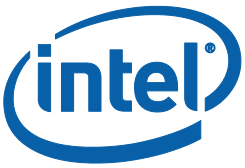


White Paper

Ai Bee Lim
Senior Platform Application Engineer
Embedded Communications Group
Performance Products Division
Intel Corporation

Jack R Johnson
Senior Platform Application Engineer
Embedded Communications Group
Performance Products Division
Intel Corporation

Processor Reorder Buffer (ROB) Timeout

Debug Guide

October 2010

Document Number: 324353

Abstract

This paper provides an overview of the Processor Reorder Buffer timeout and provides methodology to debug these types of system issues. Using the debug methods and debug tools suggested in this document should help reduce the time to debug these system issues. The process is to gather more information about the failure until the cause is identified and then put preventive steps in place to eliminate the failure.

Contents

1.0 Introduction

Processor Reorder Buffer (ROB) timeout is not new, yet debug engineers often spend a lot of time debugging system issues that result from seeing a Processor ROB timeout. The purpose of this paper is to give context and guidance to help hardware engineers and software engineers troubleshooting these issues.

Typically processors indicate a ROB timeout with an IERR# signal assertion. Interestingly IERR# assertion does not mean ROB timeout condition only, this means that the processor has experienced an internal error, and it may be a result of issues such as an error condition in the cache unit, error conditions in the internal bus etc.

For processors that support the Intel® Quick Path Interconnect interface, there is no longer IERR# or MCERR# signals from the processors. Instead they have been replaced by the CATERR# signal pin to indicate that a catastrophic error condition has been experienced by the processor.

If the Machine Check capability of the processor is enabled, this event can also be recorded in the Machine Check Status register. The processor ROB timeout is only one of the Machine Check events that can be recorded. This paper will only focus on the processor ROB timeout error condition, and provide guidance on debugging this Machine Check event.

2.0 Processor ROB Timeout

First, let's examine the meaning of a processor ROB timeout. Figure 1 is an example of a P6 Processor Micro-architecture with Advance Transfer Cache Enhancement.

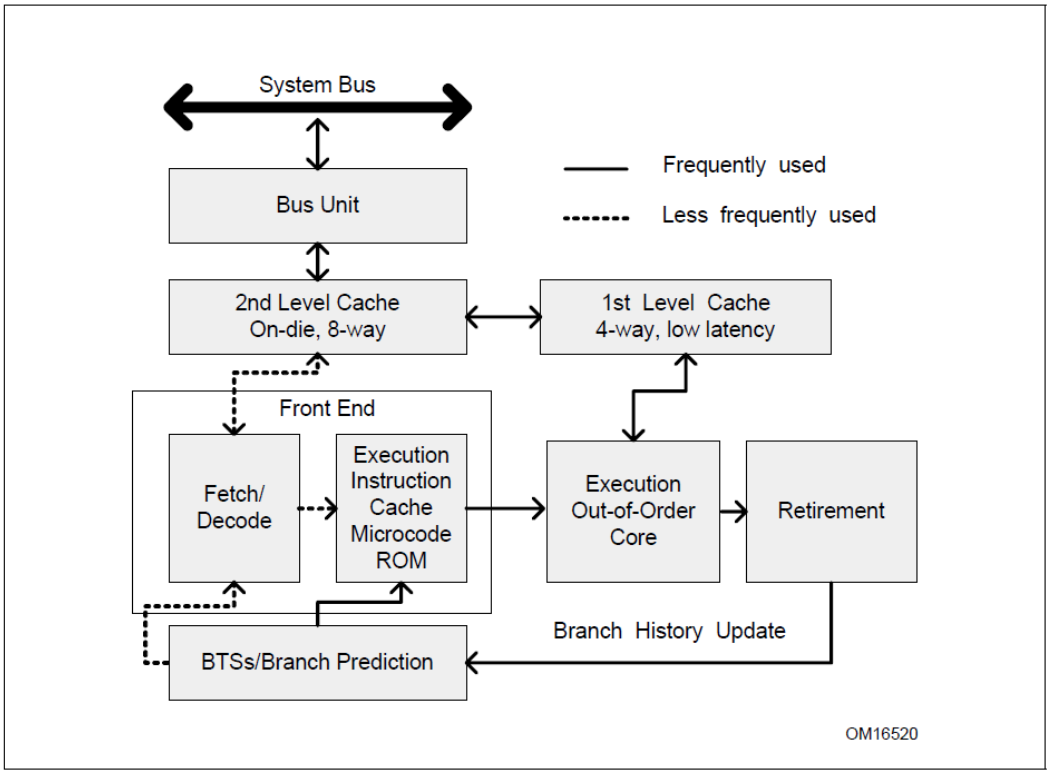


Figure 1 Intel® P6 Processor Micro-architecture with Advance Cache Transfer Enhancement

From the above figure, the processor execution consists of a few blocks:

- **Bus unit** which interacts with the system bus, known as the Front Side Bus for earlier Processors, and Intel® Quick Path Interconnect for more recent processors
- **Second Level Cache unit** which interacts with the Fetch/Decode Unit and the First Level Cache unit
- **First Level Cache unit** which interacts with the Out-of-Order Execution Unit.
- **Execution Out-of-Order Core unit** which is handling out of order execution
- **Retirement unit** which is responsible for retiring processor instructions in order
- **Branch Prediction** unit which offers branch predicting hints for the processor

In the processor Retirement unit, the processor instructions are retired in order even though the processor can support out-of-order execution. It is important to note that the instructions must retire in order to ensure the correctness of program execution. The ROB timer is reset on retirement of each micro-instruction. During

normal operation the processor retires instructions before the ROB timer times out. When the ROB timer expires, something usually is going on within the system hardware or software or both. This document discusses some examples of ROB timeout events.

3.0 Machine Check Status

As described earlier, the ROB timeout is a type of Machine Check event, thus it is recommended that Machine-Check Architecture events are enabled in the system in order to capture information related to a Machine Check event.

Note: Please refer to Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3A: System Programming Guide Part 1 for details on Machine-Check Architecture initialization.

Processor ROB timeout is reported in bits [15:0] of the MCi_STATUS – the MCACOD field – as an internal timer error condition with MCACOD == 0x400. Bit 38 of MC0_STATUS will also be set in the processor to report a BINIT# (Bus Init) timeout condition. This Processor signature is referenced in Table E-1 and E-3 of the *Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3B: System Programming Guide Part 2.*

Note: The MCi_STATUS[56:17] report processor model specific information.

4.0 Causes and Examples

4.1 Outstanding Read

When the next instruction to be retired is a read operation and that read does not complete before the ROB timer expires, a ROB timeout will be reported. This means that a memory read, an IO read, a Memory mapped IO read, or a Configuration read can cause this error.

When any single thread issues a read operation, it may be able to execute other instructions that do not require the completion of the read. At some point the thread needs the read result and it will stall waiting for a completion. When that completion does not occur, the system is headed for a ROB timeout event.

There are several conditions that could prevent the read from completing before the ROB timer expires. Some of the more interesting ones include a device which never responds to a device status read, a device which partially responds or one that returns an error result to other parts of the system but never actually completes the read.

4.2 Outstanding Write

Typically, writes are posted and should not, by themselves, cause an ROB timeout. It is possible that some issues downstream have consumed all the transaction resources causing the functional unit to push back on the system bus, resulting in no progress as seen at the processing unit. This eventually leads to a ROB timer expiring since the pending write instruction cannot be retired within the ROB timeout interval.

5.0 Debug Tips and Tools

This section focuses on the debug tips and tools that may be used to determine the root cause of a ROB timeout event.

5.1 The Basics

It is important to ensure that the system has all the basic tasks completed in order to avoid spending unnecessary time working issues that are already well known. Some of the first steps include:

- Ensure that System BIOS has the latest Processor Microcode Update (MCU) and the latest Chipset Configuration information
- Review known Machine Check or System Hang issues which are identified in the Processor Specification Update and Chipset Specification Update and any other Component Specification Updates (this applies to all components used on the system board)
- Review the silicon steppings of all the components in the system to verify that they are the latest available.
- Correlate the failure as much as possible by testing multiple systems and collecting as much data as possible. Work hard to understand all the variables that are involved with generating the failing case.

5.2 Machine Check Handler

Ensure that the Machine-Check Architecture of the processor is enabled in order to confirm that the processor is seeing a ROB timeout Machine Check event. This is done by setting the MCE bit – bit 6 - in Control Register four (CR4), and following the initialization of the Machine Check routine as outlined in the *Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3A: System Programming Guide Part 1* in the Machine Check Architecture chapter.

In order to confirm that a system has enabled the Machine Check Architecture, check to make sure that the MCE bit is set in the Processor Control Register number four (CR4.MCE==1). Check the MCi_CTL register to make sure that the register

does not read 0x0. When MCi_CTL reads 0x0, the Machine Check Initialization may not have taken place during BIOS initialization.

Part of the Machine Check initialization is to install a software handling routine at vector 0x18h. At a minimum, the Machine Check handler at vector 0x18 should report all the Machine Check Status registers and the relevant Machine Check Address or Miscellaneous registers if they are valid.

The routine attached to vector 0x18 can also report the Global Error status register for the chipset to have a snapshot of the system condition leading to the Machine Check Event. For most of the Machine Check cases, this register will provide more information about the system condition. In rare events that will be discussed later in this paper, the execution may stop and the handling routing may not execute correctly. Under this condition, the problem is more difficult to debug.

5.2 Some signals of interest on the Processor

Processors that support Front Side Bus (FSB) Architecture

Processor IERR# signal is asserted when the Processor is experiencing a serious internal error condition when the Machine Check Architecture (MCA) is disabled. IERR# will be asserted until the processor is reset.

When this signal is asserted it indicates that the processor is experiencing a serious internal error condition causing the system to reset. Assertion of this signal suggests that the Machine Check Architecture has not been enabled or that a serious error occurred while attempting to handle the Machine-Check event

The Processor MCERR# signal is asserted when the processor is experiencing a Machine Check event. If MCA is enabled and the system is correctly configured, then it will drive this signal on the system bus. During an uncorrectable Machine check condition, the MCERR# will be asserted for three clocks and the Machine Check Exception Handling routing installed at vector 0x18 will be called. If the system sees this condition, it verifies that the system has experienced a Machine Check Event and that the Machine Check Initialization has completed properly in order to have MCA enabled. If the MCERR# is not asserted during a Machine Check event, the platform configuration settings should be reviewed in order to confirm the correct settings needed to drive the MCERR# signal on the system bus.

Processor Block Next Request (BNR#) signal is asserted by any FSB agent to insert a bus stall when the agent cannot accept any more new bus transactions due to resource availability and to prevent overflow. When this signal is asserted, the bus owner cannot issue a new transaction on the bus. This signal indicates that there is backpressure on the bus.

Processors that support Intel® QuickPath Interconnect Architecture

For the current generation of processors that support the Intel QuickPath Interconnect Architecture, a few of the processor signals have been consolidated and thus there is no longer separate IERR# and MCERR# signals. The Processor

Catastrophic Error (CATERR#) signal is asserted when processor is experiencing a serious internal error condition.

The CATERR# signal is asserted until the processor is reset (PLTRST# signal asserted) when the Machine Check Exception is not enabled. The CATERR# signal will pulse when the Machine Check Exception is initialized and enabled correctly. Hence, looking at a waveform of this signal will confirm if the system has enabled the Machine Check Exception properly.

5.3 In Target Probe Tool (XDP/Arium)

By design, Machine check status registers retain their value thru system reset but the initialization of the Machine-Check procedures will clear these registers. In order to check the status registers for the error condition that led to the Machine-Check event, one should take a snapshot of these registers before they are cleared on the next boot. If the Machine Check Exception is enabled, the exception routine at a minimum will record the MCi_STATUS registers information. For some rare occasions, the Machine Check Exception is not executed properly. It may be possible to read the MCi_STATUS registers which are cleared at the next boot by the Machine-Check Initialization procedures. This can be done with an In-circuit debug tool that allows manual control of the processor's execution. For example it may be possible to stop the processor at a certain boot location with a breakpoint and then inspect the processor Machine Check Registers in order to debug the system.

The In-circuit debug tool is very powerful and allows the operator to stop the processor at various software locations and can help find clues to the problem area of the code before the ROB timeout occurs.

5.4 PCI Express Completion Timer Timeout

If the system has experienced a ROB timeout, it is always good practice to enable all the PCI Express End Points Completion Timeouts. Some root complexes allow enabling of PCI Express Completion timeout on Configuration accesses. If this is the case, enable this feature as well. This helps avoid an outstanding configuration request targeting a PCI Express end point which is not being completed before the ROB timer expires. Since the completion timeout is in milliseconds and the ROB timer is in seconds, the completion timeout should alert the system to take appropriate action before a ROB timeout occurs, even if the target cannot provide a response.

Note: For a PCI Express transaction to time out, the transaction must have actually been initiated on the PCI Express interface.

5.5 PCI express Logic Analyzer

If a certain PCI express End Point in the system is suspected to as the cause of the problem, then one should collect a PCI Express Logic Analyzer trace. This should be an unfiltered capture of all transactions.

Review the Logic Analyzer traces to understand if there are any outstanding requests going downstream that has not been completed which may cause ROB timeout.

5.6 Bus Logic Analyzer

Processors that support the Front Side Bus (FSB) Architecture

An FSB bus trace will provide visibility of the bus transactions between the CPU and the Chipset. The FSB Logic Analyzer decodes these transactions so that the system user will be able to make sense of the all the CPU to Chipset activity.

In the case of a debugging an ROB timeout, the system user will be able to trace which transaction the CPU is waiting on before the CPU resets. In an ideal case, this bus trace is important to understand what caused the ROB timeout problem. Unfortunately getting access to this debug tool or getting access to all the FSB signals may be challenging so this is one of the last options in most debug efforts.

Processors that support the Intel QuickPath Interconnect Architecture

For processors that support the Intel® QuickPath Interconnect Architecture, similar Logic Analyzing capability is available through the Mirror Port or Mid-Bus Probe. In this case, the system BIOS needs to be configured to allow this activity. Also, there are challenges in setting up the system board with the Intel® QuickPath Interconnect Logic Analyzer. Thus, it is also recommended that all the other methods are used to debug the problem before this step unless one of the analyzers is readily available.

6.0 ROB Timeout Examples

6.1 Chipset Known Behavior Issue

This is an example of ROB timeout that is caused by known Chipset behavior. In this example, the chipset involved is the Intel® 5100 MCH Chipset, so all the known chipset behavior issues are documented in the component specification update that can be obtained from the Intel products website.

In Section 5.0, one of the first debug steps is to review all the known issues first in order to not expend time and effort on something that has already been discovered. This erratum item is described and a workaround recommendation is also provided to avoid this condition from occurring in a similar implementation using this component.

35.	**PCI Express* transaction I/O ordering queue overflow**
Problem:	Under some corner case scenarios when the system is stressed with heavy I/O traffic, a hang condition could occur as the transactions to/from the PCI Express* port are blocked due to an I/O order queue overflow.
Implication:	The overflow condition will eventually cause the system to deadlock because many outstanding transactions are unable to make forward progress. The system level implication is a system hang with IERR or MCERR or PCI Express* fatal error asserting NMI, if enabled.
Workaround:	Please refer to the latest *RS - Intel® 5100 Memory Controller Hub Chipset BIOS Specification* release.
Status:	No Fix

The PCI Express I/O ordering queue will overflow when all the advertised credits for resources are consumed. If a transaction is outstanding long enough for the posted credit to be returned and a new posted transaction arrives from the PCIe device, the transaction I/O ordering queue will overflow. This overflow condition causes the system to deadlock because transactions cannot make progress and this eventually results in an IERR, MCERR, or PCI Express Fatal error asserting NMI if enabled. This issue can cause the system to IERR or MCERR due to an ROB timeout. This item is included in a workaround for the BIOS, so the first rule of thumb is to understand the workaround and ensure that this issue is ruled out.

6.2 ROB timeout due to Outstanding Read

In order to debug a ROB timeout issue, we need to understand the system memory map and how an access from the processor maps to a physical device. From a processor perspective, a read from an address means reading a system location which could further translate to either a read to system memory if the address falls into physical system memory or it could translate to an IO read if it is targeting IO address space. It could also translate into a configuration read to an internal PCI device or an external PCI device, depending on enumeration of all the PCI devices in the system. For this example, the processor read translates to a configuration read to an external PCI Express device card within the system.

Typically it is not easy to identify all the potential contributors to the problem. Hence, taking some register dumps of the complete system during the problem or before and after the problem occurs are good practices to help zoom in and debug the problem. Certain error conditions being asserted in a PCI Bus hierarchy may lead us to a possible root cause.

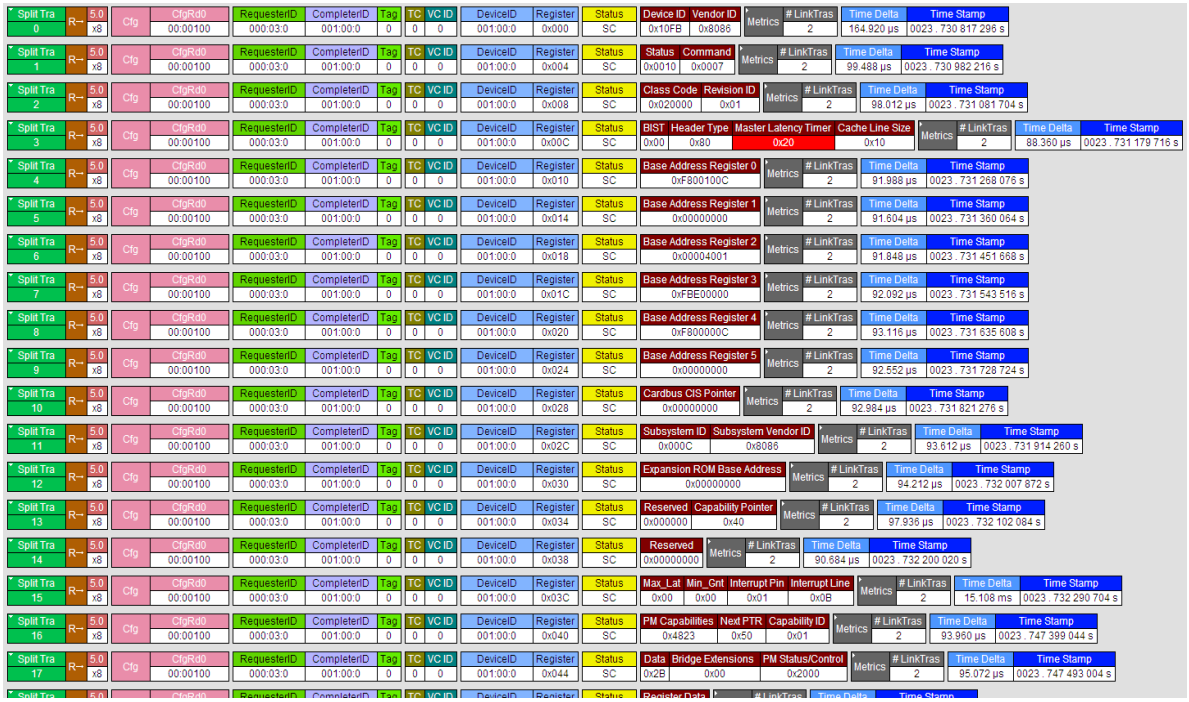
Another technique is using an in-circuit debug tool (e.g., Intel XDP or American Arium XDP tool) to help focus on processes right before the Machine Check event to understand the problem a little better. With this step, we may be able to identify potential contributors and collect more data to help work toward root cause.

If we believe that one of the end points is the primary suspect, we may want to use a PCI express Analyzer to collect all the transactions leading up to the problem in order to diagnose the issue properly and efficiently.

Normal Operating Case

Figure 2 shows a normal PCI Express trace for this specific End point responding to configuration reads from the host.

Figure 2 PCI Express Trace with End Points Responding



At the console, all the configuration reads are completed successfully.

```
[root@localhost ~]# lspci -xxx -d 8086:10fb
01:00.0 Ethernet controller: Intel Corporation Unknown device 10fb (rev 01)
00: 86 80 fb 10 07 00 10 00 01 00 00 02 10 20 80 00
10: 0c 10 00 f8 00 00 00 00 01 40 00 00 00 00 e0 fb
20: 0c 00 00 f8 00 00 00 00 00 00 00 00 86 80 0c 00
30: 00 00 00 00 40 00 00 00 00 00 00 00 0b 01 00 00
40: 01 50 23 48 00 20 00 2b 00 00 00 00 00 00 00 00
50: 05 70 80 01 00 00 00 00 00 00 00 00 00 00 00 00
60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
70: 11 a0 3f 00 04 00 00 00 04 20 00 00 00 00 00 00
80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
a0: 10 00 02 00 c2 8c 00 10 20 18 00 00 82 f4 02 00
b0: 40 00 82 10 00 00 00 00 00 00 00 00 00 00 00 00
c0: 00 00 00 00 1f 00 00 00 00 00 00 00 00 00 00 00
d0: 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

End Point Stops Responding

Figure 3 shows an example PCI Express trace for an event where all the Configuration accesses are indicating an incomplete status. This may point to an issue with the End Point.

Figure 3 PCI Express Trace with Configuration Status Incomplete

Note: The above issue does not result into a ROB timeout since the Completion Timeout mechanism is enabled in the system.

For the PCI express trace above, since the Completion Timeout Mechanism is enabled at the root port, the transaction is master aborted by the root port, and we are seeing all 0xFs returned to complete the transaction to the requestor (this case the requestor is the processor).

The following is what is seen at the at the console:-

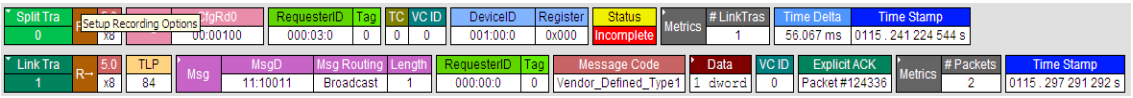
```
01:00.0 Ethernet controller: Intel Corporation Unknown device 10fb (rev ff)
00: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
10: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
20: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
30: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
40: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
50: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
60: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
70: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
80: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
90: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
a0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
b0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
c0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
```

```
d0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
e0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
f0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
```

Completion Timeout Mechanism Disabled at Root Port

Figure 4 shows a PCI express Logic Analyzer trace that has an incomplete status for the transaction. In this case, the completion timeout mechanism is disabled in the root port.

Figure 4 PCI Express Trace Incomplete Transaction



Since the Completion Timeout Mechanism is not enabled, the transaction is not master aborted by the root port, the transaction is seen going out to the PCI express end point but it never completes. In this case, the processor is waiting and it cannot retire this transaction and this eventually leads to a ROB timeout seen by the processor. The following is what may be seen at the the console when the processor hangs:

```
01:00.0 Ethernet controller: Intel Corporation Unknown device 10fb (rev ff)
```

If Machine-Check events are enabled in the Operating System and an INT18 handler is installed, this will trigger a Machine Check event and initiate the INT18 handling routine that is installed in the Operating system.

Note: In the above example the PCI express trace captures a configuration read that the End Point does not respond to that eventually results in a ROB timeout Machine Check event. The same will occur with Memory Read targeting IO memory that the end point does not respond to.

6.3 ROB timeout due to Outstanding Writes

Understanding the system address of the target helps in debugging of the problem. As mentioned earlier, writes are typically posted and it's very rare that a write would lead to a ROB timeout. Let's consider a unique example of a ROB timeout that is caused by outstanding writes.

Figure 5 shows a PCI topology bus in an Intel Architecture System. The system debug engineer should be aware of accesses from the host targeting PCI Express devices in the system in order to effectively debug system issues.

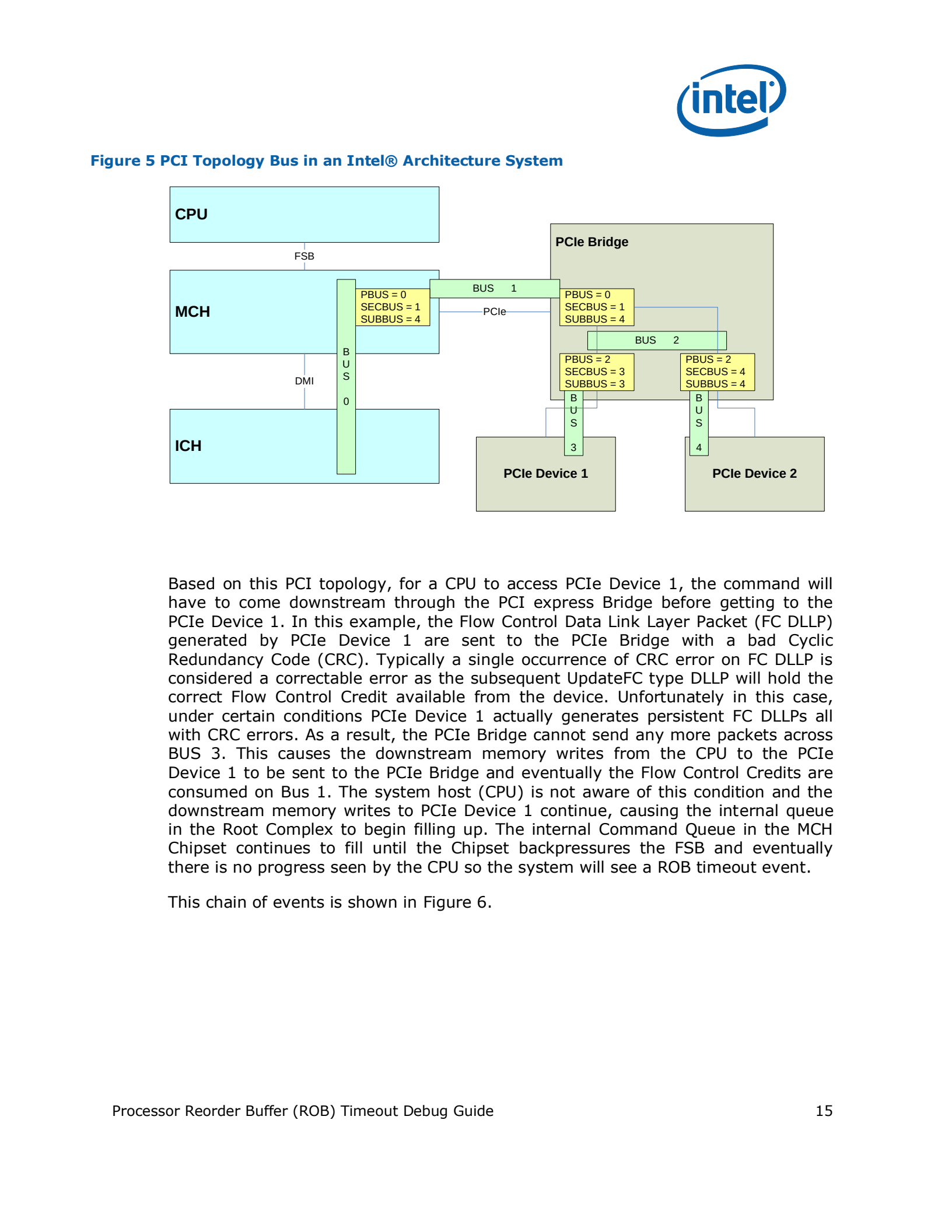


Figure 5 PCI Topology Bus in an Intel® Architecture System

Based on this PCI topology, for a CPU to access PCIe Device 1, the command will have to come downstream through the PCI express Bridge before getting to the PCIe Device 1. In this example, the Flow Control Data Link Layer Packet (FC DLLP) generated by PCIe Device 1 are sent to the PCIe Bridge with a bad Cyclic Redundancy Code (CRC). Typically a single occurrence of CRC error on FC DLLP is considered a correctable error as the subsequent UpdateFC type DLLP will hold the correct Flow Control Credit available from the device. Unfortunately in this case, under certain conditions PCIe Device 1 actually generates persistent FC DLLPs all with CRC errors. As a result, the PCIe Bridge cannot send any more packets across BUS 3. This causes the downstream memory writes from the CPU to the PCIe Device 1 to be sent to the PCIe Bridge and eventually the Flow Control Credits are consumed on Bus 1. The system host (CPU) is not aware of this condition and the downstream memory writes to PCIe Device 1 continue, causing the internal queue in the Root Complex to begin filling up. The internal Command Queue in the MCH Chipset continues to fill until the Chipset backpressures the FSB and eventually there is no progress seen by the CPU so the system will see a ROB timeout event.

This chain of events is shown in Figure 6.

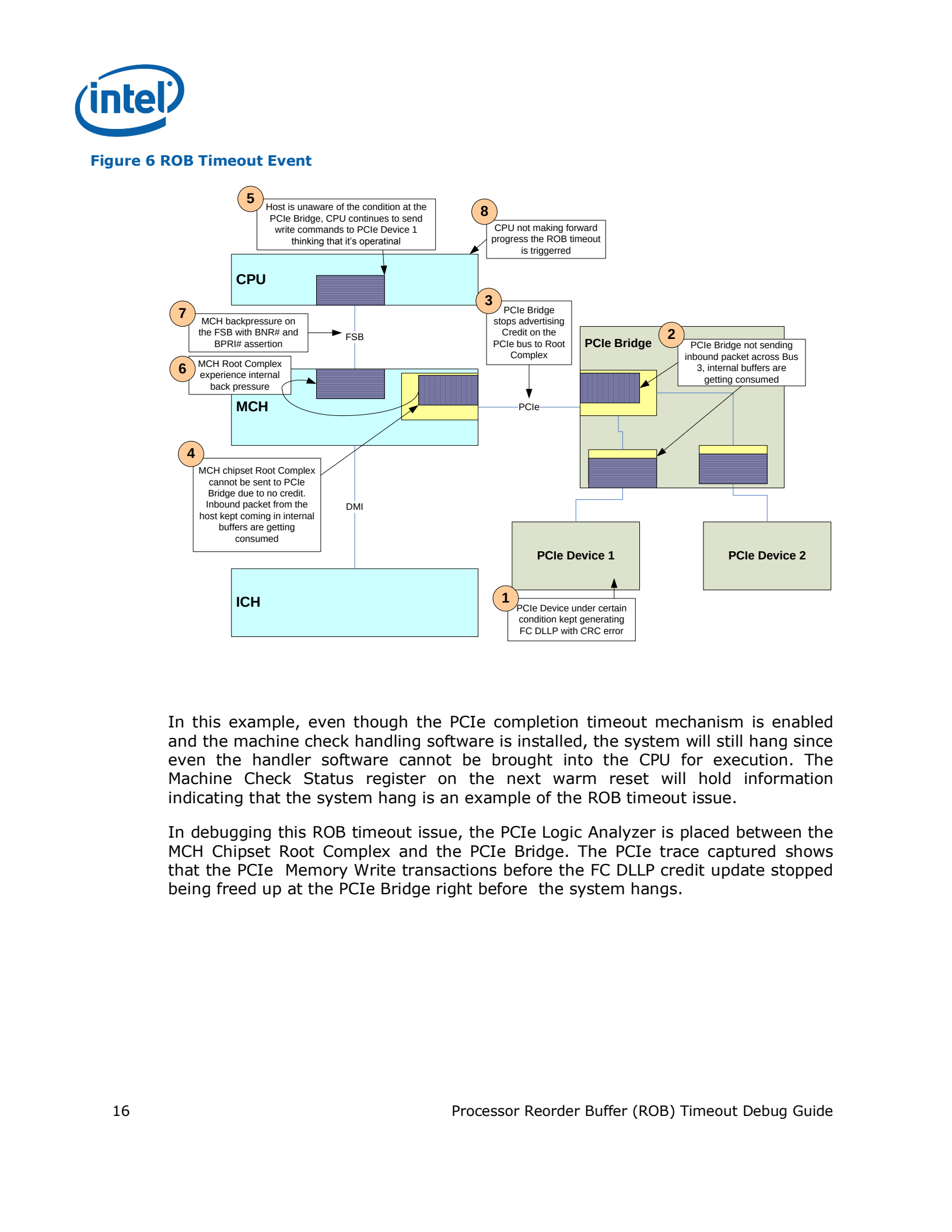


Figure 6 ROB Timeout Event

In this example, even though the PCIe completion timeout mechanism is enabled and the machine check handling software is installed, the system will still hang since even the handler software cannot be brought into the CPU for execution. The Machine Check Status register on the next warm reset will hold information indicating that the system hang is an example of the ROB timeout issue.

In debugging this ROB timeout issue, the PCIe Logic Analyzer is placed between the MCH Chipset Root Complex and the PCIe Bridge. The PCIe trace captured shows that the PCIe Memory Write transactions before the FC DLLP credit update stopped being freed up at the PCIe Bridge right before the system hangs.

7.0 Summary

Debugging a system problem is never easy, and the more experience you have the better. The debug of a problem needs to be very methodical. Data collection needs to be precise, and every variable needs to be documented with the relevant observations so that the debug path can be identified easily. Processor ROB timeout is a specific problem, and this paper attempts to explain the causes and methods to help resolve the problem as quickly as possible.

Special acknowledgements go to **Sivakumar Radhakrishnan (**Senior Chipset Architect in Intel Architecture Group, Intel Corporation) and **Jeffrey D Gilbert** (Senior Principal Engineer in the Intel Architecture Group, Intel Corporation) for providing support and guidance.