ADIN1100/ADIN1110/ADIN2111 Link and Cable Diagnostics

INTRODUCTION

The ADIN1100, ADIN1110, and ADIN2111 are industrial Ethernet PHY/MAC-PHY devices compliant with the IEEE® 802.3cg™-2019 standard for Ethernet that enables 10 Mbps data transmission over a single twisted pair of conductors over 1 km long cables.

These three devices provide a set of tools that helps the debug and verification of the correct operation of the ADIN1100/ADIN1110/ADIN2111 during design, deployment, and after installation in the field. This application note explains the operation, required configuration, and capabilities of each of the following tools:

- Time domain reflectometry (TDR) engine
- Link quality monitoring
- Physical medium attachment (PMA) test modes
- Loopback modes
- On-chip frame generator and frame checker

TABLE OF CONTENTS

REVISION HISTORY

7/2023—Revision 0: Initial Version

TIME DOMAIN REFLECTOMETRY (TDR)

Given that the 10BASE-T1L compliant PHY enables communication over long cables, debugging a faulty cable may become costly and may be difficult without the right tools. To aid with this, Analog Devices, Inc., 10BASE-T1L products provide a TDR engine that enables cable fault detection, distance to fault, and cable length estimation.

The diagnostics solution is the combination of a highly accurate on-chip TDR engine and a set of algorithms that run on a host microcontroller, allowing maximum flexibility for a wide variety of cables and more advanced cable diagnostic capabilities.

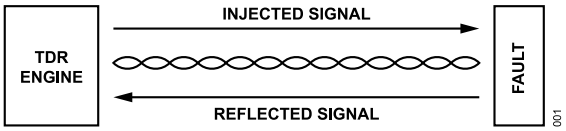

Figure 1. ADIN1100/ADIN1110/ADIN2111 TDR Engine

FAULT DETECTION WITH THE TDR ENGINE

The Analog Devices TDR fault detector has a time resolution of 8.3 ns, which translates to a length resolution of less than 1 m and a detection range of over 1600 m, with an accuracy of 2%.

This fault detector is capable of finding open and short fault conditions even when the device is physically connected to another PHY through their media dependent interface (MDI), which implies that the link partner PHY is potentially transmitting autonegotiation signals. Traditional TDR methods struggle to find faults if other signal sources or noise is also present in the same link. This is not the case of the Analog Devices solution, which makes it suitable for debugging when there is no control over the remote end.

The fault detector is provided as a C-code library containing the high-level functions required for diagnostics. These functions have been optimized to not utilize any advanced processing. Therefore, any low-power microcontroller can execute these functions.

A single function call is sufficient to execute the fault detector. The function returns the type of fault and the distance to the fault in meters from the MDI connector.

TDR Offset Calibration

The C-code library includes a function to calibrate the offset of the TDR measurement. This function in the library is useful given that different MDI circuits may introduce variable delays in the signal path, which can contribute to the offset of the length measurement. For instance, an isolation transformer on the MDI is highly likely to introduce a signal delay that corresponds to a couple of meters in length.

This calibration is not required to run the fault detector, and an average value is provided by default. However, this calibration is recommended for short cables if accuracy is required. If required, the offset can be calibrated once in the lab for a specific MDI circuit implementation, and the offset value can then be stored in nonvolatile memory for future use.

To perform this calibration, the MDI port must be left open or shorted. No load or cable can be connected to the MDI port.

Cable Calibration

By default, the algorithm is optimized to support long reach cables compliant with the IEEE 802.3cg-2019 standard. However, given the wide variety of cable types, which have different insertion loss, return loss, and signal delay characteristics, the C-code library includes a calibration function that optimizes the algorithm to operate with any cable and estimates the nominal velocity of propagation (NVP) of the cable for more accurate length estimations. The length accuracy mainly depends on the accuracy of the NVP value.

To run this calibration, a cable with a known length must be attached to the MDI port, and the end of this cable must be left open or shorted. NVP values are generally between 0.5 and 0.9, and are a property of the construction of the cable. In general, an average NVP value of approximately 0.65 can be assumed. This calibration is not required to run the fault detector, unless higher length accuracy is needed or if nonstandard cables are utilized. A given cable can be calibrated once in the laboratory, and the values can be stored in nonvolatile memory.

Length/Distance to Fault Accuracy

The accuracy of the distance to fault or length measurements mainly depends on the NVP value, which is determined by the accuracy of the cable length used to perform the calibration.

Table 1 provides results for induced faults and distance to fault measurements for different cables and lengths. In all cases, the algorithm was successful in finding the open or short fault conditions induced during the test. The NVP value for the Profibus PA cable used in this test was roughly estimated, and the same was used for the Cat5E and Cat6 cables.

The accuracy of the Cat5E and Cat6 cables in Table 1 can be improved by calibrating the NVP values of these cables using the cable calibration function.

Table 1. Length Estimation Error for Different Cables

Cable Type	Estimated Length (m)	Length Error (%)
Fieldbus Type A—AWG 18	50.2	0.7
Fieldbus Type A—AWG 18	102.1	2.1
Fieldbus Type A—AWG 18	403.4	0.8
Fieldbus Type A—AWG 18	807.6	0.8
Fieldbus Type A—AWG 18	1045.3	1.0

Table 1. Length Estimation Error for Different Cables (Continued)

Cable Type	Estimated Length (m)	Length Error (%)
Fieldbus Type A—AWG 18	1462.9	2.0
Cat5E	133.1	2.4
Cat5E	244.4	1.8
Cat6	73.6	5.1
Cat6	137.2	5.6

LINK QUALITY MONITORING

The ADIN1100, ADIN1110, and ADIN2111 provide the mean squared error (MSE) measurement of the received signal, which directly relates to signal-to-noise ratio (SNR) seen by the PHY receiver. The MSE or SNR can be mapped to a signal quality indicator (SQI) and can be used for assessing the overall 10BASE-T1L link segment/channel quality.

The link quality may be affected by the cable length, the cable properties such as insertion and return loss, presence, quality, and connection of the cable shield, number and quality of possible interconnections between cable segments, as well as level of noise in the environment around the devices and the cable. Therefore, the link quality can provide useful information during a device design, product testing, as well as at installation in the system, and during the lifetime of the system.

SIGNAL-TO-NOISE RATIO AND BIT ERROR RATE

There is a statistical relation between a communication channel SNR and bit error rate (BER). The relation between white noise SNR and 10BASE-T1L BER is shown in Figure 2.

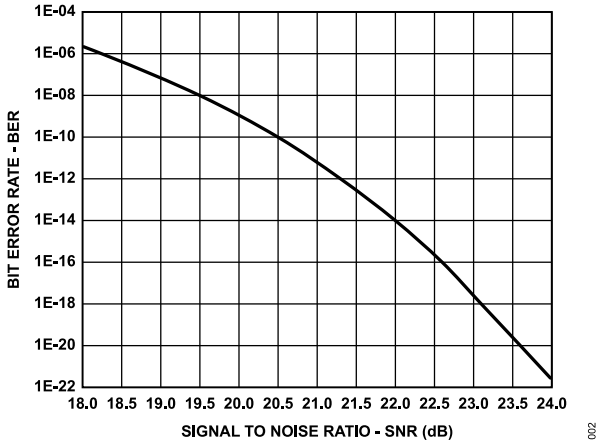

Figure 2. Statistical Relation Between SNR and BER in 10BASE-T1L

The IEEE 802.3cg-2019 standard requires the 10BASE-T1L BER ≤ 10^{-9}, in the presence of the relevant noise. For context, the BER 10^{-9} means 1 bit error every 100 sec in continuous 10 Mbps data, which translates to approximately an SNR of 20.0 dB on the 10BASE-T1L PHY as shown in Figure 2.

With an SNR of 21.0 dB, the BER must be 10^{-11}, which is 1 bit error every 10,000 sec or 2¾ hour, and with an SNR of 22.0 dB, the BER must be 10^{-14}, which is 1 bit error in 115 days. These examples illustrate how the SNR relates to the reliability of the 10BASE-T1L Ethernet.

There are always some errors in any data communication channel. The communication protocols, implemented and operating above the Ethernet physical layer, such as TCP/IP in general use cases, or the specific protocols for industrial or building automation ensure data integrity by frame repetition or error correction as appropriate for a given application. However, the link quality and related error rate of the physical layer must be kept at a certain level for reliable connection. The acceptable error rate may be different in noncritical monitoring compared to a time critical automation network or safety application.

Mean Squared Error at PHY Slicer

The link quality monitoring inside the PHY is implemented as an MSE measurement.

The 10BASE-T1L Ethernet uses PAM3 modulation—the data sent over the cable is coded into symbols of three voltage levels. Inside the receiver, after analog and digital signal processing, is a device called a slicer, which makes the decisions whether the incoming signal voltage level represents a +1, 0, or –1 symbol. An ideal received and scaled signal is already at these exact levels. However, the noise coupled to the Ethernet channel from various sources affects the real signal.

The PHY measures, for each received symbol, an error between the output of the slicer and the received signal already scaled to the correct amplitude level as shown in Figure 3. The mean square value of these errors is then calculated and reported in the PHY MSE_VAL register.

There is a direct relation between the MSE and SNR.

$$SNR = \frac{1}{MSE} \quad (1)$$

$$SNR(\text{dB}) = -MSE(\text{dB}) \quad (2)$$

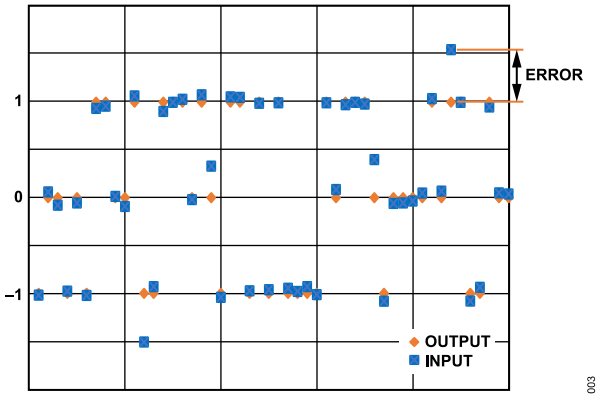

Figure 3. Error Between Ethernet PHY Slicer Input and Output

MSE Reading

The ADIN1100/ADIN1110/ADIN2111 automatically measure the MSE in the background when the 10BASE-T1L link is active and makes it available in the MSE_VAL register.

The MSE_VAL register can be read via the management interface (MDIO or SPI) anytime. After power-up or reset, before the first link is up, the MSE_VAL register value is zero. When the 10BASE-T1L link is up, the MSE_VAL register is updated after each received symbol (every 133 ns). When the link drops, the register still updates. However, the MSE value is incorrect. Therefore, reading and processing the MSE is logical only when the 10BASE-T1L link is up.

The frequency of reading the MSE_VAL register is not limited, and it can be read as often as the management interface allows. Therefore, how often MSE_VAL must be read depends on how fast the link quality is expected to be changing and on how often the link quality needs to be assessed and reported or recorded in the end system application.

An example of polling the link status and reading the MSE is as follows:

1. Read the PMA_PMD_STAT1 register (Device Address 0x01, Register Address 0x0001).
2. Check the PMA_LINK_STAT_OK bit (Bit 2, Mask 0x0004) value. If the bit value is 0, indicating the link is down, skip the following steps and start over again. If the bit value is 1, indicating link is up, continue with the following steps.
3. Read the MSE_VAL register (Device Address 0x01, Register Address 0x830B).
4. Process/use the measured MSE.

MSE Interpretation

The easiest use of the measured MSE is to compare the value read from the MSE_VAL register directly with the MSE register value range and to interpret the link quality as outlined in Table 2 or Table 3.

Alternatively, the MSE_VAL register value can be interpreted as the MSE as follows:

$$MSE\left(\text{dB}\right) = 10\log_{10}\left(MSE_{VAL} \times \frac{1.5523}{2^{18}}\right) \quad (3)$$

And the SNR can be calculated as follows:

$$SNR\left(\text{dB}\right) = -10 \times \log_{10}\left(MSE_{VAL} \times \frac{1.5523}{2^{18}}\right) \quad (4)$$

where:
1.5523 is a coefficient related to the 10BASE-T1L modulation and symbol coding.
2^{18} is a coefficient coming from the implementation of the on-chip logic mapping the 16-bit register to a useful range.

Table 2. Link Quality vs. MSE Register Value

Link Quality	SNR (dB)	MSE Register Value Range (hex)	BER
Poor	<19.5	>0x0766	$>10^{-8}$
Marginal	19.5 to 20.5	0x05E1 to 0x0766	10^{-8} to 10^{-10}
Good	>20.5	<0x05E1	$<10^{-10}$

Table 3. Signal Quality Indicator vs. MSE Register Value

SQI	SNR (dB)	MSE Register Value Range (hex)	BER
0	<18	>0x0A74	$>10^{-7}$
1	18 to 19	0x084E to 0x0A74	$>10^{-7}$
2	19 to 20	0x0698 to 0x084E	10^{-9} to 10^{-7}
3	20 to 21	0x053D to 0x0698	10^{-11} to 10^{-9}
4	21 to 22	0x0429 to 0x053D	10^{-14} to 10^{-11}
5	22 to 23	0x034E to 0x0429	$<10^{-14}$
6	23 to 24	0x02A0 to 0x034E	$<10^{-14}$
7	>24	<0x02A0	$<10^{-14}$

PHY SLICER SPIKES AND ERRORS

The MSE quantity provides an important tool to evaluate the link quality and the effect of noise on the performance of the 10BASE-T1L link. However, given that the MSE is taken as an average value over a period, in cases where the interference is a short transient, the value of the MSE may not reflect this. Yet, there may be enough affectation on the received symbols to produce packet errors.

For this type of transient interference, the ADIN1100, ADIN1110, and ADIN2111 include indicators that keep track of the maximum slicer input error and the number of error spikes at the input of the slicer. These indicators also offer the advantage that can be read while there is a 10BASE-T1L link and normal data flow. Thus, these indicators can be utilized to track the link integrity before a hard fault occurs.

Slicer Maximum Absolute Error

As noted in the Mean Squared Error at PHY Slicer section, after the received analog signal is processed, the slicer makes the decision whether the received signal corresponds to a +1, 0, or −1 (PAM3 symbol). For instance, a processed received signal may have a value of 0.8. Thus, the slicer outputs a +1 symbol, given that 0.8 is closer to +1 than to 0 or −1. The closer the processed signal is to the ideal symbol, the more reliable the communication is.

Figure 4 shows the received processed signals at the input of the slicer and the corresponding ideal symbols at the output of the slicer. Notice that the signal marked as 3 has a value of 0.4. Therefore, the slicer outputs a 0 symbol and the actual error is 0.4. If the error is greater than 0.5, the received signal is closer to a +1 symbol than to its ideal 0 symbol. Thus, the slicer interprets the symbol as a +1, producing a bit error in the received frame.

The slicer maximum absolute error must always be less than a value of 0.5. Values close to 0.5 or greater than 0.5 indicate that the received signal integrity has been affected.

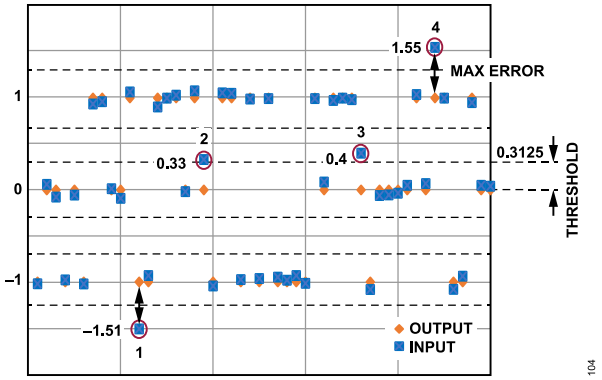

Figure 4. Maximum Slicer Input Error

This maximum error can be tracked over time, before it reaches 0.5, to provide an early indication of link degradation.

Slicer Error Spike Counter

In addition to the maximum slicer absolute error, which only tracks the absolute error of the most deviated symbol, the ADIN1100, ADIN1110, and ADIN2111 keep track of the number of received symbols with an absolute error greater than a threshold. This counter is called the slicer error spike counter. The slicer error spike counter tracks the number of received symbols with an absolute error above 0.3125 threshold.

Figure 4 shows four signals at the input of the slicer, with errors greater than the 0.3125 threshold. Thus, the slicer error spike counter reports a count of four.

Notice that the threshold (0.3125) has been chosen in a way to provide enough headroom for spikes to be detected before they can produce bit errors.

Relevant Register Information

Table 4 shows the relevant register information. The slicer error spike counter is stored in a 16-bit unsigned format. Therefore, its value is the direct read value from the corresponding register. Reading both registers clears their values and the detection re-starts.

The slicer input maximum absolute error can be converted to symbol units as follows:

$$SlicerMaxAbsError = \frac{SLCR\_ERR\_MAX\_ABS\_VAL}{4096} \quad (5)$$

Register Configuration

To perform register configuration, follow these steps:

1. Write a 0x2 to the SPIKE_CNTRS_CNTRL register.
2. Write a 0x2 to the MAX_ABS_VALS_CNTRL register.
3. Read the SLCR_ERR_MAX_ABS_VAL register, which corresponds to the slicer maximum error.
4. Read the SLCR_ERR_SPIKE_CNT register, which corresponds to the slicer error spike counter.

Perform Step 3 and Step 4 before any test to make sure that the spike counter and maximum absolute errors are cleared. This action is particularly useful while performing tests such as electro-magnetic compliance because it is desired to isolate results in pretest and during test.

Table 4. Registers to the Slicer Spike and Error Counters

Register Name	Device Address	Register Address	Description
SLCR_ERR_MAX_ABS_VAL	0x01	0x8308	Slicer maximum absolute error. Latches the value of SLCR_IN_MAX_ABS_VAL.
SLCR_ERR_SPIKE_CNT	0x01	0x8305	Slicer error spike counter. Latches the value of SLCR_IN_SPIKE_CNT.
SPIKE_CNTRS_CNTRL	0x01	0x800E	Specifies whether the spike counters are held when there is no link.
MAX_ABS_VALS_CNTRL	0x01	0x800F	Specifies whether the maximum values are held when there is no link.

The information from the registers in Table 4 can be color coded in a simplified way to provide relevant diagnostics to the end user. A recommended interpretation is explained in Table 5.

Table 5. Link Quality Indication Using Slicer Error Spike Counter and Slicer Maximum Error

Link Quality	Color Indication	Conditions
Poor	Red	Slicer error spike counter > 0 Slicer maximum absolute error ≥ 0.5
Marginal	Yellow	Slicer error spike counter > 0 0.3125 ≤ slicer maximum absolute error < 0.5
Good	Green	Slicer error spike counter = 0 Slicer maximum absolute error < 0.3125

PMA TEST MODES

Subclause 146.5.2 from the IEEE 802.3cg-2019 standard describes three test modes to output different transmission patterns required for the measurement of the electrical characteristics of the transmitter, specified in Subclause 146.5.4 of the same standard. The ADIN1100/ADIN1110/ADIN2111 support all three test modes.

Table 6. PMA Test Modes Description

Test Mode	Description	Measurements
1	Repeated data symbol sequence (+1, −1)	Transmitter output voltage, output jitter, and clock frequency
2	Repeated data symbol sequence of ten +1 followed by ten −1	Transmitter output droop
3	Normal interframe idle signals	Power spectral density (PSD) and power level

Additionally, the ADIN1100/ADIN1110/ADIN2111 provide a transmit disable mode (Subclause 45.2.1.186a.2 of the IEEE 802.3cg-2019 standard), which keeps both transmit and receive paths in normal operation of the PHY. However, this mode transmits 0 symbols only. This mode can be used to measure the MDI return loss specified in Subclause 146.8.3 of the IEEE 802.3cg-2019 standard.

There are two ways to observe the PMA test mode signals across the MDI port with an oscilloscope.

- Terminating the MDI port with a 100 Ω resistor and using a differential active probe directly across MDI+ and MDI−.
- Using two channels of the oscilloscope to observe MDI+ and MDI− independently and then use the math function to perform the subtraction of both signals.

In the second case, it is recommended to configure the input impedance of both channels of the oscilloscope to 50 Ω to guarantee better signal quality. No load resistor needs to be externally provided to the MDI port because the input impedance of the oscilloscope acts like the load.

In both cases, it is recommended to use two 1 µF capacitors in series with MDI+ and MDI− to block the DC component of the signals and avoid damaging the measurement equipment.

The PMA Test Mode 1 section, the PMA Test Mode 2 section, and the PMA Test Mode 3 section contain information related to the three PMA test modes, the Transmit Disable Mode section contains information about transmit disable mode, and the PMA Test Mode 1 to PMA Test Mode 3 Configuration section contains the recommended configuration routine of each of the three PMA test modes. Table 7 shows the register details of interest that are related to the PMA test modes.

Table 7. Registers of Interest Related to PMA Test Modes

Name	Device Address	Register Address	Bits
CRSM_SFT_PD	0x1E	0x8812	0
CRSM_SFT_PD_RDY	0x1E	0x8818	1
AN_EN	0x07	0x0200	12
AN_FRC_MODE_EN	0x07	0x8000	0
B10L_TX_TEST_MODE	0x01	0x08F8	15:13
B10L_TX_DIS_MODE_EN	0x01	0x08F6	14

PMA TEST MODE 1

When the PHY is terminated with a 100 Ω load, the output waveform corresponding to PMA Test Mode 1 ideally looks like a 1.0 V p-p or 2.4 V p-p square waveform (depending on the selected transmitter output level) with a frequency of 3.75 MHz. In practice, this output waveform has nonidealities, which must be within the limits specified in Subclause 146.5.4 of the IEEE 802.3cg-2019 standard. Table 8 contains the electrical specifications that the PHY must meet in Test Mode 1.

Table 8. Electrical Specifications Measured in Test Mode 1

Measurement	Lower Limit	Max Limit
Transmitter Output Voltage		
1.0 V p-p Mode	0.85 V p-p	1.05 V p-p
2.4 V p-p Mode	2.04 V p-p	2.52 V p-p
Transmitter Clock Frequency	7,499,625 Bd	7,500,375 Bd
Transmitter Timing Jitter	N/A[1]	10 ns

[1] N/A means not applicable.

Figure 5 shows the typical waveform differentially measured across the MDI port when the ADIN1100/ADIN1110/ADIN2111 are configured for PMA Test Mode 1 and 1.0 V p-p transmit level.

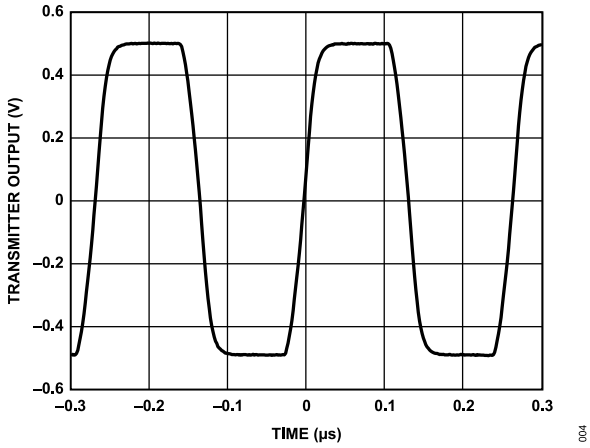

Figure 5. Differential Voltage Across MDI Port in Test Mode 1, 1.0 V p-p Transmit Level

PMA TEST MODE 2

Similar to Test Mode 1, the signal in Test Mode 2 measured across the MDI looks like a square waveform with the same nominal amplitude (1.0 V p-p or 2.4 V p-p), but with a frequency of approximately 375 kHz. Different factors, including the circuitry around the MDI port, surge protection, power coupling inductors, and galvanic isolation transformer, affect the high and low states of the waveform. One of the effects is often seen as decay on the voltage level during the high and low times of the waveform. The measurement of this decay is called transmitter output droop and is specified in Subclause 146.5.4.2 of the IEEE 802.3cg-2019 standard. Test

Mode 2 is used to perform the voltage droop measurement, which is defined as follows:

$$Droop\,\% = 100\,\% \times \frac{V_{133.3} - V_{800}}{V_{133.3}} \quad (6)$$

where:
$V_{133.3}$ and V_{800} are the voltages measured 133.3 ns and 800 ns, respectively, after the zero crossing of the Test Mode 2 waveform. *Droop* is defined both for the positive (positive droop) and negative (negative droop) portions of each cycle of the waveform, and both must be below 10%.

Note that this voltage droop measurement requires a 100 Ω load termination at the MDI port. Figure 6 shows the waveform corresponding to the ADIN1100/ADIN1110/ADIN2111 in Test Mode 2 in the 1.0 V p-p transmit level, where the $V_{133.3}$ and V_{800} points corresponding to the positive droop measurement are indicated.

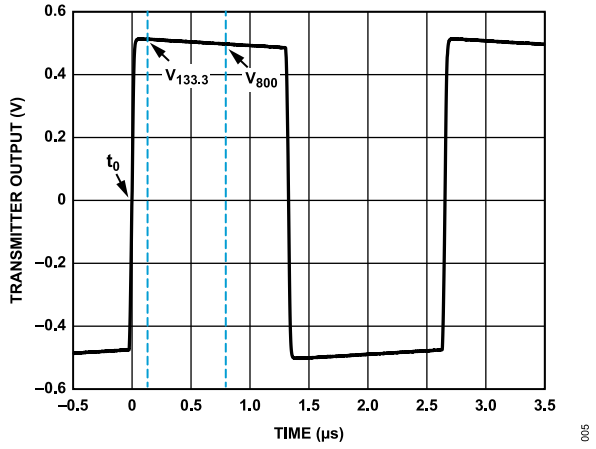

Figure 6. Differential Voltage Across MDI Port in Test Mode 2, 1.0 V p-p Transmit Level

The transmitter output droop specification has been amended in the IEEE 802.3dd-2022 standard to increase the output droop limit from 10% to 25% but only when the MDI circuitry includes a Clause 104 Type E power sourcing equipment (PSE) or powered device (PD). In this case, the measurement points become 400 ns and 1066.7 ns, respectively, after the zero crossing, instead of 133.3 ns and 800 ns as initially stated in the IEEE 802.3cg-2019 standard.

PMA TEST MODE 3

The waveform corresponding to PMA Test Mode 3 looks different compared to the waveforms of PMA Test Mode 1 and PMA Test Mode 2 because PMA Test Mode 3 forces the transmission of idle signals. PMA Test Mode 3 is used to measure PSD and power level. This test mode also requires a 100 Ω termination at the MDI port. Figure 7 shows a short snapshot of the waveform corresponding to this test mode whereas Figure 8 and Figure 9 show the PSD measured on the ADIN1100/ADIN1110/ADIN2111 in PMA Test Mode 3 for the 1.0 V p-p and 2.4 V p-p transmit levels, respectively.

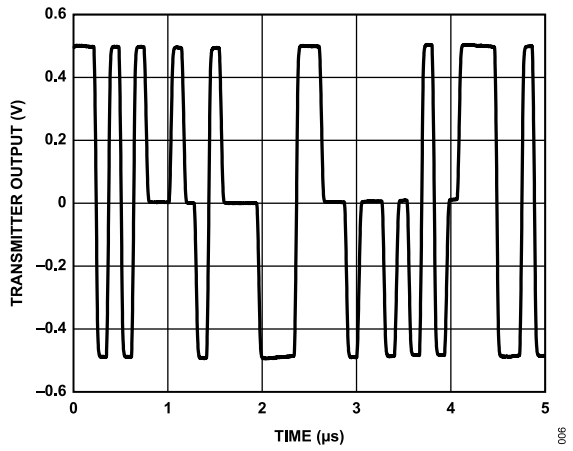

Figure 7. Differential Voltage Across MDI Port in Test Mode 3, 1.0 V p-p Transmit Level

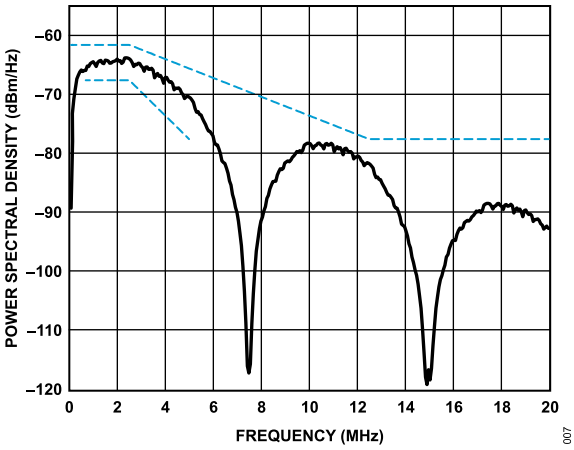

Figure 8. Power Spectral Density Measured in Test Mode 3, 1.0 V p-p Transmit Level and Limit Curves Defined by the IEEE 802.3cg-2019 Standard

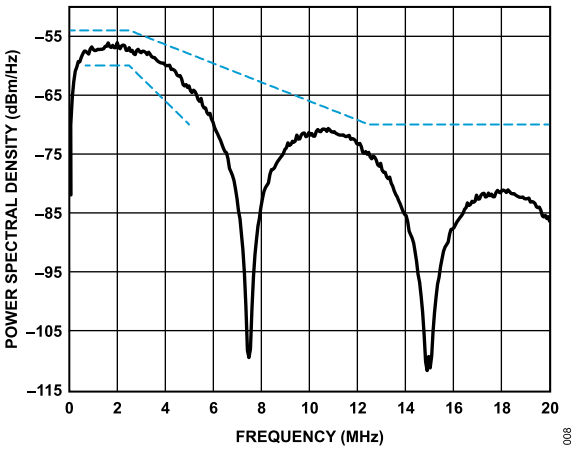

Figure 9. Power Spectral Density Measured in Test Mode 3, 2.4 V p-p Transmit Level and Limit Curves Defined by the IEEE 802.3cg-2019 Standard

PMA TEST MODE 1 TO PMA TEST MODE 3 CONFIGURATION

To configure the PMA test modes, use the following register write steps:

1. Write a 1 to CRSM_SFT_PD to enter software power-down mode.
2. Wait for CRSM_SFT_PD_RDY to be asserted to check if the device has entered software power-down mode.
3. Write a 0 to the AN_EN bit to disable autonegotiation.
4. Write a 1 to the AN_FRC_MODE_EN bit to set autonegotiation forced mode.
5. Write the appropriate value to B10L_TX_TEST_MODE to select the desired test mode. Table 9 contains the values that must be written for each test mode.
6. Write a 0 to CRSM_SFT_PD to exit software power-down mode.

Table 9. PMA Test Modes Configuration

PMA Test Mode	B10L_TEST_MODE
Test Mode 1	3'b001
Test Mode 2	3'b010
Test Mode 3	3'b011

TRANSMIT DISABLE MODE

The transmit disable mode implements Subclause 45.2.1.186a.2 of the IEEE 802.3cg-2019 standard. This mode keeps both transmit and receive paths in normal operation of the PHY. However, this mode transmits 0 symbols only. This mode can be used to measure the MDI return loss specified in Subclause 146.8.3 of the IEEE 802.3cg-2019 standard, which establishes that the MDI return loss must meet the limits shown in Table 10.

Table 10. MDI Return Loss Specifications Based on IEEE 802.3cg-2019 Standards

Frequency Range	Limit in dB
0.1 ≤ f < 0.2 MHz	Use Equation 7
0.2 ≤ f ≤ 1 MHz	20
1 < f ≤ 10 MHz	Use Equation 8
10 < f ≤ 20 MHz	Use Equation 9

$$20 - 18 \times \log_{10}\left(\frac{0.2}{f}\right) \quad (7)$$

$$20 - 16.7 \times \log_{10}(f) \quad (8)$$

$$3.3 - 7.6 \times \log_{10}\left(\frac{f}{10}\right) \quad (9)$$

where *f* is the frequency in MHz.

Similar to the transmitter output droop, the MDI return loss equations were amended in the IEEE 802.3dd-2022 standard. In the cases where a Clause 104 Type E PSE or PD is part of the MDI circuitry, the new equations must meet the limits shown in Table 11.

Table 11. MDI Return Loss Specifications Based on IEEE 802.3dd-2022 Standards

Frequency Range	Limit in dB
0.1 ≤ f < 0.5 MHz	Use Equation 10
0.5 ≤ f ≤ 1 MHz	20
1 < f ≤ 10 MHz	Use Equation 8
10 < f ≤ 20 MHz	Use Equation 9

$$20 - 21.5 \times \log_{10}\left(\frac{0.5}{f}\right) \quad (10)$$

where *f* is the frequency in MHz.

If no PSE or PD is part of the MDI circuitry, the original equations in Table 10 remain the same.

To enable the transmit disable mode, follow these steps:

1. Write a 1 to CRSM_SFT_PD to enter software power-down mode.
2. Wait for CRSM_SFT_PD_RDY to be asserted to check if the device has entered software power-down mode.
3. Write a 0 to the AN_EN bit to disable autonegotiation.
4. Write a 1 to the AN_FRC_MODE_EN bit to set autonegotiation forced mode.
5. Set B10L_TX_DIS_MODE_EN = 1 to enable transmit disable mode.
6. Write a 0 to CRSM_SFT_PD to exit software power-down mode.

LOOPBACK MODES

The ADIN1100/ADIN1110/ADIN2111 provide several loopback configurations that facilitate the debug and verification of the local PHY, remote PHY, and cable link. The available functions are PMA loopback, physical coding sublayer (PCS) loopback, MAC interface loopback, and MAC interface remote loopback. An external MII/RMII/RGMII loopback can also be configured on the ADIN1100. However, they cannot be configured on the ADIN1110/ADIN2111.

For all loopback modes, first enter software power-down mode through the following steps:

1. Write a 1 to CRSM_SFT_PD to enter software power-down mode.
2. Wait for CRSM_SFT_PD_RDY to be asserted to check if the device has entered software power-down mode.

After entering software power-down mode, follow the configuration steps described in each loopback. The registers referenced in this section are shown in Table 12.

Table 12. Registers of Interest to Loopback Modes

Name	Device Address	Register Address	Bit
CRSM_SFT_PD	0x1E	0x8812	0
CRSM_SFT_PD_RDY	0x1E	0x8818	1
AN_EN	0x07	0x0200	12
AN_FRC_MODE_EN	0x07	0x8000	0
B10L_LB_PMA_LOC_EN	0x01	0x08F6	0
AN_LINK_STATUS	0x07	0x0201	2
B10L_LB_PCS_EN	0x03	0x08E6	14
MAC_IF_LB_EN	0x1F	0x8055	0
MAC_IF_REM_LB_EN	0x1F	0x8055	2
RMII_TXD_CHK_EN	0x1F	0x8050	0
MAC_IF_REM_LB_RX_SUP_EN	0x1F	0x8055	3

PMA LOCAL LOOPBACK

The PMA local loopback is intended as an implementation of the IEEE 802.3cg-2019 standard, Subclause 146.5.6 PMA local loopback. The PMA local loopback function is useful to perform a full test and verification of the PMA transmit function, PMA receive function, and PCS. To use this loopback, leave the MDI port open, with no cable nor load attached. The PHY receive function then uses the echo reflections of the transmission, decodes the signals, and passes the decoded data to the MAC interface. The transmitted packets must match the received packets if the PHY is working properly. The verification can be performed by an external MAC client that sends, receives, and compares the packets or by the on-chip frame generator and frame checker in the ADIN1100/ADIN1110/ADIN2111.

To use the PMA local loopback, follow these configuration steps:

1. Enter software power-down mode as described in the Loopback Modes section.
2. Write a 0 to the AN_EN bit in the AN_CONTROL register to disable autonegotiation.
3. Write a 1 to the AN_FRC_MODE_EN bit to set autonegotiation forced mode.
4. Set the B10L_LB_PMA_LOC_EN bit to 1 in the B10L_PMA_CNTRL register to enable the PMA local loopback.
5. Write a 0 to CRSM_SFT_PD to exit software power-down mode.
6. Poll the link status bit, AN_LINK_STATUS. This bit must read as a 1, indicating that a link is up.

If using the on-chip frame checker, set FC_TX_SEL = 0 to configure the on-chip frame checker to analyze the data coming from the PHY. This setting is the default configuration.

When using this loopback, the MSE can be used and compared to the values expected from a good link, which can be determined by link quality monitoring. Refer to the Link Quality Monitoring section.

PCS LOOPBACK

The PCS loopback function loops the receive data back to the transmit data within the PCS of the PHY. When this function is enabled, no data is transmitted to the MDI port. As opposed to the PMA local loopback, this function can be used when the system is deployed without requiring the disconnection of the cable from the MDI.

To configure this function, follow these steps:

1. Enter software power-down mode.
2. Write a 0 to the AN_EN bit in the AN_CONTROL register to disable autonegotiation.
3. Write a 1 to the AN_FRC_MODE_EN bit to set autonegotiation forced mode.
4. Set the B10L_LB_PCS_EN bit to 1 in the B10L_PCS_CNTRL register to enable the PCS loopback.
5. Write a 0 to CRSM_SFT_PD to exit software power-down mode.
6. Poll the link status bit, AN_LINK_STATUS. This bit must read as a 1, indicating that a link is up.

Similar to the PMA loopback mode, the transmitted packets must match the received packets, and an external MAC client, or the on-chip frame generator and frame checker, can be used to generate and check the same (see the Frame Generator and Checker section for details).

MAC INTERFACE LOOPBACK

The MAC interface loopback function loops the received data on the MAC interface TXD pins back to the RXD pins. Thus, this function can be used to check the physical connectivity from the external MAC client to the PHY. If there is no issue on the physical interconnections, the transmitted packets match the received packets.

To configure this function, follow these steps:

1. Enter software power-down mode.
2. Set the MAC_IF_LB_EN bit to 1 in the MAC_IF_LOOPBACK register to enable the MAC interface loopback.

3. Write a 0 to CRSM_SFT_PD to exit software power-down mode.

An external MAC client can be used to transmit and receive packets, and verify that these packets match.

MAC INTERFACE REMOTE LOOPBACK

The MAC interface remote loopback can be used when there is a link up with another PHY to loop the data received from one PHY back to it. This linking allows a remote PHY to verify a complete link, ensuring that the PHY receives the proper data. Note that if the MAC_IF_REM_LB_RX_SUP_EN bit within the same register is set, which is its default state, then the data received by the PHY is suppressed and not sent to the MAC interface.

To configure this function, follow these steps:

1. Enter software power-down mode.
2. Set the MAC_IF_REM_LB_EN bit to 1 in the MAC_IF_LOOPBACK register.
3. When using the RMII interface, set the RMII_TXD_CHK_EN bit to 1.
4. Write a 0 to CRSM_SFT_PD to exit software power down mode.

Figure 11 shows a common use of this loopback where PHY 2 has been configured in this function while the frame generator has been enabled on PHY 1. The generated frames are then transmitted through the single twisted pair to PHY 2, which decodes the signals and passes the data to its MAC interface, which is configured to loop the packets back to the digital and analog blocks and subsequently transmitted through the single twisted pair cable back to PHY 1. The frame checkers are analyzing the received packets in both ends of the link. If the link has no issues, the frame checker must find no errors.

This loopback can also be used in a single PHY without a link partner by configuring a PMA or PCS loopback at the same time, which can be useful to self test the whole PHY.

EXTERNAL MII/RMII/RGMII LOOPBACK

The external MAC interface loopback is only possible on the ADIN1100 and not on the ADIN1110/ADIN2111. This function is equivalent in functionality to the MAC interface remote loopback explained in the MAC Interface Remote Loopback section. Although the MAC interface remote loopback interconnects internally between the corresponding interface pins through the configuration bit setting, the external MAC interface loopback requires those connections to be physically made externally as shown in Table 13.

Table 13. MII, RMII, and RGMII External Loopback Interconnections

MII[1]	RMII	RGMII
TXD_0 to RXD_0	TXD_0 to RXD_0	TXD_0 to RXD_0
TXD_1 to RXD_1	TXD_1 to RXD_1	TXD_1 to RXD_1
TXD_2 to RXD_2	N/A[2]	TXD_2 to RXD_2
TXD_3 to RXD_3	N/A[2]	TXD_3 to RXD_3
TX_EN to RX_DV	TX_EN – CRS_DV[3]	TX_CTL to RX_CTL[4]
TX_ER to RX_ER	N/A[2]	TXC to RXC[5]

[1] When using the MII interface, do not connect RX_CLK to TX_CLK. This action causes malfunctioning of the device.

[2] N/A means not applicable.

[3] In RMII mode, CRS_DV is physically the same pin as RX_DV when setting RMII_TXD_CHK_EN = 1.

[4] In RGMII mode, TX_CTL and RX_CTL are physically the same pins as TX_EN and RX_DV, respectively.

[5] In RGMII mode, TXC and RXC are physically the same pins as TX_CLK and RX_CLK, respectively.

To use this loopback function, follow these steps:

1. Enter software power-down mode.
2. If using RMII loopback, assert the RMII_TXD_CHK_EN bit.
3. Physically interconnect the corresponding pins as shown in Figure 10.
4. Write a 0 to CRSM_SFT_PD to exit software power-down mode.

Similar to the MAC interface remote loopback, Figure 11 shows a common use case for this function. The frames generated using the frame generator on PHY 1 are sent through the link and looped back at the MAC interface of PHY 2.

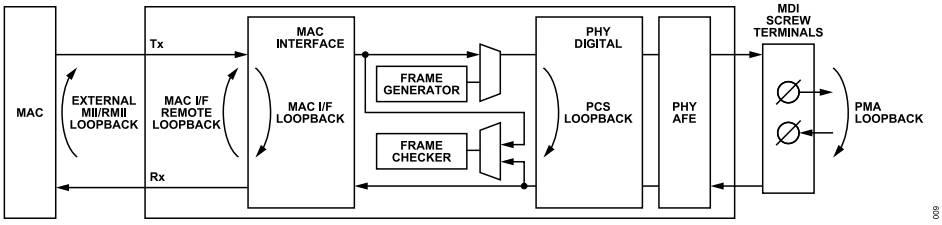

Figure 10. Loopback Modes Available on the ADIN1100

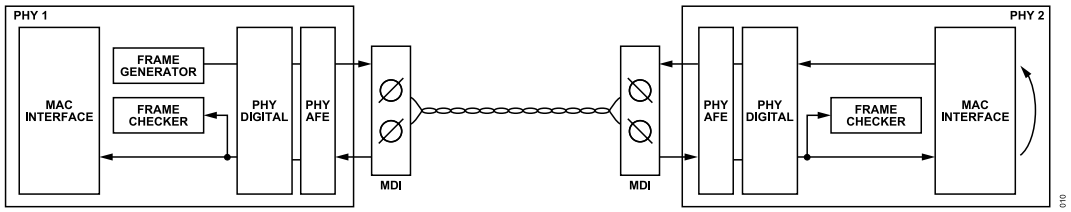

Figure 11. MAC Interface Remote Loopback and External Loopback Common Use Case

FRAME GENERATOR AND CHECKER

The ADIN1100/ADIN1110/ADIN2111 can be configured to generate frames and to check received frames. The frame generator and checker can be used independently or simultaneously. The Frame Checker section and the Frame Generator section explain the basic functionality and configuration of both features.

FRAME GENERATOR

When the frame generator is enabled, the source of data for the PHY comes from the frame generator and not from the MAC interface. The frame generator control registers configure the type of frames to be sent, frame length, gap between frames, and the number of frames to be generated.

To enable and start the frame generator, set FG_EN = 1. When the generation of the frames is completed, the frame generator done bit is set (FG_DONE).

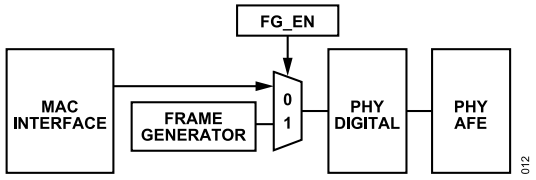

Figure 12. Frame Generator Selection

The details related to the registers referenced in this section can be found in Table 14.

Table 14. Frame Generator Basic Configuration Registers

Name	Device Address	Register Address	Bits
FG_EN	0x1F	0x8020	0
FG_RSTRT	0x1F	0x8021	3
FG_CNTRL	0x1F	0x8021	2:0
FG_CONT_MODE_EN	0x1F	0x8022	0
FG_FRM_LEN	0x1F	0x8025	15:0
FG_IFG_LEN	0x1F	0x8026	15:0
FG_NFRM_H	0x1F	0x8027	15:0
FG_NFRM_L	0x1F	0x8028	15:0
FG_DONE	0x1F	0x8029	0

Frame Generator Behavior If the Link Drops

The frame generator is designed to operate only when there is a link with another PHY or when a PCS or a PMA loopback has been configured. If the frame generator is used to send frames to another PHY and the link is interrupted, the frame generation is immediately interrupted. Then, if the link recovers and the configuration has not changed, the frame generator starts over the transmission with the previous configuration.

Frame Length

To set the frame length, write the desired frame length in bytes to FG_FRM_LEN. Given that the frame generator produces proper Ethernet frames, add 18 to the value of FG_FRM_LEN (6 bytes for source address, 6 bytes for destination address, 2 bytes for the length field, and 4 bytes for the FCS field) to determine the frame length in bytes. The minimum Ethernet frame length is 64 bytes. Thus, FG_FRM_LEN must not be lower than 64. However, if FG_FRM_LEN is set to a number lower than 64, the frame generator uses the specified value and does not implement any padding.

Interframe Gap

The on-chip frame generator allows varying the gap to be inserted between frames. To change the gap, write the desired gap in bytes to the FG_IFG_LEN register.

Burst Mode

When in burst mode, the higher and the lower bytes (FG_NFRM_H and FG_NFRM_L, respectively) define the number of frames to be generated. In other words, these bytes must be set as follows:

$$FG\_NFRM\_H = (Nfrms >> 16) \;\&\; 0xFFFF \quad (11)$$

$$FG\_NFRM\_L = Nfrms \;\&\; 0xFFFF \quad (12)$$

where *Nfrms* is the desired number of frames to be generated.

After the frames have been generated, FG_DONE is asserted. FG_DONE is cleared upon reading. To restart the frame generator, assert FG_RSTRT, which causes the frame generator to start a new burst. If the frame generator is restarted before the current frame generation is complete, the frame generation is interrupted.

Continuous Mode

To configure the frame generator to operate in continuous mode, write a 1 to FG_CONT_MODE_EN. In this mode, the frame generator keeps generating frames indefinitely.

Frame Payload

To configure the frame payload of the on-chip frame generator, write the appropriate value to FG_CNTRL according to Table 15.

Table 15. Frame Payload Configuration

FG_CNTRL Value	Description
3'b000	No frames after current frame
3'b001	Random payload (default)
3'b010	Payload of 0x00 repeated
3'b011	Payload of 0xFF repeated
3'b100	Payload of 0x55 repeated
3'b101	Payload in decrementing bytes

Frame Generator Configuration Sequence

To configure this function in burst mode, follow these steps:

1. Write a 0 to both FG_NFRM_H and FG_NFRM_L to set the number of frames to zero.
2. Write a 1 to FG_EN to enable the frame generator.
3. Set FG_FRM_LEN to configure the desired frame length in bytes.

4. Write the value in bytes to FG_IFG_LEN to configure the interpacket gap.
5. Set FG_NFRM_H and FG_NFRM_L to define the number of frames to be generated.
6. Assert FG_RSTRT to restart the frame generator.
7. Wait for FG_DONE to be asserted, which indicates that the frame generation is complete.

To run the frame generator again with the same configuration, assert FG_RSTRT again. If the frame generator is to be reconfigured, repeat Step 3 through Step 7 with the new configuration.

To configure the frame generator in continuous mode, follow these steps:

1. Write a 0 to both FG_NFRM_H and FG_NFRM_L to set the number of frames to zero.
2. Write a 1 to FG_EN to enable the frame generator.
3. Write FG_CONT_MODE_EN = 1 to enable continuous mode.
4. Set FG_FRM_LEN to configure the desired frame length in bytes.
5. Write the value in bytes to FG_IFG_LEN to configure the interframe gap.
6. Assert FG_RSTRT to restart the frame generator.

The frame generator continues to indefinitely transmit frames.

FRAME CHECKER

The frame checker is enabled by default (frame checker enable bit, FC_EN = 1) and can be configured to check and analyze received frames from either the MAC interface or the PHY interface (refer to Figure 13). The frame checker reports the number of frames received, frames with symbol or CRC errors, undersized or oversized frames, and frames with an odd number of nibbles. Table 17 contains the complete list of relevant counters.

The basic operation of the frame checker only requires the following two steps:

1. Set FC_EN = 1 (which is 1 by default) to enable the frame checker.
2. Select the data source through FC_TX_SEL.

Table 16. Frame Checker Basic Configuration Registers

Name	Device Address	Register Address	Bits
FC_EN	0x1F	0x8001	0
FC_TX_SEL	0x1F	0x8005	0

Frame Checker Source Selection

The frame checker analyzes the source data, which can be chosen between the frames from the PHY interface or frames from MAC interface. To select the source, write the appropriate value to FC_TX_SEL. The FC_TX_SEL register default value is 0, which selects the frames from the PHY interface. Set to 1 for observing frames coming from the MAC interface.

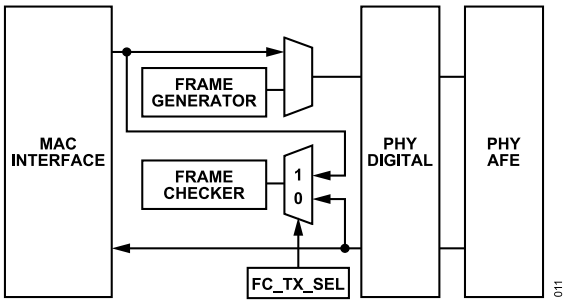

Figure 13. Frame Checker Source

Frame Counters Synchronization

To ensure synchronization between the frame checker error counters and frame checker counters, all the counters are latched when the receive error counter register is read (RX_ERR_CNT). Therefore, when using the frame checker, read RX_ERR_CNT first, and then read all other frame counters and error counters. All internal counters are cleared when RX_ERR_CNT is read, in such a way that no events or errors are missed.

Frame Checker Configuration Sequence

To configure the frame checker, follow these steps:

1. Assert FC_EN to enable frame checker. The frame checker is enabled by default.
2. Write the appropriate value to FC_TX_SEL to select the source to analyze.
3. If needed, read RX_ERR_CNT to clear all counters before starting the frame generation.
4. Read RX_ERR_CNT to latch all counters and read the rest of the counters of interest according to Table 17.

Note that the total number of received frames with no errors can be obtained as follows:

$$Received\ \ frames = (FC\_FRM\_CNT\_H << 16) + FC\_FRM\_CNT\_L \quad (13)$$

Table 17. Frame Checker Counter Registers

Register Name	Device Address	Register Address	Description
RX_ERR_CNT	0x1F	0x8008	Receive error counter.
FC_FRM_CNT_H	0x1F	0x8009	Bits 31:16 of the frame checker count.
FC_FRM_CNT_L	0x1F	0x800A	Bits 15:0 of the frame checker count.
FC_LEN_ERR_CNT	0x1F	0x800B	Frame checker length error count.
FC_ALGN_ERR_CNT	0x1F	0x800C	Frame checker alignment error count.
FC_SYMB_ERR_CNT	0x1F	0x800D	Frame checker symbol error count.
FC_OSZ_CNT	0x1F	0x800E	Frame checker oversized frame error count.

Table 17. Frame Checker Counter Registers (Continued)

Register Name	Device Address	Register Address	Description
FC_USZ_CNT	0x1F	0x800F	Frame checker undersized frame error count.
FC_ODD_CNT	0x1F	0x8010	Frame checker odd nibble frame count.
FC_ODD_PRE_CNT	0x1F	0x8011	Frame checker odd preamble packet counter.
FC_FALSE_CARRIER_CNT	0x1F	0x8013	Frame checker false carrier count register.