2-11. Fiber Network Circuit

Note

For PECL and LVPECL applications. For the recommended additional capacitors and resistors needed, see *DP83822 EVM User's Guide*.

Note

SFP Fiber Transceiver usually have integrated AC coupling capacitors. Adding external capacitors may not be needed.

The DP83822 provides IEEE 802.3 compliant 100BASE-FX operation. Hardware bootstrap or register configuration can be used to enable 100BASE-FX operation.

The DP83822IF and DP83822HF are the fiber capable variants of the DP83822. Bit 2 in Register 0x0001 indicates link status for both Copper and Fiber modes of operation. In copper mode, this register behaves as expected: Bit 12 will toggle between '1' and '0' according to the Link Status. However in Fiber, this bit will not toggle when the link status changes. In order to check the link status accurately, a Soft-Reset must be performed (Set Register 0x001F = 4000) before reading Register 0x0001.

The DP83822 also has the added feature of a signal detection pin for direct connection to an industry standard fiber transceiver. When enabling 100BASE-FX operation using the FX_EN bootstrap, AMDIX_EN bootstrap turns into SD_EN bootstrap. If 100BASE-FX operation is enabled by setting FX_EN to either bootstrap mode 2 or 3, SD_EN will enable signal detection pin, LED_1, when SD_EN is set to either bootstrap mode 3 or 4. Please see Verify Strap Configurations During Initialization for mode information regarding hardware bootstraps.

Note

100BASE-FX signal detect pin (LED_1) polarity is controlled by bit[0] in the Fiber General Configuration Register (FIBER GENCFG, Register 0x0465). By default, signal detect is an active HIGH polarity.

Note

TI recommends connecting Signal Detect pin from the Optical Transceiver to the LED_1 pin and enable it using SD_EN bootstrap pin in 100BASE-FX mode. The LED_1 pin is not used in design and that, if the electrical link between the fiber module and the DP83822 is broken, disconnected or otherwise disrupted, the link will recover only by initiating a soft reset through MDIO/MDC interface.

Table 2-8. Bootstrap Configuration

Pin Name	Pin#	PU/PD	Mode	Description
COL	29	PU	2 or 3	FX_EN: Enables 100BASE-FX
RX_ER	28	PU	3 or 4	SD_EN: Enables 100BASE-FX Signal Detection on LED_1 when set to '1'. FX_EN strap must be enabled for SD_EN strap to be functional. Signal Detection is Active HIGH, but polarity can be changed using the Fiber General Configuration Register (FIBER GENCFG, Register 0x0465).

Table 2-9. 0x0465 Fiber General Configuration Register (FIBER GENCFG)

Bit	Name	Type	Default	Function
0	100Base-FX Signal Detect Polarity	R/W	0	100Base-FX Signal Detect Polarity: 1 = Signal Detect is Active LOW 0 = Signal Detect is Active HIGH When set to Active HIGH, Link drop will occur if SD pin senses a LOW state (SD = '0'). When set to Active LOW, Link drop will occur if SD pin senses a HIGH state (SD = '1'). Note: To enable 100BaseFX Signal Detection on LED_1 (pin #24), strap SD_EN = '1'

2.9 Debug the MAC Interface

RGMII

The RGMII signals are summarized in Table 2-10.

Table 2-10. RGMII Signals

Function	Pins
Data Signals	TX_D[3:0]
	RX_D[3:0]
Transmit and Receive Signals	TX_CTRL
	RX_CTRL
Clock	TX_CLK
	RX_CLK

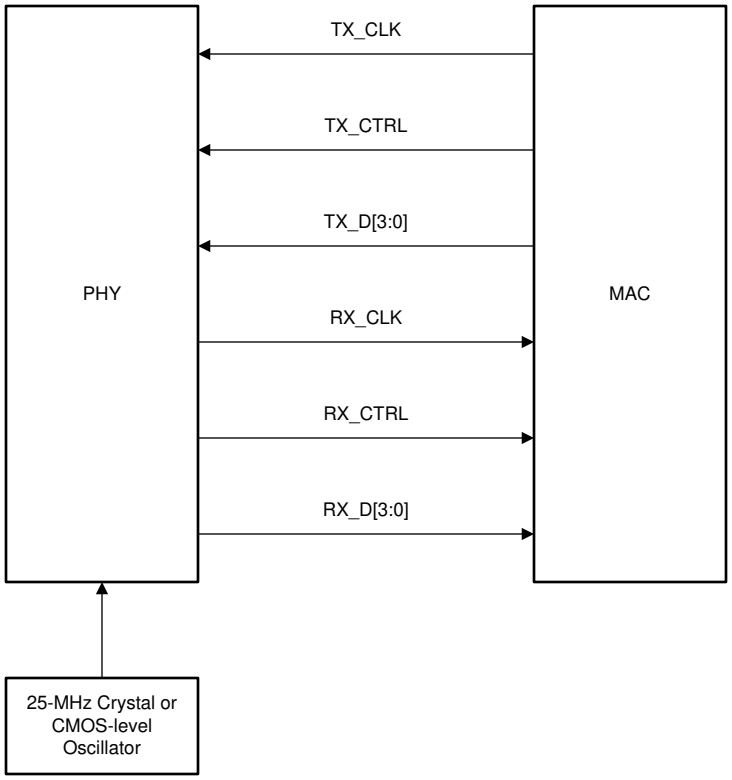

Figure 2-12. RGMII Signaling

In order for the MAC to be able to transmit and receive the correct data from the PHY, the correct RGMII modes must be selected such that both the PHY and the MAC are not simultaneously in align mode or shift mode for the Tx and Rx side. Table 2-11 lists the correct RGMII delay configurations.

Table 2-11. RGMII Shift Configurations

MAC Configuration	Required PHY Configuration
RGMII Align on Rx	RGMII Shift on Rx
RGMII Shift on Rx	RGMII Align on Rx
RGMII Align on Tx	RGMII Shift on Tx
RGMII Shift on Tx	RGMII Align on Tx

Reference the waveforms below to verify the expected MAC data and clock signals for RGMII Mode. The table displays specs taken from the device-specific data sheet that are shown in the waveforms.

Figure 2-13. RX_CLK and RX_D0 Timing in RGMII Align Mode (Yellow Waveform (Channel 1) = RX_CLK, Blue Waveform (Channel 2) = RX_D0)

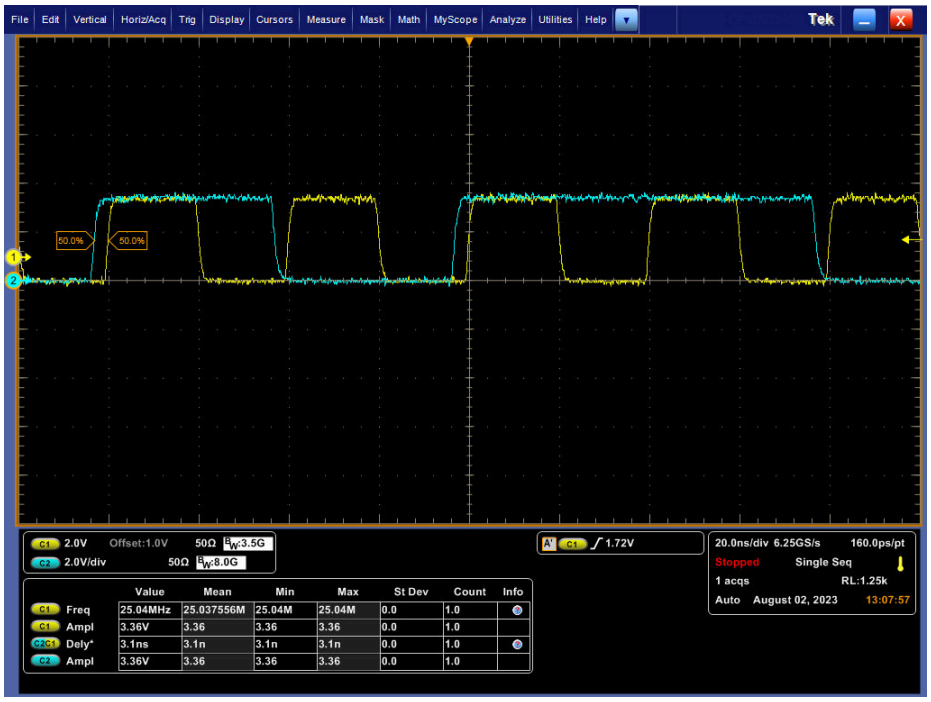

Figure 2-14. RX_CLK and RX_D0 Timing in RGMII RX Shift Mode (Yellow Waveform (Channel 1) = RX_CLK, Blue Waveform (Channel 2) = RX_D0)

For RGMII Rx shift mode, verify that RX_CLK is shifted by 3.5 ns and for RGMII TX Clock Shift that TX_CLK is shifted by 3.5 ns.

Table 2-12. RGMII Input Timing Specifications

Parameter	Test Condition	Min	Typ	Max	Unit
T_{cyc}	TX_CLK / Clock Cycle Duration	36	40	44	ns
$T_{setup(align)}$	TX_D[3:0], TX_CTRL setup to TX_CLK (align mode)	1	2		ns
$T_{hold(align)}$	TX_D[3:0], TX_CTRL hold to TX_CLK (align mode)	1	2		ns

Table 2-13. RGMII Output Timing Specifications

Parameter	Test Condition	Min	Typ	Max	Unit
$T_{skew(align)}$	RX_D[3:0], RX_CTRL delay from RX_CLK (align mode)	-500	0		ps
$T_{setup(shift)}$	RX_D[3:0], RX_CTRL delay from RX_CLK (shift mode enabled, default)	1.2	2		ns
T_{cyc}	RX_CLK / Clock Cycle Duration	36	40	44	ns
Duty_G	RX_CLK / Duty Cycle	40	50	60	%
T_r/T_f	RX_CLK / Rise, Fall Time (20% to 80%)			750	ps

RMII

The incorporates the Reduced Media Independent Interface (RMII) as specified in the RMII specification from the RMII consortium. The purpose of this interface is to provide a reduced pin count alternative to the IEEE 802.3u MII as specified in Clause 22. Architecturally, the RMII specification provides an additional reconciliation layer on either side of the MII, but can be implemented in the absence of an MII. The offers two types of RMII operations: RMII Slave and RMII Master. In RMII Slave operation, the operates off of a 50-MHz CMOS-level oscillator connected to the XI pin and shares the same clock as the MAC. In RMII Master operation, the operates off of either a 25-MHz CMOS-level oscillator connected to XI pin or a 25-MHz crystal connected across XI and XO pins. A 50-MHz output clock referenced from any of the three GPIOs is connected to the MAC.

Note

If RMII Master mode is configured through bootstraps, a 50-MHz output clock will automatically be enabled on RX_D3 (GPIO3).

The RMII specification has the following characteristics:

- Supports 100BASE-FX, 100BASE-TX and 10BASE-Te.
- Single clock reference sourced from the MAC to PHY (or from an external source)
- Provides independent 2-bit wide transmit and receive data paths
- Uses CMOS signal levels, the same levels as the MII interface

In this mode, data transfers are two bits for every clock cycle using the internal 50-MHz reference clock for both transmit and receive paths.

The RMII signals are summarized in Table 2-14.

Table 2-14. RMII Signals

Function	Pins
Data Signals	TX_D[1:0]
	RX_D[1:0]
Transmit and Receive Signals	TX_EN
	CRS_DV

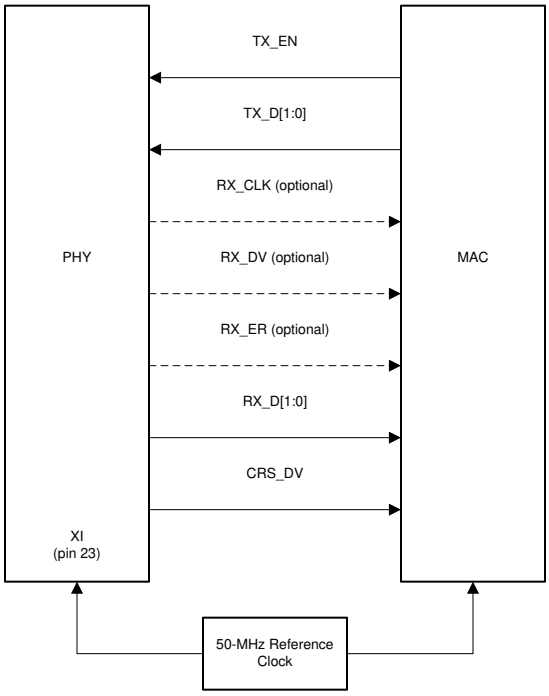

Figure 2-15. RMII Slave Signaling

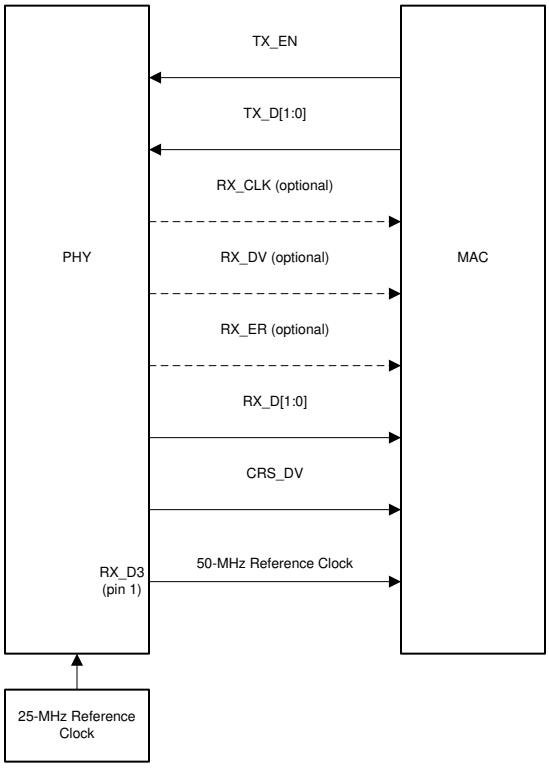

Figure 2-16. RMII Master Signaling

Note

For using the DP83822 in RMII repeater mode, see the DP83822 RMII Repeater Mode.

For more information on reduced media independent interface, see the *Reduced Media Independent Interface (RMII)* section of the *DP83822 Robust, Low Power 10/100 Mbps Ethernet Physical Layer Transceiver Data Sheet*.

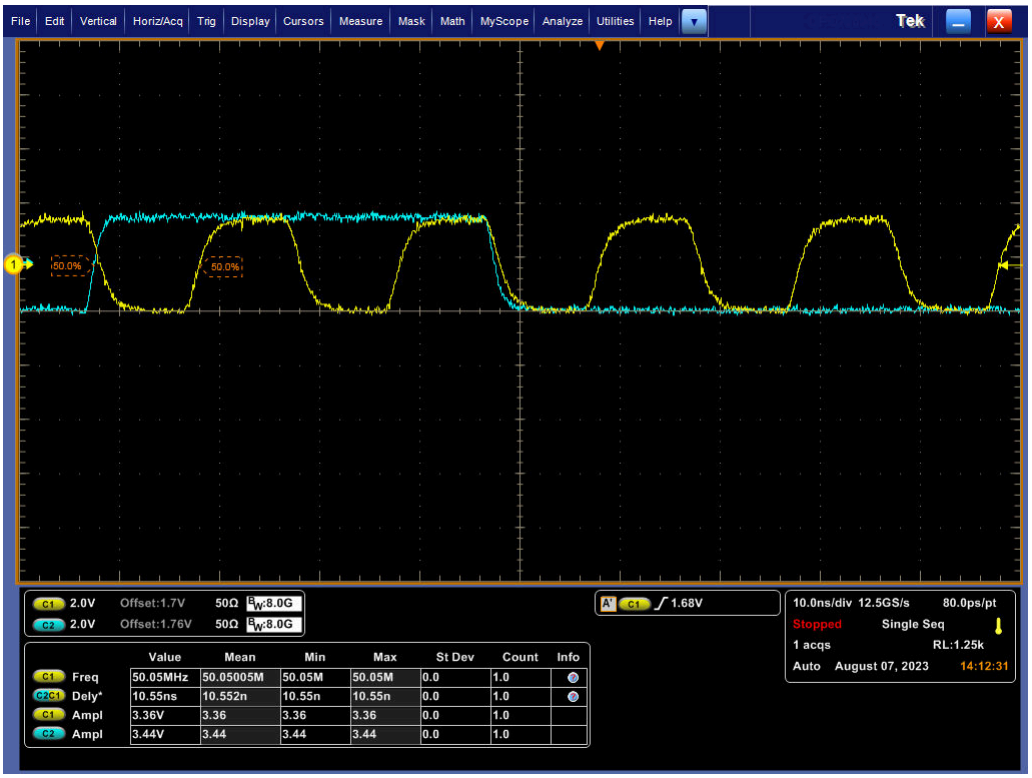

Figure 2-17. RX_CLK and RX_D0 Timing for RMII (Yellow Waveform (Channel 1) = RX_CLK, Blue Waveform (Channel 2) = RX_D0)

MII

The Media Independent Interface is a synchronous 4-bit wide nibble data interface that connects the PHY to the MAC . The MII is fully compliant with IEEE 802.3-2002 clause 22.

The MII signals are summarized in Table 2-15.

Table 2-15. MII Signals

Function	Pins
Data Signals	TX_D[3:0]
	RX_D[3:0]
Transmit and Receive Signals	TX_EN
	RX_DV
Line-Status Signals	CRS
	COL
Clock	TX_CLK
	RX_CLK

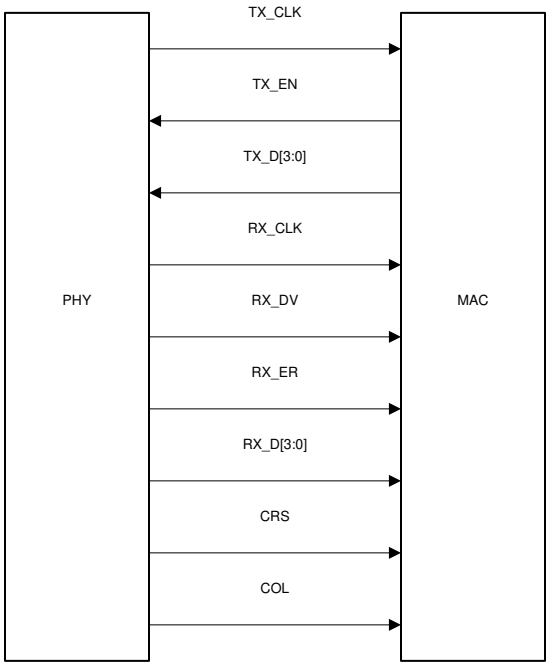

Figure 2-18. MII Signaling

Additionally, the MII interface includes the carrier sense signal (CRS), as well as a collision detect signal (COL). The CRS signal asserts to indicate the reception or transmission of data. The COL signal asserts as an indication of a collision which can occur during Half-Duplex mode when both transmit and receive operations occur simultaneously.

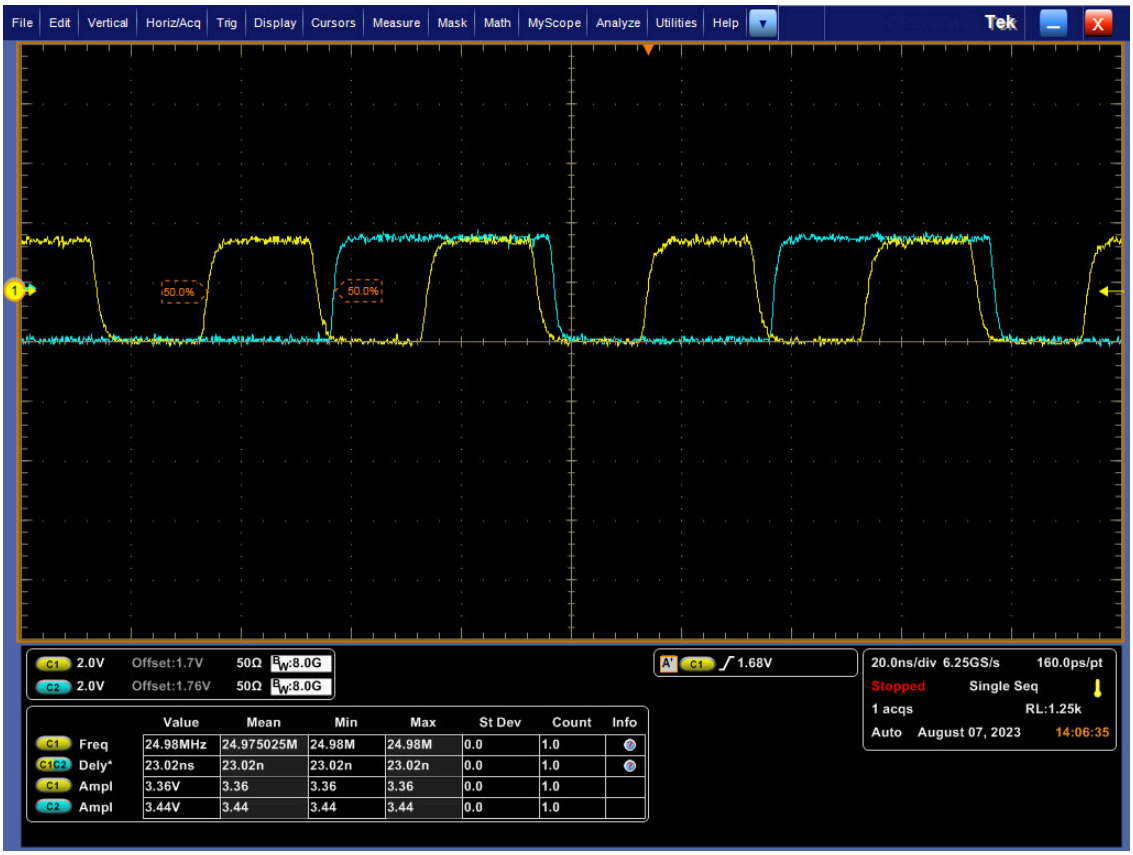

Figure 2-19. RX_CLK and RX_D0 Timing for MII (Blue Wave (Channel 2) = RX_CLK, Purple Wave (Channel 3) = RX_D0)

2.10 Debug the Start of Frame Detect

The IEEE 1588 indication pulse at the SFD can be delivered to any of the following pins: LED_0, LED_1 (GPIO1), COL (GPIO2), RX_D3 (GPIO3), INT/PWDN_N and CRS. The exact timing of the pulse can be adjusted via Register 0x003F.

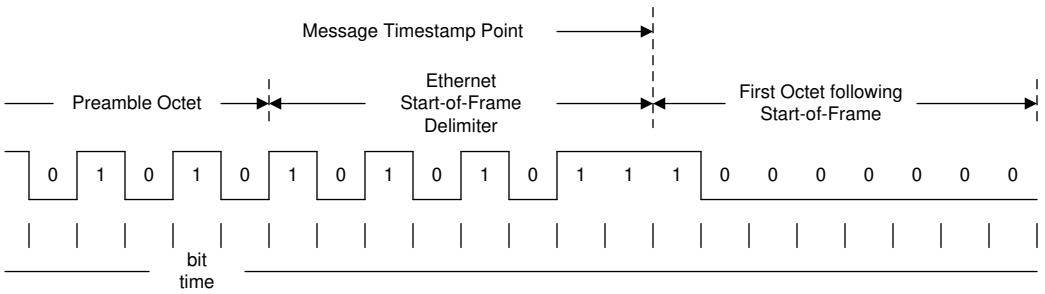

Figure 2-20. IEEE 1588 Message Timestamp Point

There are three registers that are able to control the routing of the IEEE 1588 transmit and receive indications. Register 0x003E is able to route both transmit and receive indications to LED_0 (GPIO1), COL (GPIO2), CRS and INT/PWDN_N. Two additional registers allow for additional pin selections and a centralized location for GPIO controls through the use of the IO MUX GPIO Control Registers, Register 0x0462 and Register 0x0463. After enabling/setting the RX_SFD and TX_SFD pins in registers IOCTRLx, write the following two registers:

- Program (Register 0x0456 = value 0x000A)
- Program (Register 0x04A0 = value 0x1080)
  - Note that register 0x04A0 is enabling "bit 7: WOL". This helps improve the accuracy of SFD detection and is not a must change. This does not cause PHY to detect WoL packets, as WoL function needs additional register configurations.

Note

A software reset has to be performed to load these register values (Register 0x001F = value 0x4000).

2.11 Tools and References

2.11.1 DP83822 Register Access

If register access is not readily available in the application, USB-2-MDIO GUI is available from TI and can be used with an MSP430 Launchpad, purchasable through the TI eStore. The GUI supports reading and writing registers as well as running script files. It can be used with the DP83822 and the other devices in TI's Ethernet portfolio. The USB-2-MDIO User's Guide and GUI are available for download at: http://www.ti.com/tool/usb-2-mdio.

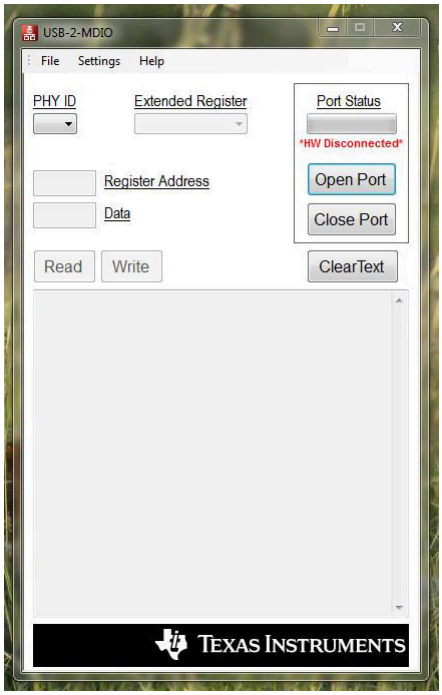

Figure 2-21. USB-2-MDIO GUI

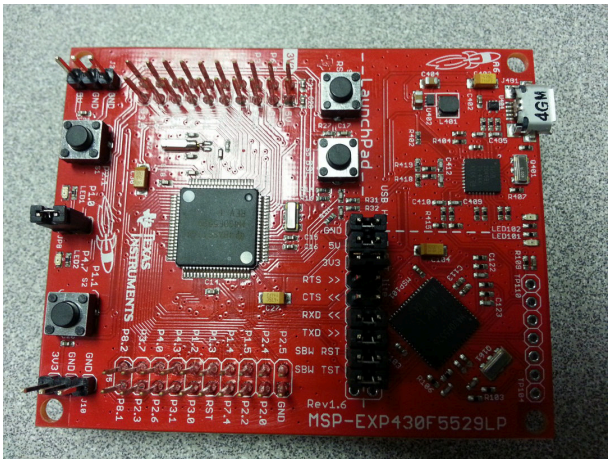

Figure 2-22. MSP430 LaunchPad

Below is an example script that can also be found in the USB-2-MDIO GUI in the Help menu:

```
// This is how you make a comment. All scripts must start with 'begin'
begin
// To read a register, all you need to do is put down the 4 digit
// HEX value of the registers (from 0000 to FFFF)
// Example to read registers 0001, 000A, and 0017
0001
000A
0017
// To write a register, all you need to do is put down the 4 digit
// HEX value of the register (from 0000 to FFFF) followed by the
// HEX you desire to configure the register to (from 0000 to FFFF)
// Example to write 2100 to register 0000 and
// Example to write 0110 to register 0016
0000 2100
0016 0110
// You must end the script by adding 'end' once you are finished
end
```

The Serial Management Interface defined by IEEE 802.3 is a single master bus. The MDC clock is generated by the bus master, typically an Ethernet MAC. To use the USB-2-MDIO GUI, connections must be made directly between the MSP430 Launchpad and the DP83822 MDIO and MDC pins.

- MSP430 Pin 4.2 → PHY's MDIO Pin
- MSP420 Pin 4.1 → PHY's MDC Pin

2.11.2 Extended Register Access

To read and write registers in extended register space, see the following procedures:

Write procedure for MMD "1F" registers:

write reg<000D> = 0x001F

write reg<000E> = <address>

write reg<000D> = 0x401F

write reg<000E> = <value>

Read procedure for MMD "1F" registers:

write reg<000D> = 0x001F

write reg<000E> = <address>

write reg<000D> = 0x401F

read reg<000E>

To write a register in the extended register set and automatically increment the address register to the next higher value following the write operation:

Write (Post Increment) procedure for MMD "1F" registers:

write reg<000D> = 0x001F

write reg<000E> = <address>

write reg<000D> = 0x401F

write reg<000E> = 0x0C50

write reg<000E> = <value>

To read a register in the extended register set and automatically increment the address register to the next higher value following the read operation:

Read (Post Increment) procedure for MMD "1F" registers:

write reg<000D> = 0x001F

write reg<000E> = <address>

write reg<000D> = 0x801F

read reg<000E>

read reg<000E>

Note

Above write and read procedure is normally used for registers with address greater than 0x001F, but the procedure can also be used for any address in gernal.

2.11.3 Software and Driver Debug on Linux

The two essential components required for the PHY to function on a Linux system are the device tree and driver file, for which the DP83822 drivers can be found here. Below is a sample format of what a device tree looks like.

```
mdio0 {
#address-cells = <1>;
#size-cells = <0>;
ethphy0: ethernet-phy@0 {
reg = <0>;
rx-internal-delay-ps = <1>;
tx-internal-delay-ps = <1>;
};
};
```

Note

The example bindings file can be found in this path: root/Documentation/devicetree/bindings/net/ti,dp83822.yaml.

2.11.3.1 Common Terminal Outputs and Solutions

Using the terminal command "dmesg | grep mdio", there might be several clues on what's causing the PHY to not function appropriately from a software standpoint.

```
$ dmesg | grep "mdio"
```

One of the possible outputs is as follows:

```
$ mdio_bus xxx.ethernet-x: MDIO device at address 8 is missing
```

This message indicates that the PHY is not found on the MDIO bus, which could be caused by several issues, the most common one being a missing or incorrect device tree, but could also be due to a non-functional PHY or a bad SMI connection.

Once the PHY can be detected on the MDIO bus, another common error message is as follows:

```
$ Generic PHY xxx.ethernet-x: attached PHY driver [Generic PHY]
```

This message indicates that the driver file for the corresponding PHY is not loaded correctly or not present at all, and Linux loaded in a generic driver that most likely won't work with the PHY. In that case, verify that the driver successfully compiled and added to Linux, making sure the driver matches with the model of PHY used.

Finally, a message like this could display:

```
$ am65-cpsw-nuss c000000.ethernet eth3: PHY [c000f00.mdio:05] driver [TI DP83822] (irq=POLL)
```

This message shows that the PHY has the correct driver loaded and is detected successfully. Run ifconfig to verify the network interface is present.

Example ifconfig output when the PHYs are successfully recognized as network adapter(s):

```
root@j7-evm:~# ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500  metric 1
        ether 24:76:25:a2:62:8b  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536  metric 1
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 82  bytes 6220 (6.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 82  bytes 6220 (6.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The next step is to verify the successful data transfer.

3 References

For information on hardware and software configurations for EMC.EMI compliance tests, see the following documents.

Texas Instruments: *How to Pass IEEE Ethernet Compliance Tests*

Texas Instruments: *How to Configure DP838xx for Ethernet Compliance Testing*

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265