Application Note

Time Domain Reflectometry with DP83826

ABSTRACT

This application note outlines how Time Domain Reflectometry helps solve various kinds of cable fault challenges of Ethernet-based communication systems. The application note describes how to use the TDR feature of the DP83826 to implement the cable diagnostics feature in the system.

Table of Contents

List of Figures

List of Tables

Trademarks

1 Time Domain Reflectometry

Time Domain Reflectometry (TDR) only works for twisted pair connections. TDR involves sending a pulse on TX and RX pair and observing results on either pair. By measuring voltage amplitude, polarity, and the time interval, the PHY can determine the nature and position of the fault. The DP83826 TDR generator sends pulse on the TX and RX channel, then monitors both channels to observe reflections. It sends a pulse one channel at a time, and if reflections are observed on the other channel, then the PHY TDR realizes that the wires have been crossed. The DP83826 can detect one peak for each transmit and receive channel. TDR can be used for the following:

- Cable open
- Cable short
- Impedance discontinuity

TDR can only be used when the link is down.

1.1 Example Connections

The following sections are example connections where TDR can be used.

1.1.1 Open Circuit Cable

Open cable is easy to diagnose since it generates a strong reflection. No reflection is observed on the other channel. The reflection due to the open circuit is in-phase with the transmitted pulse (positive polarity). Any kind of inductive impedance discontinuity generates in-phase reflection and the amplitude depends on the amount of impedance discontinuity.

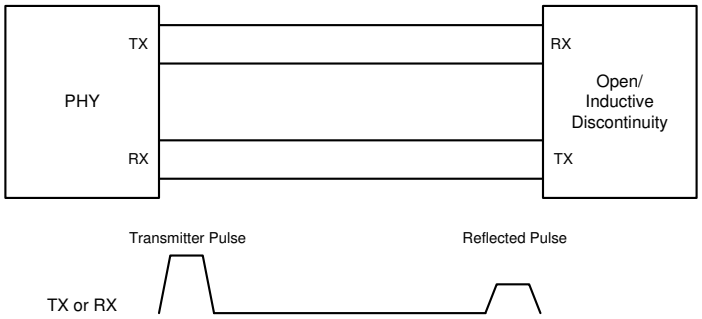


Figure 1-1. Open Circuit Cable

1.1.2 Short Circuit Cable

Short-circuited cables also generate a strong reflection, but this reflection is out-of-phase with the original pulse (negative polarity). Any kind of capacitive impedance discontinuity generates out-of-phase reflection. The amplitude depends on the amount of impedance discontinuity.

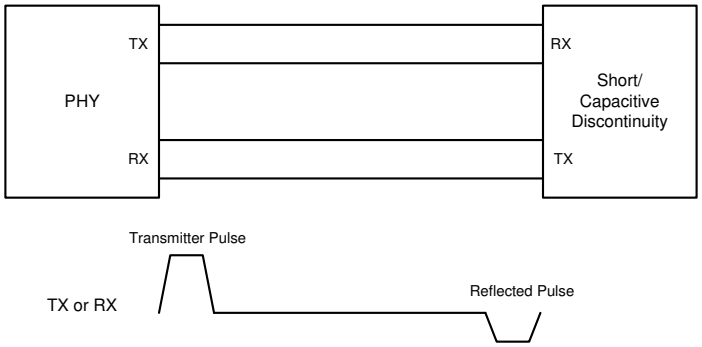


Figure 1-2. Short Circuit Cable

2 TDR Implementation

2.1 DP83826 TDR Configuration

The DP83826 can support 150 meter and above cable reach. Long cable reach causes the transmitted signal to face higher attenuation with cable length, so the gain and other settings need to be tweaked with cable length to achieve reliable fault detection using TDR. The cable is assumed to be divided into the following five segments:

- Segment 1: 0 m to 10 m
- Segment 2: 10 m to 20 m
- Segment 3: 20 m to 40 m
- Segment 4: 40 m to 80 m
- Segment 5: 80 m to 190 m

In order to a run complete TDR simulation, data from all five segments must be collected. Each segment requires a different set of register writes and segment 1 requires two register writes.

A fault can only be detected by the PHY's TDR if a fault (open or short) is introduced to the circuit. Follow these steps to run TDR:

1. Configure registers 0x170, 0x173, 0x175, 0x178, 0x416, 0x411, 0x456, and 0x1E to the corresponding values for Segment 1A shown in the Table 2-1. The registers used for configuration are *extended* registers. To configure extended registers, see the *DP83826 Deterministic, Low-Latency, Low-Power, 10/100 Mbps, Industrial Ethernet PHY*, data sheet for more information.
2. Confirm that TDR has finished running for that segment by checking 0x001E[1] = 1.
3. Read registers 0x180, 0x185, and 0x18A and fill out the corresponding row of the Table 2-2. The order of the rows of the table does not correspond to the order of the segments.
4. Repeat Step 1, 2 and 3 for segments 1B, 2, 3, 4, and 5 in that order. You may choose to only run TDR for certain segments. For instance, if your cable is 20m, you may fill out the rows for Segment 4 and 5 with zeros.
5. Process the data collected using the TDR Algorithm described in the next section to determine the fault location and type.

Table 2-1. Register Configurations for Each Segment

Register	Value					
	Segment 1A	Segment 1B	Segment 2	Segment 3	Segment 4	Segment 5
0x170	0x5C12	0x5C12	0x5C22	0x5E32	0x5E42	0x5E52
0x173	0xD00	0x0D0C	0x0D13	0x1A20	0x343A	0x8F6E
0x175	0x1007	0x1007	0x1007	0x1007	0x100A	0x100D
0x178	0x002	0x0002	0x0002	0x0002	0x0002	0x0006
0x416	0x1FA0	0x1FA0	0x1FA0	0x1FA0	0x1FA0	0x1F90
0x411	0x0813	0x0813	0x0813	0x0815	0x0816	0x816
0x456	0x0608	0x0608	0x0608	0x0608	0x0608	0x0608
0x01E	0x8102	0x8102	0x8102	0x8102	0x8102	0x8102

Table 2-2. TDR Data Table

Segment	peak_index (0x180)	peak_value (0x185)	peak_sign (0x18A)
2			
3			
4			
5			
1B			
1A			

2.2 TDR Algorithm

This section describes how to process the TDR register data. All of the following steps must be carried out in code.

1. Define 6x3 matrix from the table you entered in the section DP83826 TDR configuration. Then, define each column as an array and name them peak_index, peak_value, and peak_sign.
2. Initialize the following variables:

Table 2-3. Initial Variables

Variable	Type	Initial Value	Description
i	int	5	Current row of the matrix
threshold	int	10	The threshold that peak_indx must exceed for a fault to be detected
fault_detected	bool	FALSE	Defines whether a fault has been detected
fault_location	float	0	Fault location
prop_dly	float		Propogation delay variable. This value can be adjusted based on the cable type. See the Prop_dly Values table for suggested values for different cable types.
offset	int	5	Variable used in calculating the fault location.
fault	int	0	Defines whether a fault is a SHORT (1) or an OPEN (0) fault

Table 2-4. Prop_dly Values

Cable type	Prop_dly
Cat5/5e	5.35
Cat6	5
Cat7	4.6

3. Write a program with the following algorithm. See the TDR Algorithm Example Flow and MATLAB example code below for reference.

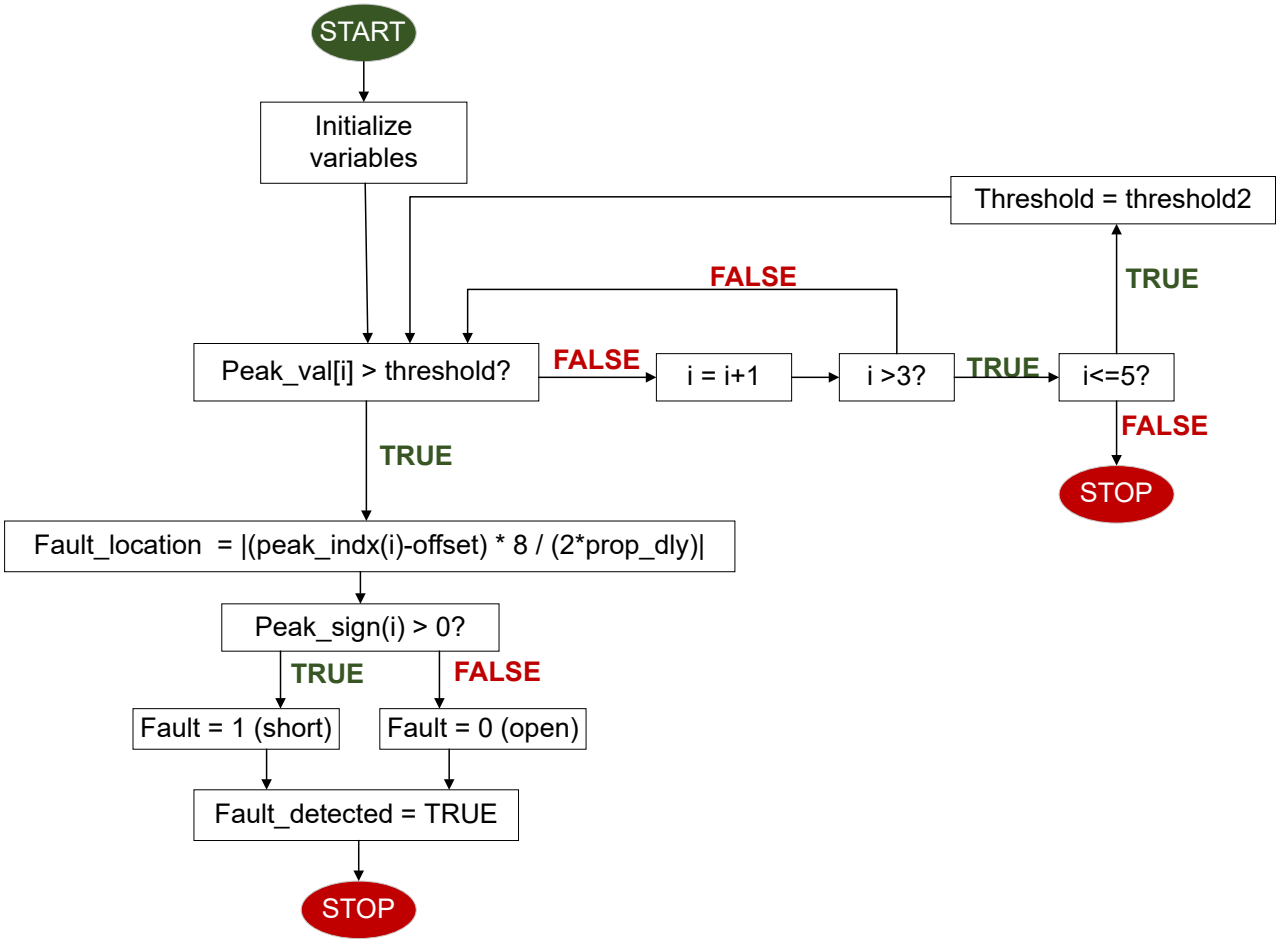


Figure 2-1. DP83826 TDR Algorithm

4. The fault type is in the *Fault* variable and the fault location is stored in fault_location.

2.2.1 TDR Algorithm Example Flow

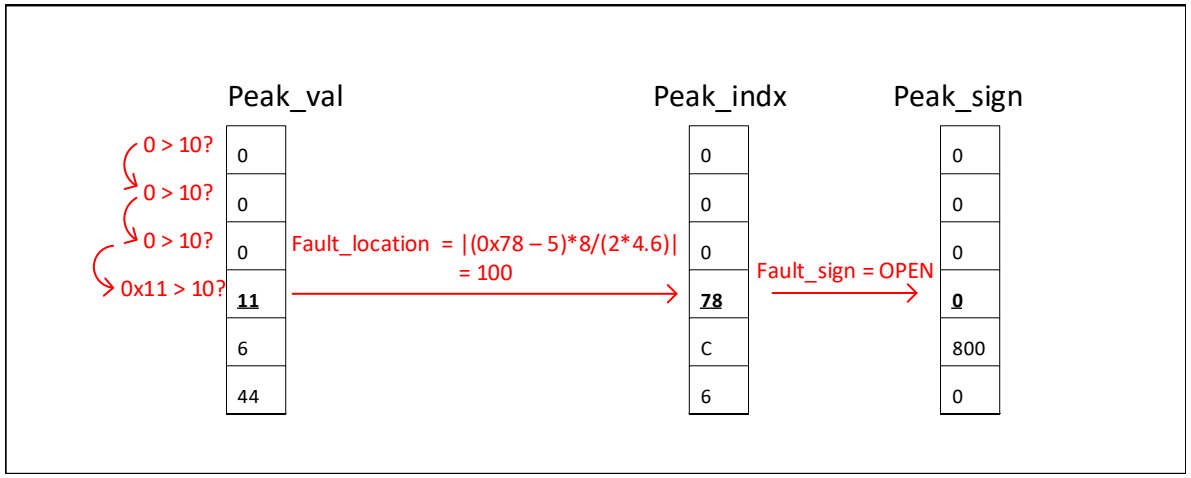
Table 2-5 shows the TDR results from a 100 m CAT7 cable.

Table 2-5. TDR Data Table Example

Segment	peak_index (0x180)	peak_value (0x185)	peak_sign (0x18A)
2	0	0	0
3	0	0	0
4	0	0	0
5	78	11	0
1B	C	6	800
1A	6	44	0

The algorithm starts by checking to see if the first element the array *peak_val* is greater than the current threshold, 10. If it is not greater than 10, it moves onto the next row. If it is greater, the algorithm will look for the corresponding row in the array peak_indx and use that value to calculate the location of the fault. Then, the algorithm will look at the corresponding row of peak_sign to determine the fault type. A *0* means the fault is open, a nonzero value means that the fault is a short.

Figure 2-2. DP83826 TDR Example Flowchart



2.2.2 TDR Algorithm Matlab Example Code

The following code is an example of how to implement the TDR algorithm in Matlab. The input to the program is the 6x3 matrix.

```
function [tdr_results] = tdr_826(input_matrix)


%segment = tmp(1:4:end);
peak_indx = tmp(1:3:end)
peak_indx %first column in 6x3 matrix
peak_val = tmp(2:3:end)
peak_val %second column
peak_sign = tmp(3:3:end)
peak_sign %third column


thr = 10;
prop_dly = 5.2;
offset = 5;



flt_found = 0;
flt_loc = 0;
flt_sign = 0;



%% Process the TDR data from Segment 2 onwards
for jj = 0:3
   if peak_val(jj+1) > thr
     flt_loc = ((peak_indx(jj+1) - offset)*8)/(2*prop_dly);
     if peak_sign(jj+1) > 0
       flt_sign = 1;
     else
       flt_sign = 0;
     end
     flt_found = 1;
     break;
   end
end

%% Process the TDR data for Segment 1..
thr_seg1_2 = 24;

if flt_found == 0
  if peak_val(5) > thr_seg1_2
    flt_loc = ((peak_indx(2) - offset)*8)/(2*prop_dly);
    if peak_sign(2) > 0
      flt_sign = 1;
    else
      flt_sign = 0;
    end
    flt_found = 1;
  end
end

thr_seg1_1 = 24;

if flt_found == 0
  if peak_val(6) > thr_seg1_1
    flt_loc = ((peak_indx(1) - offset)*8)/(2*prop_dly);
    if peak_sign(1) > 0
      flt_sign = 1;
    else
      flt_sign = 0;
    end
    flt_found = 1;
  end
end



%% Print the Results..
if flt_found == 1
  fprintf('\n');
  if flt_sign == 0
```

```
    fprintf('Fault location = %6.2f; Fault = Open\n',flt_loc);
  else
    fprintf('Fault location = %6.2f; Fault = Short\n',flt_loc);
  end
else
  fprintf('\n');
  fprintf('No Fault found\n');

end


tdr_results.flt_loc = flt_loc;
tdr_results.flt_sign = flt_sign;

return

end
```

3 Summary

This application note explains the basics of TDR and how to use the TDR functionality of the DP83826 Industrial Ethernet PHY.

4 References

- Texas Instruments, *DP83826 Deterministic, Low-Latency, Low-Power, 10/100 Mbps, Industrial Ethernet PHY*, data sheet.
- Texas Instruments, *How to use the TDR Feature of DP83822*, application note
- Texas Instruments, *Solving Cable Faults Challenges with TI Ethernet PHYs*, application note.

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