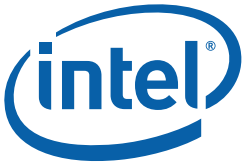


White Paper

Mark J. Sullivan
Network Software Engineer
Intel Corporation

Intel® Xeon® Processor C5500/C3500 Series Non-Transparent Bridge

January 2010

Document Number: 323328-001

Abstract

This paper discusses the non-transparent bridge device included in the Intel® Xeon® Processor C5500/C3500 Series. An overview of the hardware is provided along with software usage models.

Contents

Figures

1 Introduction

The Intel® Xeon® Processor C5500/C3500 Series is the next generation of high-performance multi-core architecture targeted for embedded and storage use cases. Based on 45nm technology, the architecture integrates multiple processor cores, memory controller, PCI Express* (PCIe) interface, Crystal Beach DMA engines, and IO virtualization blocks in a single chip. Augmenting several embedded features is the integration of Non-Transparent Bridge (NTB) which enables high speed connectivity between one Intel Xeon Processor-based platform to another (or other IA or non-IA platform via the PCIe interface).

This paper provides a detailed look at the non-transparent bridge device integrated into the Intel® Xeon® Processor C5500/C3500 Series.

1.1 Terminology

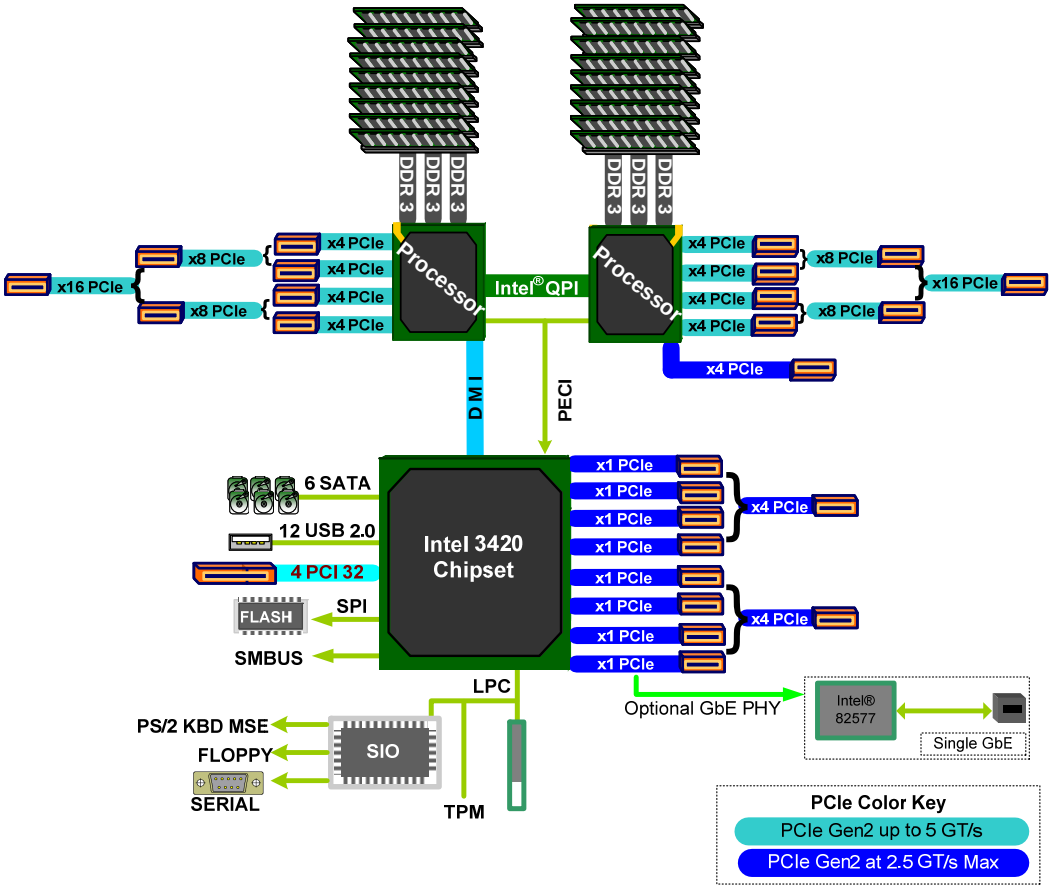
Term	Description
B2B	Back to Back
BAR	Base Address Register
DMA	Direct Memory Access
DP	Dual-Processor
IA	Intel® Architecture
IIO	Integrated Input Output
MMIO	Memory Mapped I/O
NTB	Non-Transparent Bridge
PCIe	PCI Express
PPD	PCIe Port Definition
RP	Root Port
TB	Transparent Bridge
UP	Uni-Processor
VT-d	Intel® Virtualization Technology for Directed I/O

2 Intel® Xeon® Processor C5500/C3500 Series NTB Hardware

The Intel Xeon Processor C5500/C3500 Series contains an integrated IIO, which supports 16 PCIe Gen2 lanes configurable as a single x16 port, two x8 ports, or four x4 ports. A x8 along with two x4 ports is also possible. The first port (port 0), configured as either x4 or x8, can be configured as a non-transparent bridge.

Figure 1 presents a block diagram of an Intel Xeon Processor dual-socket-based system.

Figure 1. Intel® Xeon® Processor C5500/C3500 Series Dual-Socket-Based System Block Diagram



2.1 NTB Features at a Glance

The following sections describe the NTB features supported by the Intel Xeon Processor C5500/C3500 Series.

2.1.1 Supported NTB Features

- PCIe Port 0 can be configured to be either a transparent bridge (TB) or an NTB.
  - NTB link width can support one x4 or one x8. The remaining x4 in the former case can still be used as a transparent bridge.
- The NTB port supports Gen1 and Gen2 PCIe speeds.
- The NTB supports two usage models
  - NTB attached to a Root Port (RP)
  - NTB attached to another NTB
- Supports three 64-bit BARs
  - BAR 0/1 for NTB internal configuration
  - BAR 2/3 and BAR 4/5 are prefetchable memory windows that can access both 32-bit and 64-bit address space through 64-bit BARs
  - BAR 2/3 and BAR 4/5 support direct address translation
  - BAR 2/3 and BAR 4/5 support limit registers
- Limit registers can be used to limit the size of a memory window to less than the size specified in the PCI BAR. PCI BAR sizes are required to be a power of 2 (for example: 4GB, 8GB, 16GB). The limit registers allow the user to select any value to a 4KB resolution within any window defined by the PCI BAR. For example, if the PCI BAR defines 8GB region, the limit register could be used to limit that region to 6GB.
  - One use case for limit registers also provides a mechanism to allow separation of code space from data space.
- Supports posted writes and non-posted memory read transactions across the NTB.
- Supports sixteen 32-bit scratchpad registers, (total 64 bytes) that are accessible through the BAR0 configuration space.
- Supports two 16-bit doorbell registers (PDOORBELL and SDOORBELL) that are accessible through the BAR0 configuration space.
- Supports INTx, MSI and MSI-X mechanism for interrupts on both sides of the NTB in the upstream direction only.
  - For example, a write to the PDOORBELL from the link partner attached to the secondary side of the NTB will result in an INTx, MSI, or MSI-X in the upstream direction to the local Intel Xeon processor.
  - A write from the local host on the Intel Xeon platform to the SDOORBELL will result in an INTx, MSI, or MSI-X in the upstream direction to the link partner connected to the secondary side of the NTB.
- Capability for passing doorbell/scratchpad across back-to-back NTB configuration.

2.1.2 Features Not Supported on the NTB

The NTB does not support the following:

- x16 link configuration
- IO space BARs
- Vendor-defined PCIe message transactions; if received, these messages are silently dropped.

2.2 Intel® Xeon® Processor PCIe Port Topologies

2.2.1 PCIe Ports Configured as Root Ports

Figure 2 illustrates a simple system with the Intel Xeon processor IIO exposing three root ports, two connected to PCIe devices and the third connected to a PCIe switch, which, in turn, is connected to three additional PCIe devices.

Figure 2. Intel® Xeon® Processor-Based System with no NTB

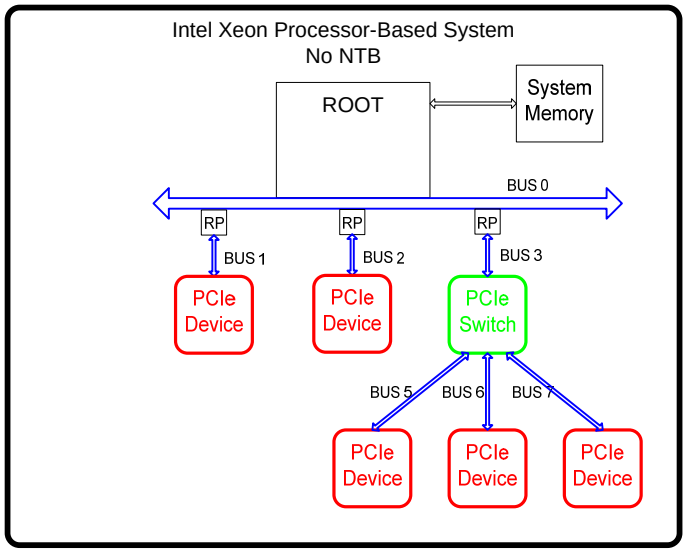
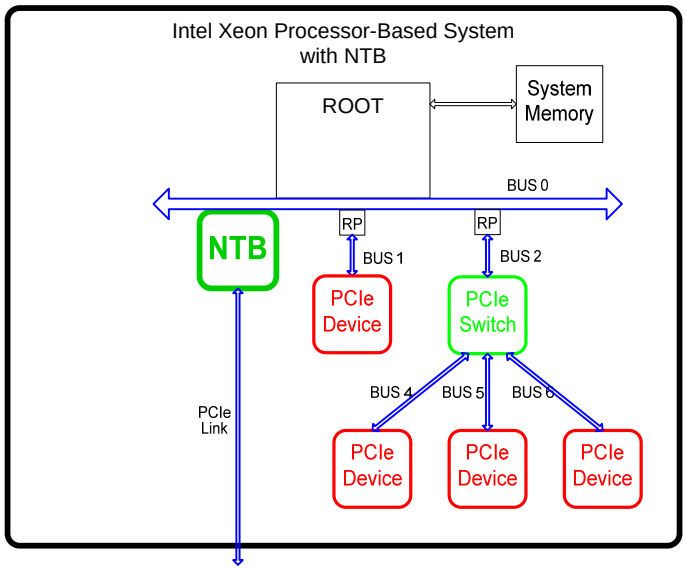


Figure 3 shows the same system, but with the NTB enabled, replacing the first root port. This enablement is done by the BIOS running on the Intel Xeon Processor-based system by writing to a register in the port's configuration space. This register must be written before the BIOS runs enumeration software.

The NTB is an integrated device on PCI Bus 0. It consists of two devices: the Primary device, and the Secondary device. Each of the two devices has its own PCI Configuration space. The Primary device is the device seen in the PCI tree of the Intel Xeon Processor-based system. In the processor, it is hardwired at Bus 0, Device 3, Function 0. The Secondary side of the NTB exposes a PCIe link that is connected to a PCIe tree in a second system. An example of this is shown in Figure 4. This link is connected to a PCIe port, either a root port as illustrated, or a switch port of a second system (referred to in the figure as the "PCI Express System").

2.2.2 Intel® Xeon® Processor-Based System with Port 0 Configured as NTB

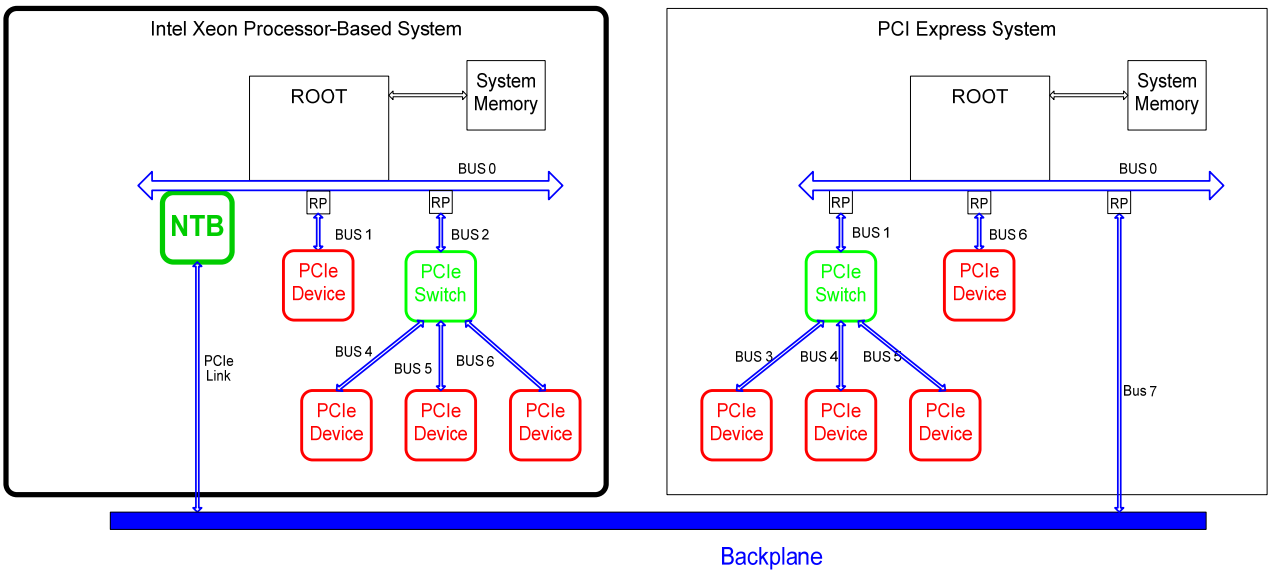
Figure 3. Intel® Xeon® Processor-Based System with NTB



Note: Additional information is provided in Figure 5.

2.2.3 NTB Connected to Root Port to Other Platform

Figure 4. NTB Connection



In the configuration shown in Figure 4, the system labeled Intel Xeon Processor-Based System will find the primary side of the NTB on Bus 0, Device 3, Function 0. The NTB configuration space defines three Base Address Registers (BAR) in the primary configuration space. All of these BARs are 64-bit addressable and can be placed anywhere in the Intel Xeon processor 40-bit address space.

The PCIe link exposed by the secondary side of the NTB is found in the PCIe topology of the system labeled "PCI Express System". In the figure, it would be at Bus 7, Device 0, Function 0.

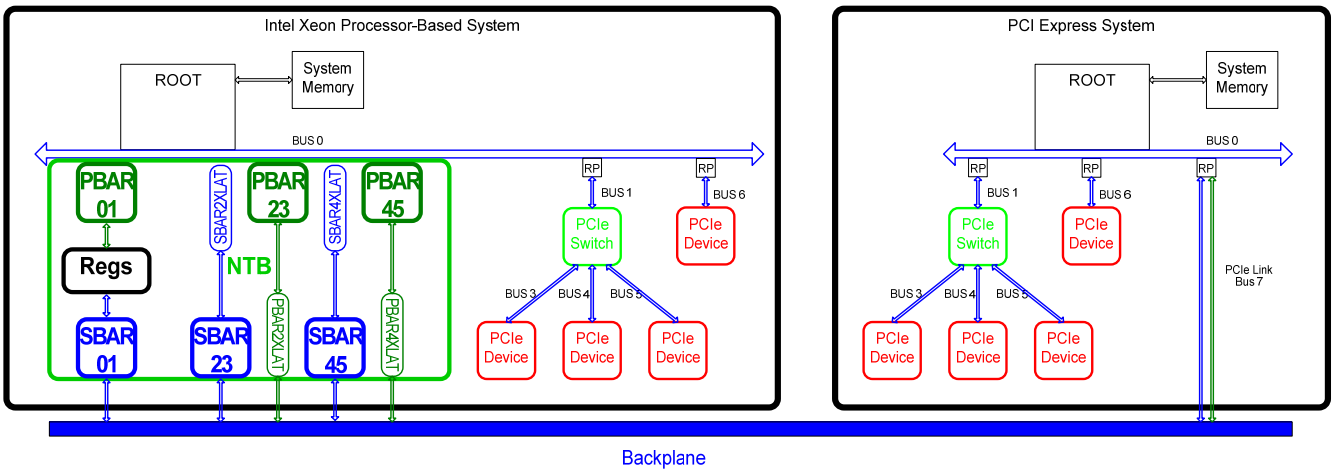
2.3 NTB Connected Systems

Figure 5 presents an enlargement of the NTB resources within the two connected systems. The color coding shows, at a high level, the data paths taken to access resources on one system from the other system.

The processor on the Intel Xeon Processor-Based System would read or write PBAR23 or PBAR45. The NTB would capture that read or write packet and (using PBAR2XLAT or PBAR4XLAT) convert it into a new address in the memory map of the PCI Express System.

Similarly, the PCI Express system would read or write SBAR23 or SBAR45, which would get translated by SBAR2XLAT or SBAR4XLAT into the memory map of the Intel Xeon Processor-Based System.

Figure 5. Data Paths



2.4 NTB Primary

The view into the NTB from the processor on the primary side of the NTB is comprised of its PCI Configuration space and three BARs: PBAR01, PBAR23 and PBAR45. PBAR01 contains the MMIO base address of the NTB internal registers while PBAR23 and PBAR45 are used to gain access into the memory map of the system connected to the secondary side of the NTB.

2.4.1 Primary Side PCI Configuration Space

The primary side of the NTB exposes a PCI Type 0 configuration space, including all the PCI-required registers such as Vendor ID, Device ID, and Base Address Registers (BAR).

The configuration space also contains several PCI capability structures, including MSI, MSI-x and the required PCI Express capability.

In addition to the PCI defined registers, the configuration space also contains five registers to define the functionality of the port. The values for these registers are user-configurable and are normally stored in non-volatile storage (such as flash memory) by the BIOS.

One of these registers, PPD, defines the behavior of the port. It can be configured in one of three ways:

1. as a PCI Express Root Port
2. an NTB connected to a PCIe port of a second system
3. an NTB connected back-to-back with another NTB on a second platform

This register is programmed by BIOS software, and must be programmed before any PCI enumeration software is executed. Programming this register causes the NTB hardware to expose the correct configuration space to the enumeration software.

The other four registers (PBAR23SZ, PBAR45SZ, SBAR23SZ, and SBAR45SZ) define the sizes of the BARs used to gain access into the attached system. The former two BARs are on the primary side of the NTB and the latter two BARs are on the secondary side. These registers also must be programmed prior to PCI enumeration software running on either side of the NTB. Valid values for each of these registers are from 12 to 39, yielding BAR sizes that range from 4KB (2^{12}) to 512GB (2^{39}). If the system attached to the secondary side of the NTB powers up before the Intel Xeon processor-based system, it will not see the NTB device. When the system is powered up, a hot-plug event will be required to include the NTB in the memory map of the second system.

2.4.2 PBAR01 – Primary Side PCIe Base Address Register 0-1

The first BAR, PBAR01, is used to control the behavior of the NTB as well as for inter-processor communication. This BAR is memory-mapped I/O (MMIO) and has a fixed size of 64KB.

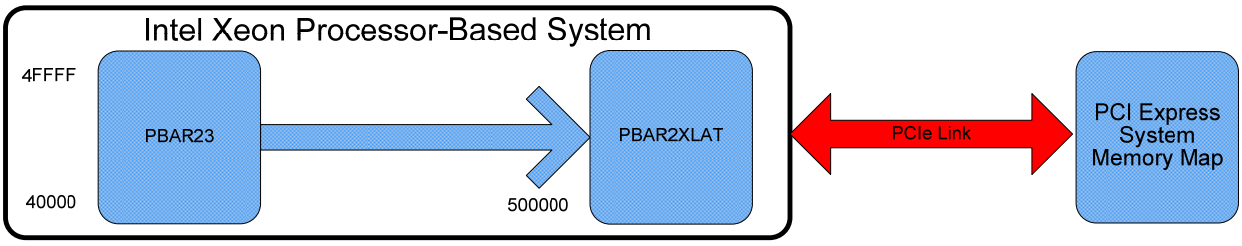
The following registers are included in this MMIO space:

- PBAR2LMT and PBAR4LMT – Primary BAR Limit Registers
  The PCI specification defines BAR sizes to be a power-of-2. These registers are used to place an upper cap on the size of BAR23 and BAR45 on the primary side of the NTB, i.e., if a 3GB BAR size is desired, the PBARnSZ register would be set to 32 (2^{32} = 4GB). Using the appropriate value in PBARnLMT register would limit the effective BAR address space to 3GB and cause the BAR to reject any accesses to the addresses between 3GB and 4GB. Note that the PBARnLMT registers cannot be modified from the secondary side of the NTB.

- SBAR2LMT and SBAR4LMT – Secondary BAR Size Limit Registers
  These registers perform the same function as the PBARnLMT registers, but they affect the BARs on the secondary side of the NTB. These registers are writable from both sides of the NTB whereas the PBARnLMT registers cannot be modified from the secondary side of the NTB.

- PBAR2XLAT and PBAR4XLAT – Primary Address Translation registers
  These registers are associated with the memory map of the system on the secondary side of the NTB. They are used to direct the accesses to the BARs on the primary side of the NTB into the system connected to the secondary side of the NTB. An example of this can be seen in Figure 6. PBAR23 contains a value of 0x40000 and is 64KB in size. PBAR2XLAT is initialized with a value of 500000. Any access from the Intel Xeon Processor to an address between 0x40000 and 0x4FFFF would be captured by the NTB and converted to an access into the memory of the PCI Express System at 0x500000 – 0x50FFFF.

Figure 6. Translate Example



- SBAR2XLAT and SBAR4XLAT – Secondary Address Translation Registers
  These registers are similar to PBARnXLAT registers, but are used to translate addresses from the PCI Express System to the Intel Xeon Processor-Based System. These registers are writable only from Primary side.

- SBAR0BASE, SBAR2BASE and SBAR4BASE – These registers are a copy of the three BAR registers located in the configuration space of the secondary side of the NTB. They allow the software running on the primary side of the NTB to modify the physical BAR registers on the secondary side of the NTB. These are included to enable certain usage models where there is no agent on the secondary side of the NTB capable of running PCI enumeration such as NTB/NTB configuration.

- NTBCNTL – NTB Control Register
  The NTB has a feature where CPU snooping of data packets destined for system memory can be enabled or disabled. This feature is controllable in the NTBCNTL register on a per-BAR basis. NTBCNTL also allows the PCIe link to be disabled/enabled. This is important because the registers on the secondary side of the NTB must be configured before enumeration can be run on the NTB device. In the case where the system connected to the secondary side of the NTB is powered on first, disabling the PCIe link will prevent the enumeration software from discovering the NTB. In this case, when the link is enabled, a hot plug controller should notify the CPU that a device was added. There is also a bit in this register to prevent writes to the configuration space from the CPU on the secondary side of the NTB. This allows the software on the primary side of the NTB to completely control the NTB.

- PDOORBELL and SDOORBELL – Primary and Secondary doorbell registers
  PDOORBELL is written by the CPU on the secondary side of the NTB to generate an interrupt to the CPU on the primary side of the NTB. Only bits 13:0 are available for interrupt generation. Bit 14 is used to notify completion of a cache flush operation described in Section 2.6,Cache Flush Mechanism, and bit 15 generates an interrupt whenever the link state on the PCIe link changes. SDOORBELL is written by the CPU on the primary side of the NTB to generate an interrupt to the CPU on the secondary side of the NTB. For both registers, writing a 1 causes the interrupt to be generated. The CPU getting the interrupt writes a 1 to clear the bit.

- PDBMASK and SDBMASK – These registers allow interrupt generation to be prevented. Any bit written to a 1 will disable interrupt generation from the corresponding bit in the doorbell registers.

- SPADnn – The NTB has a shared set of sixteen 32-bit scratchpad registers available to software on both CPUs for inter-processor communication.

- SPADSEMA4 – This register can be used by software running on the CPUs on both sides of the NTB to regulate access to the SPADnn registers. The attribute on this register is Read-to-set/Write-1-to-clear. Software will gain ownership of the semaphore by reading a value of 0; if a 1 is returned when the register is read, the semaphore is already owned. Then the owner must clear the register to release the semaphore. It is important to note that this register in no way affects writing to the scratchpad registers by CPUs on either side of the NTB, i.e., it is an honor system.

- The MMIO space of PBAR01 also contains a via into the secondary side PCI configuration space. This area starts at offset 0x500 and contains the entire 4K configuration space. This allows the device driver running on the primary side of the NTB to control the operation of the secondary side. This is useful for some of the usage models.

2.4.2.1 PBAR23/PBAR45

PBAR23 and PBAR45 are used to gain access into the memory map of the system connected to the secondary side of the NTB (the "PCI Express System" on the right side of Figure 5) from the "Intel Xeon Processor-Based System". Each of these BARs can be configured to be any power-of-2 in size from 4KB (2^{12}) to 512GB (2^{39}).

The access path for the Intel Xeon Processor-Based System into the PCI Express System is:

- The Intel Xeon Processor-Based System does a read or a write to either of PBAR23 or PBAR45.

- The NTB hardware captures this read or write packet and converts it into a read or write into the PCI Express system. The translation is done through the registers called PXLAT23 or PXLAT45, depending on which PBARxx is accessed. The address generated into the PCI Express system is calculated:

  PXLATnn + (InputAddress – PBARnn)

- The read or write packet may target system memory on the PCI Express system, or if the PCI Express system supports peer-to-peer, the packet may target another PCIe device.

- The PCI Express system owns the PXLAT23 and PXLAT45 registers, so the software running on that system is responsible for directing the accesses to the appropriate locations.

- The NTB also performs the translations required from system to system
  - 3- or 4-dword PCIe header translations
  - Requestor and completer IDs in the PCIe packets

2.4.3 NTB Secondary

2.4.3.1 NTB Secondary Configuration Space

The secondary side of the NTB exposes a PCI Type 1 configuration space, including all the PCI required registers such as Vendor ID, Device ID and Base Address Registers (BAR).

The configuration space also contains several PCI capability structures, including MSI, MSI-x, and the required PCI Express capability.

The first BAR, SBAR01, is used to control the behavior of the NTB as well as for inter-processor communication. This BAR is memory mapped I/O (MMIO) and has a fixed size of 32KB.

Registers included in this MMIO space include:

- PBAR2LMT and PBAR4LMT – The PCI specification defines BAR sizes to be a power-of-2. These registers are used to place a cap on the size of BAR23 and BAR45 on the primary side of the NTB, i.e., if a 3MB BAR size is desired, the PBARnSZ register would be set to 22 (2^{22} = 4MB) and the PBARnLMT register would be loaded with the appropriate value to cause the BAR to reject any accesses to the addresses between 3MB and 4MB. Registers used to control the primary side of the NTB are not writable from the secondary side of the NTB.

- SBAR2LMT and SBAR4LMT – These registers perform the same function as the PBARnLMT registers, but they affect the BARs on the secondary side of the NTB. These registers are writable from both the primary and secondary sides of the NTB.

- PBAR2XLAT and PBAR4XLAT – These registers are associated with the memory map of the system on the secondary side of the NTB. They are used to direct the accesses to the BARs on the primary side of the NTB into the system connected to the secondary side of the NTB. An example of this can be seen in Figure 6. PBAR23 contains a value of 0x40000 and is 64KB in size. PBAR2XLAT is initialized with a value of 500000. Any access from the Intel Xeon Processor to an address between 0x40000 and 0x4FFFF would be captured by the NTB and converted to an access into the memory of the PCI Express System to its local address of 0x500000 – 0x50FFFF.

- SBAR2XLAT and SBAR4XLAT – These registers are similar to PBARnXLAT registers, but are used to direct accesses into the Intel Xeon Processor-Based System from the PCI Express System.

- SBAR0BASE, SBAR2BASE and SBAR4BASE – These registers are a copy of the three BAR registers located in the configuration space of the secondary side of the NTB. They allow the software running on the primary side of the NTB to modify the physical BAR registers on the secondary side of the NTB. These are included to enable certain usage models where there is no agent on the secondary side of the NTB capable of running PCI enumeration.

- NTBCNTL – From the secondary side, the only bits writable are those that control the snoop attribute for incoming packets.

- PDOORBELL and SDOORBELL – PDOORBELL is written by the CPU on the secondary side of the NTB to generate an interrupt to the CPU on the primary side of the NTB. Only bits 13:0 are available for interrupt generation. Bit 14 is used to notify completion of a cache flush operation described in Section 2.6,Cache Flush Mechanism, and bit 15 generates an interrupt whenever the link state on the PCIe link changes. SDOORBELL is written by the CPU on the primary side of the NTB to generate an interrupt to the CPU on the secondary side of the NTB. For both registers, writing a 1 causes the interrupt to be generated. The CPU getting the interrupt also writes a 1 to clear the bit.

- PDBMASK and SDBMASK – These registers allow interrupt generation to be prevented. Any bit written to a 1 will disable interrupt generation from the corresponding bit in the doorbell registers.

- SPADnn – The NTB has a shared set of sixteen 32-bit scratchpad registers available to software on both CPUs for inter-processor communication.

- SPADSEMA4 – This register can be used by software running on the CPUs on both sides of the NTB to regulate access to the SPADnn registers. The attribute on this register is Read-to-set/Write-1-to-clear. Software will gain ownership of the semaphore by reading a value of 0; if a 1 is returned when the register is read, the semaphore is already owned. Then the owner must clear the register to release the semaphore. It is important to note that this register in no way affects writing to the scratchpad registers by CPUs on either side of the NTB, i.e., it is an honor system.

2.4.3.2 SBAR23/SBAR45

SBAR23 and SBAR45 are used to gain access into the memory map of the system connected to the primary side of the NTB, the Intel Xeon Processor-Based System on the left side of Figure 5, from the PCI Express System. Each of these BARs can be configured to be any power-of-2 in size from 4KB (2^{12}) to 512GB (2^{39}).

The access path for the PCIe system into the Intel Xeon Processor-Based System is:

- PCIe system does a read or a write to either of SBAR23 or SBAR45.

- The NTB hardware captures this read or write packet and converts it into a read or write into the PCI Express System. The translation is done through the registers called SXLAT23 or SXLAT45, depending on which SBARxx is accessed. The address generated into the PCI Express System is calculated:

  SXLATnn + (InputAddress – SBARnn)

- The read or write packet may target system memory on the Intel Xeon Processor-Based System, or if the system supports peer-to-peer, the packet may target another PCIe device. The root ports of the system IIO support peer-to-peer, but if the NTB is connected to a switch port, it may not.

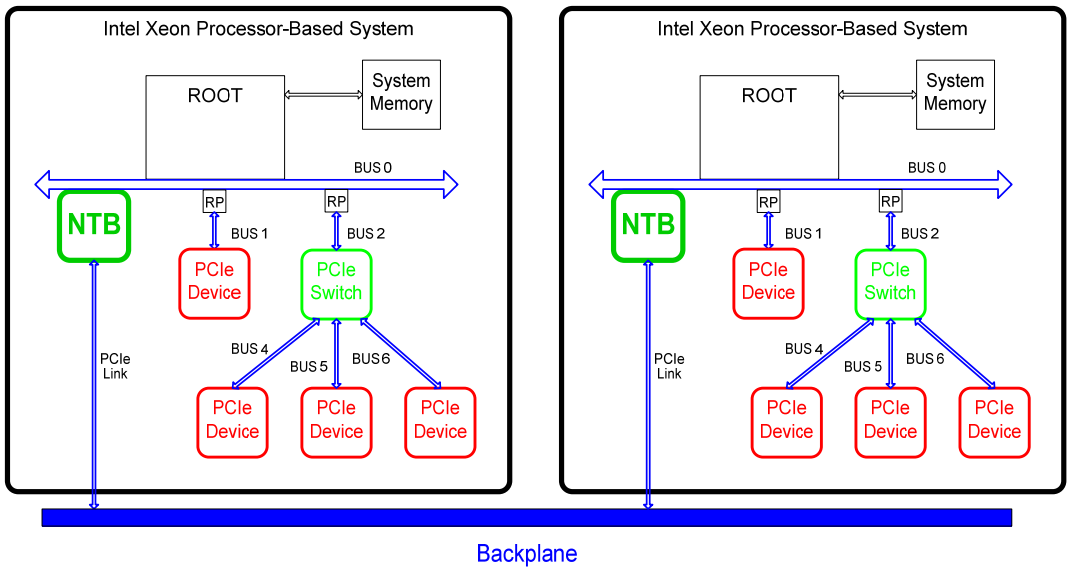
- The Intel Xeon Processor-Based System owns the SXLAT23 and SXLAT45 registers, so the software running on that system is responsible for directing the accesses to the appropriate locations.

- The NTB also performs the translations required from system to system.
  - 3- or 4-dword PCIe header translations
  - Requestor and completer IDs in the PCIe packets

2.5 Considerations for NTB Back-to-Back

While ordinarily the NTB's secondary side PCIe port would be connected to a PCIe port on a second system, the NTB can also be connected to another NTB on the second system as shown in Figure 7. One reason for doing this may be to allow an OEM to connect identical platforms together, eliminating the need for multiple SKUs.

Figure 7. NTB Back-to-Back



2.5.1 Hardware Differences

Connecting two Intel Xeon processor NTBs back-to-back creates an area between them that is not contained in the address map of either Intel Xeon processor, preventing access to the secondary side BAR areas from the processor. The mechanism for creating doorbell interrupts or writing to the scratchpad registers visible to the Intel Xeon processor is to write to NTB secondary side BAR0/1. This can be accomplished in two ways:

- One of BAR2/3 or BAR4/5 could be translated to point to BAR0/1 on the other side of the link, making accesses to that BAR generate PCIe packets destined for BAR0/1 to write to the doorbell or scratchpad registers. But this would waste one of the two BARs available for accessing resource through the NTB.
- The Intel Xeon processor NTB has a second set of registers used for the purpose of writing to doorbell and scratchpad registers. These B2BDoorbell

and B2BScratchpad registers cause the NTB to generate the PCIe packet that is forwarded to the NTB on the other side of the link.

2.5.1.1 Strapping or BIOS Option for Creating Unique Identity

There is a PCIe requirement that on any given PCIe link there is an upstream device and a downstream device. Since in ordinary operation the secondary side of the NTB is a PCIe endpoint, the link from the NTB would connect to an upstream device, such as a root port or a switch port. Now in the case of NTB-NTB configuration, if the secondary side of the NTB of each system are left as downstream devices, the link between two systems cannot train. The Intel Xeon processor NTB solves this problem using either a hardware strap or a BIOS configuration setting to override the strap setting. One of the devices must be configured as upstream and the other downstream before link training can be accomplished. So, theoretically one can connect exactly identical Intel Xeon processor-based systems with the only identity difference being the Strap or corresponding BIOS setting.

2.5.2 Software Differences

In the case of back-to-back connection, certain registers on the secondary side of each of the NTBs must be owned by the device driver on the primary side of the NTB. These include the placement of the BAR registers, translate registers. The default values for these registers are set up to be able to function with nearly any configuration, specifically:

- BAR0/1 is placed at address 0
- BAR2/3 is placed at address 256GB
- BAR4/5 is placed at address 512GB
- The corresponding translate registers on the opposite side of the link are configured to access these BARs

These defaults will function properly for any selection of BAR sizes up to 256GB for BAR2/3 and 512GB for BAR4/5. However, if the BAR sizes are much smaller, and fit within a 32-bit address space, there is a benefit to moving the secondary side BARs to addresses below 4GB. This saves the NTB hardware from having to generate a PCIe packet with a 4-dword header to be able to access the 64-bit BARs on the other side of the link.

Since the secondary side BARs are not visible in the memory map of the Intel Xeon processors on the primary side, generating doorbell interrupts through the NTB when configured in back-to-back requires changes. Either of the two bullet mechanisms described in Section 2.5.1, Hardware Differences, can be used to solve this problem. The device driver hides this difference by using the same call into the driver to write to scratchpad or doorbell registers whether the connection is to a PCIe port or another NTB.

The same two mechanisms are available to reach scratchpad registers. Another difference is that when configured in the back-to-back mode, the scratchpad registers are not shared between the two systems; therefore, use of the semaphore is not

required. System A will write the scratchpad registers on System B, but will read the scratchpad registers on System A, and vice versa.

There is no issue with the order in which the systems are powered on when connected back-to-back (such as the issue that exists when an NTB is connected to a root port).

2.6 Cache Flush Mechanism

Some use cases have a need to know when data sent from one platform to another has been committed to memory, rather than being en route or sitting is the second platform's IIO cache. The NTB provides a mechanism for the sending platform to ensure this happens. After the data is sent from the sending platform to the receiving platform, an MMIO register (WCCNTRL) bit is set by the sending platform. This causes a write into the receiving platform which causes the receiving platform's IIO cache to be flushed into memory. When all the cached data has been flushed into memory, the hardware automatically sends an interrupt back to the sending platform to notify that platform that the flush has been completed.

3 Software

3.1 BIOS

One of the IIO PCIe ports in the Intel Xeon Processor-Based System, either 4-lane or 8-lane, can be configured as either a PCIe root port or an NTB. This decision must be made before BIOS does PCI enumeration because these two device types have very different PCI configuration spaces — the PCIe root port has a Type 1 PCI configuration space and the NTB has a Type 0 PCI configuration space. This is resolved using a register in the PCI configuration space of the default PCIe root port. The configuration space was chosen because it is located at a fixed address, Bus 0, Device 3, Function 0. The BIOS writes a value obtained from some non-volatile storage location into a register called PPD. This write causes the PCI configuration space at that Bus/Device/Function location to be set to the selected device. Possible configurations are: root port, NTB connected to a PCIe port (either a root port or a switch port from another system) or a configuration where the secondary side of the NTB is connected to the secondary side of another NTB in a configuration referred to as back-to-back NTB (B2B).

If the BIOS configures the device as an NTB, several other registers in that same configuration space must also be configured. The NTB has variable-sized BARs on both the primary and secondary sides. These also must be configured prior to PCI enumeration being executed on either side of the NTB. These four single-byte registers (PBAR23SZ, PBAR45SZ, SBAR23SZ and SBAR45SZ) are written with a value of n from: BARSZ = 2^n.

3.2 Software Stack

As illustrated in Figure 8, the software stack sitting on the NTB will consist of an NTB device driver, a Client Driver, and (optionally) a Client Application.

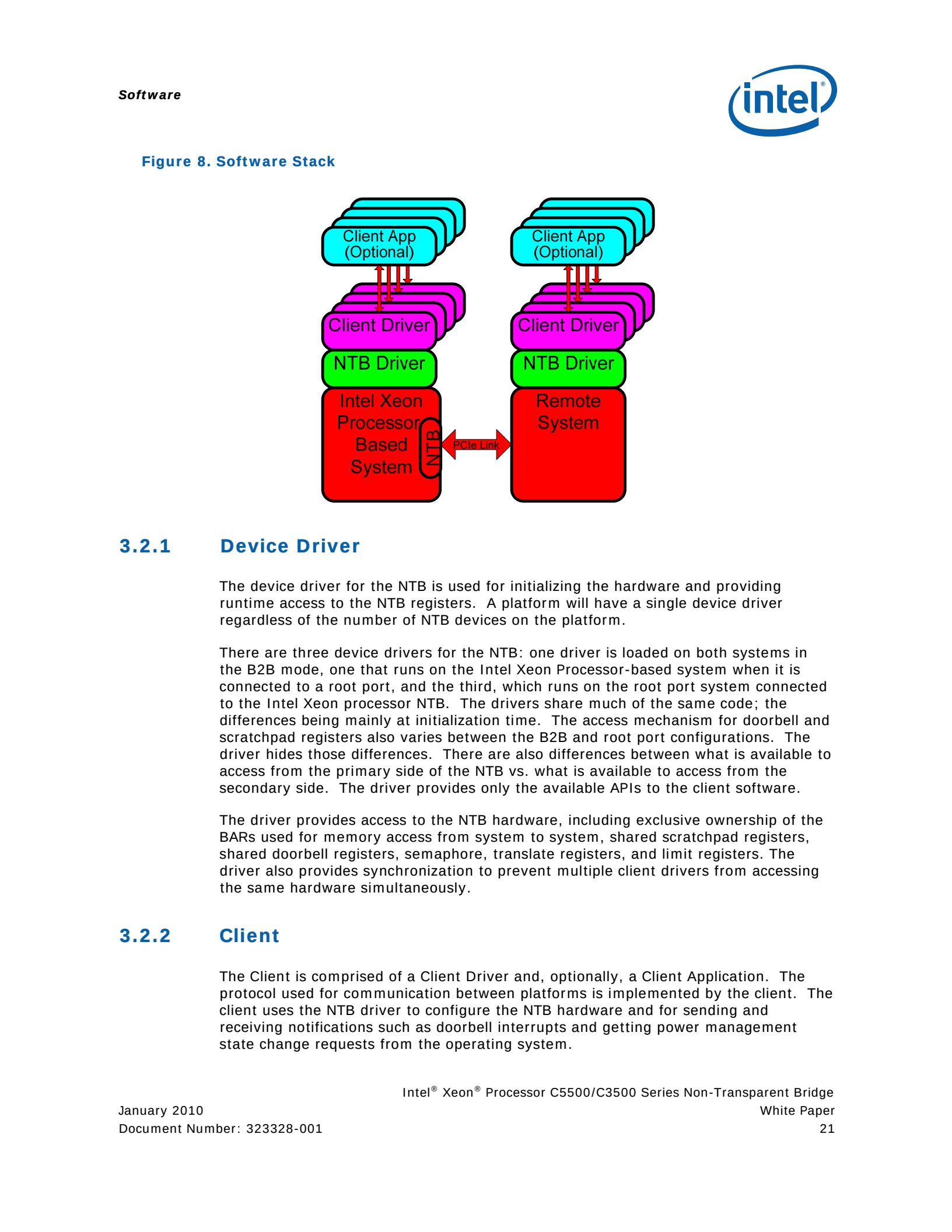


Figure 8. Software Stack

3.2.1 Device Driver

The device driver for the NTB is used for initializing the hardware and providing runtime access to the NTB registers. A platform will have a single device driver regardless of the number of NTB devices on the platform.

There are three device drivers for the NTB: one driver is loaded on both systems in the B2B mode, one that runs on the Intel Xeon Processor-based system when it is connected to a root port, and the third, which runs on the root port system connected to the Intel Xeon processor NTB. The drivers share much of the same code; the differences being mainly at initialization time. The access mechanism for doorbell and scratchpad registers also varies between the B2B and root port configurations. The driver hides those differences. There are also differences between what is available to access from the primary side of the NTB vs. what is available to access from the secondary side. The driver provides only the available APIs to the client software.

The driver provides access to the NTB hardware, including exclusive ownership of the BARs used for memory access from system to system, shared scratchpad registers, shared doorbell registers, semaphore, translate registers, and limit registers. The driver also provides synchronization to prevent multiple client drivers from accessing the same hardware simultaneously.

3.2.2 Client

The Client is comprised of a Client Driver and, optionally, a Client Application. The protocol used for communication between platforms is implemented by the client. The client uses the NTB driver to configure the NTB hardware and for sending and receiving notifications such as doorbell interrupts and getting power management state change requests from the operating system.

There may be multiple clients running on an Intel Xeon Processor-based system, all sharing the same NTB hardware. Clients register with the NTB driver and are granted exclusive use of NTB resources.

3.3 Client Protocols

Client protocols range from the very simple to quite complex. An example of a very simple protocol would be the implementation of a heartbeat. System A would periodically send a doorbell to System B to let System B know it was functioning properly. In the event of a missed heartbeat, System B could take some action, possibly sending a notification of the failure to a management system so that the malfunctioning System A could be replaced.

3.3.1 Failover Client

Failover could be handled in a number of ways and depends upon the systems being used. One scenario between two systems could be implemented as:

System A handles all active requests. As System A receives requests, it sends a copy of each request across the NTB to System B. System A and System B both send and receive heartbeats to each other. If the heartbeat interrupts stop:

- If System A stops getting heartbeats from System B, it could notify system management of the failure and System B would be repaired
- If System B stops getting heartbeats from System A, it would immediately take over processing the requests as well as notify system management of the failure.

3.3.2 Deep Packet Processing Offload

Scalability of compute-intensive workloads can be achieved by offloading CPU-intensive processing from one system to another. This is done by passing payloads that need to be processed from one platform to another through the NTB. Synchronization could be achieved using a fairly simple protocol. Here's an example: System A offloads processing to System B. System B sets up a set of buffers in a ring, each the size of the maximum expected payload. System A forwards the payload to System B and provides a notification, possibly an interrupt. System B processes the payload. The post-processed payload could be sent back to System A using a reverse path through the NTB.

3.3.3 Add-In Card Initialization

The NTB can be used to provide isolation to an add-in card in an Intel Xeon Processor-based system. For example, an add-in card might be able to offer a network packet forwarding service using a microengine or state machine controlling several PCIe Ethernet ports. A cost-saving measure could be attained by allowing the Intel Xeon processor-based card to download microcode and handle initialization on that add-in card rather than adding hardware to the card to enable it to do its own startup. This

setup could go one step further by allowing the add-in card to push those requests it was unable to handle up to the Intel Xeon Processor-based system for disposition.

3.3.4 Daisy Chaining Intel® Xeon® Processor-Based Systems

Multiple Intel Xeon processor systems can be chained together with each being able to have a window into each of the other systems in the chain. This can be done both when connected to a root port or in the back-to-back configuration, though the back-to-back configuration requires a DP system where the other configuration can be done with either UP or DP Systems.

Creating a daisy-chained topology would require careful BIOS configuration as each system must have its NTB device(s) located at specific addresses. And each system on the chain will have complete access to the memory map on each of the other systems.

The limit to the number of systems that may be daisy-chained together is resolved by the equation:

Equation 1. Daisy Chain Width

$$Width = MaxAddressLines - SystemSize + 1$$

Where:

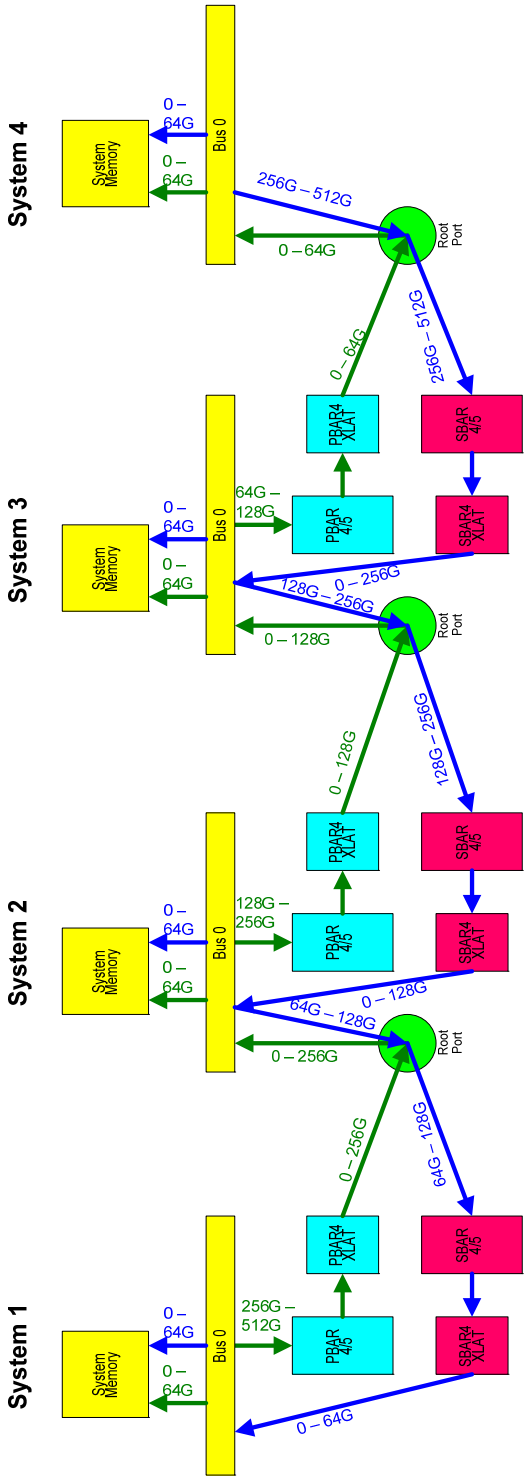
- Width is the number of systems that may be daisy-chained together
- MaxAddressLines is the number of address bits supported by the processor in the systems. The Intel Xeon Processor C5500/C3500 Series supports 40 bits of address.
- SystemSize is n in the equation 2^n = the size of the local window in each system

For example, if each Intel Xeon processor system had a local memory footprint of 64GB, then five systems could be chained together.

Figure 9 illustrates daisy chaining four 64GB systems. In the diagram, all requests emanate from Bus 0, and the addresses being used are listed on the arrows showing which direction the request is flowing.

This paper illustrates daisy chaining of root port connected systems. Systems connected back-to-back can also be daisy-chained.

Figure 9. Daisy-Chained Intel® Xeon® Processor-Based Systems



3.3.4.1 Figure 9 Breakdown

Looking from the CPU on System 1, accesses to the MMIO address range from 256GB to 512GB are accepted by one of the BARs on the NTB. These addresses are then translated by the NTB to addresses 0GB – 256GB and directed into the memory map of System 2.

The translated address range from 0GB to 64GB is directed into the local system memory, i.e., System 1 address 256GB becomes System 2 address 0. Addresses from 64GB to 128GB would also be directed into the memory map of System 2, but should not be used because accesses to this area may cause unpredictable results. Addresses from 128GB – 256GB are directed to the BAR window on the NTB of System 2 by peer-to-peer logic within the Intel Xeon processor switch. These addresses are then translated by the NTB in System 2 to addresses 0GB – 128GB in the System 3 memory map.

So, from the perspective of System 1, addresses from 384GB – 512GB become addresses 0GB – 128GB within the memory map of System 3. Of this range on System 3, 0GB – 64GB is directed into the local system memory of System 3 and the address range from 64GB – 128GB is directed to the NTB BAR on System 3 by peer-to-peer logic within the Intel Xeon processor switch. The NTB then translates this address range to 0GB – 64GB in System 4. This entire range is consumed by the local system memory on System 4.

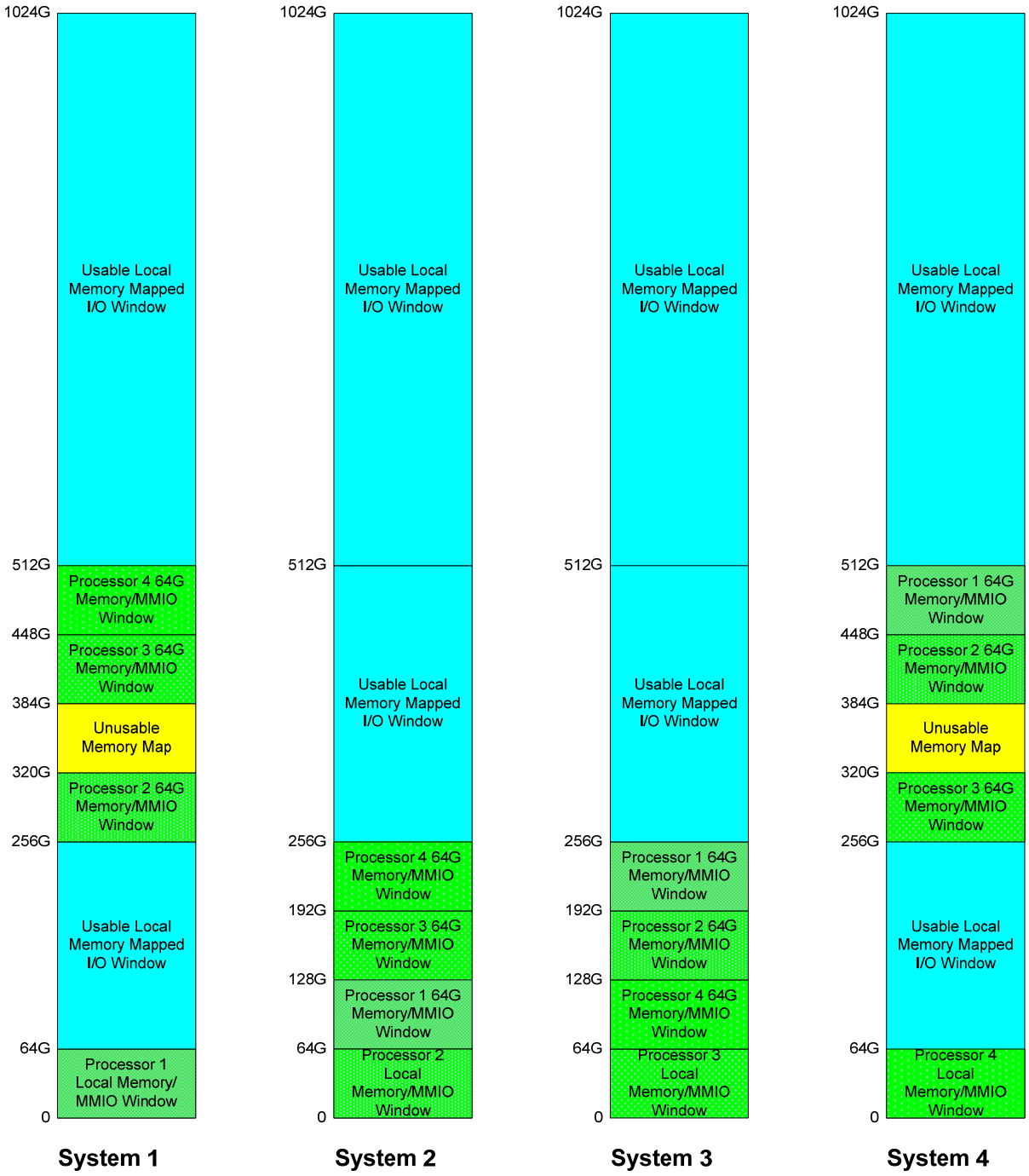
We can see that from the perspective of System 1:

- The local memory map of System 1 can be seen using the address range from 0GB – 64GB.
- The local memory map of System 2 can be seen using the address range from 256GB – 320GB.
- The local memory map of System 3 can be seen using the address range from 384GB – 448GB.
- The local memory map of System 4 can be seen using the address range from 448GB – 512GB.

Similarly each system has a mechanism to reach the entire 64GB system memory on each of the other systems in the daisy chain.

Figure 10 shows the memory map for each of the systems. If the addresses referred to as “Unusable Memory Map” are used, unpredictable results can occur.

Figure 10. Memory Layout



4 Conclusion

The NTB is a highly versatile device that will enable high-speed inter-platform communication.

§

Author

Mark J. Sullivan is a Network Software Engineer with the Intel Architecture Group at Intel Corporation.