DP83815 MacPHYTER™ and DP83816 MacPHYTER - II™ High Data Rate Stress Testing

National Semiconductor
Application Note 1287
June 2003

1.0 Scope

National Semiconductor® performs extensive stress testing on its 10/100 Mb/s Ethernet products. These tests include the Microsoft Windows Hardware Quality Labs (WHQL) Hardware Compatibility Test (HCT) for Network LAN devices to ensure proper operation with Microsoft Windows™ operating systems.

The HCT test has revealed a receive packet corruption scenario. Although this scenario has not been reproduced under other high data rate stress tests, or actual network conditions, in theory, this could result in a failure event in an application that involves high data rate transfers.

This Application Note describes this scenario, and recommends software workarounds that minimize any possible effect on overall system operation.

The National Windows driver set already implements a workaround, hence this document is for descriptive purposes only.

2.0 Introduction

To ensure hardware compatibility with Microsoft operating systems, Windows Hardware Quality Labs (WHQL) produce and support Hardware Compatibility Test (HCT) kits. The HCT kit includes documentation and tests that hardware manufacturers can use to test products and drivers for compatibility and inter-operability with Windows operating systems, and to qualify drivers for digital signature by Microsoft.

National uses the Microsoft Windows XP HCT 10.0 kit to test its DP83815 MacPHYTER™ and DP83816 MacPHYTER-II™ 10/100 Mb/s Ethernet controllers, and their drivers.

The event described in this document occurs during Microsoft's WinXP HCT 10.0 kit stress tests. A description of one of these tests, the NDIS (Network Driver Interface Specification) stress test, follows.

2.1 NDIS STRESS TEST

The NDIS stress test creates a significant amount of network traffic by sending and receiving packets of various sizes, types and content.

For NDIS testing, a typical configuration consists of a client system and at least one server system. The client system will contain the network device being tested, also referred to as the test device.

Stress testing can take place in one of three modes:

Client Mode
The client system is responsible for controlling the test and sending test packets. At the same time, the client system keeps track of the number and type of packets sent to each of the servers participating in the test.

Once initialized, the client broadcasts to all server systems in the test. Systems operating in server mode respond to the client system, register with the client, reset statistics counters, and indicate readiness to process test packets from the client.

After the registration phase, testing begins. The client continuously sends packets to each registered server in turn. This continues for either a given number of iterations or a given number of packets. After the test is completed, the client queries each server for its test statistics, displays them on the screen, and logs them to a file. Finally, it sends a packet to each server and directs it to quit the test.

Server Mode
The server system receives packets and provides the appropriate response. Depending on the command in a packet, a server responds by either sending the packet back to the client system or by discarding it. In either case, the server sets its counter to track the number of received packets, the number of packets it responded to, and any errors that occurred in the test.

Client and Server Mode
The client can also run as both a client system and a server system. This means it can send packets to server systems and return packets to itself or to other client systems.

After NDIS testing is complete, the client system directs the server system(s) to quit the test. All of the script files used for NDIS testing are executed on the client system. The client system logs all tests. The server system does not create log files.

3.0 Scenario Characteristics

When running the NDIS stress test on the DP83815 or the DP83816, there is a short hardware timing window, which can occur during packet reception, that causes the Mac portion of the device to incorrectly update its internal packet FIFO read register. This timing window can only occur when an incoming packet is rejected. In practice, the most common and likely packet rejection condition is a receive FIFO overrun. In this case, the incoming packet is discarded due to a lack of space in the receive FIFO.

If this timing window occurs, the Mac portion of the device has incorrect data in the packet currently being transferred to a host receive buffer, although the status of the packet indicates no error. Because the internal FIFO read register is set incorrectly after this condition, the receive logic indicates that there is another packet with a length greater than the size of the receive packet FIFO that is ready for transfer to the host.

The Mac portion of the device then transfers this invalid large packet to the host memory buffers. A common host receive buffer layout is to allocate a maximum packet size buffer to each receive descriptor. Because the packet is larger than this, it is scattered across multiple receive descriptors and the MORE bit in the COMMAND/STATUS (cmdsts) field is set in the first descriptor. The MORE bit may also be set in other descriptors, depending on the size of the buffer assigned to each receive descriptor.

Because the internal FIFO read/write registers are out of synchronization, future packets transferred to the host might also be affected.

4.0 System Impact

The impact of this problem depends on the requirements and assumptions made by higher layer protocol stacks. In most TCP/IP based environments, this problem should have little, if any, negative effects because TCP/IP discards any corrupt packet that fails checksum verification.

If the device's receive logic is consistently unable to keep up with the incoming packet flow, and overruns occur frequently, TCP closes its packet window down to a point where overruns do not occur. Since this occurs in an overrun case, regardless of whether packets were corrupt or not, the failure scenario should not have a negative impact on performance.

Applications that do not use TCP/IP may, or may not, be affected by this problem. If the driver takes corrective action, as outlined below, there may be no significant impact to the system. If the driver does not correct for this issue, and an upper layer transport does not validate the packet with some sort of CRC or checksum, data corruption may occur. This may result in BSOD (Blue Screen Of Death), system freeze, system shutdown or receive data corruption.

5.0 Resolutions/Workarounds

Since this scenario is not expected to occur in real network environments, no additional workaround is required for general applications.

If required, there are two approaches that the host driver can use to detect and work around this failure.

- Full-Size Receive Buffers
  This is appropriate when the host driver assigns a single full-size (max Ethernet packet) buffer to each receive descriptor. This approach is implemented in National's Windows driver set.
- Fragmented Receive Buffers
  This approach can be used when the host driver uses a receive descriptor architecture that allows packets to be fragmented over two or more receive descriptors.

5.1 FULL-SIZE RECEIVE BUFFERS

For full-size buffers (buffers that hold a full packet), packets are not fragmented across the buffers. Consequently, each packet has only one descriptor which means that the MORE status bit, of the descriptor, is always zero. Thus, for a receive packet, the driver can use the MORE bit to detect a failure event.

The driver's receive-processing logic checks each packet to determine the status of the MORE bit. If this bit is set, the driver assumes that the previous packet contains invalid data. Future packets are also affected until the receiver is reset. When the error is detected, the driver discards the previous receive packet, and all other packets indicated in the receive descriptor list, and reclaims the buffers. It then resets the Mac receiver (CR:RXR, register offset 00h), re-initializes the receive descriptor list, reprograms the RXDP (register offset 30h) and RXCFG (register offset 34h), and then enables the receiver (CR:RXE).

This solution might not detect and discard all invalid packets. There is a short window where the corrupt packet previous to a packet with the MORE bit set, straddles two passes of the driver's receive processing logic. In other words, the driver processes a receive interrupt and services the receive descriptor list. The last packet indicated on the receive descriptor list is corrupt. No subsequent packets have been indicated as complete by the hardware. The driver processes the packets believing they are valid, but the last one processed contains invalid data. In the next receive interrupt, the first packet indicated has the MORE bit set, indicating the failure event has occurred. Because the driver has already processed the previous invalid packet, and has no method to abort it, the upper layer protocols attempt to process that invalid packet.

5.2 FRAGMENTED RECEIVE BUFFERS

This method has not been validated. It is presented here as a potential workaround for drivers that use fragmented receive buffers. The method is basically the same as the previous one, but uses the reported packet length to detect the failure, instead of the MORE bit.

For a failure event, the sum of the SIZE fields in the receive packet descriptors exceeds the maximum possible packet size. The failure condition is thus detected, and is corrected using the method described above.

6.0 Summary

An anomaly has been observed on the DP83815 MacPHYTER and DP83816 MacPHYTER - II 10/100 Mb/s ethernet controllers, during Microsoft WHQL WinXP HCT 10.0 stress tests. While in theory this condition can occur in any high data rate stress test, it has only been observed in Microsoft's WHQL HCT environment.

The symptoms are corruption of the frame buffer(s) during packet receives, due to a rare intermittent hardware timing window.

As a workaround, this failure condition can be detected through software as described in Section 5.0, in two ways depending on whether full size or fragmented buffers are used. Once it has been detected the corrupted packets can be discarded and the reception of valid packets can be resumed.

National Semiconductor Corporation
Americas
Email:
new.feedback@nsc.com

National Semiconductor Europe
Fax: +49 (0) 180-530 85 86
Email: europe.support@nsc.com
Deutsch Tel: +49 (0) 69 9508 6208
English Tel: +44 (0) 870 24 0 2171
Français Tel: +33 (0) 1 41 91 87 90

National Semiconductor Asia Pacific Customer Response Group
Tel: 65-2544466
Fax: 65-2504466
Email: ap.support@nsc.com

National Semiconductor Japan Ltd.
Tel: 81-3-5639-7560
Fax: 81-3-5639-7507
Email: nsj.crc@jksmtp.nsc.com

www.national.com