

Processor Programming Reference (PPR) for AMD Family 1Ah Model 11h, Revision B0 Processors Volume 5 of 7

Legal Notices

© 2022-2024 Advanced Micro Devices, Inc. All rights reserved.

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of these materials, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of these materials, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by these materials. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.

Trademarks:

AMD, the AMD Arrow logo, and combinations thereof are trademarks of Advanced Micro Devices, Inc.

AGESA is a trademark of Advanced Micro Devices, Inc.

AMD Virtualization is a trademark of Advanced Micro Devices, Inc.

AMD-V is a trademark of Advanced Micro Devices, Inc.

Adobe is a registered trademark of Adobe.

Arm is a registered trademark of Arm Limited.

CXL is a trademark of Compute Express Link Consortium, Inc.

EPYC is a trademark of Advanced Micro Devices, Inc.

Infinity Fabric is a trademark of Advanced Micro Devices, Inc.

Linux is a registered trademark of Linus Torvalds.

MIPI I3C is a registered trademark of MIPI Alliance.

Microsoft is a registered trademark of Microsoft Corporation.

PCI Express is a registered trademark of PCI-SIG Corporation.

PCIe is a registered trademark of PCI-SIG Corporation.

SoundWire is a registered trademark of MIPI Alliance, Inc.

Windows is a registered trademark of Microsoft Corporation.

Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Reverse engineering or disassembly is prohibited.

USE OF THIS PRODUCT IN ANY MANNER THAT COMPLIES WITH THE MPEG ACTUAL OR DE FACTO VIDEO AND/OR AUDIO STANDARDS IS EXPRESSLY PROHIBITED WITHOUT ALL NECESSARY LICENSES UNDER APPLICABLE PATENTS. SUCH LICENSES MAY BE ACQUIRED FROM VARIOUS THIRD PARTIES INCLUDING, BUT NOT LIMITED TO, IN THE MPEG PATENT PORTFOLIO, WHICH LICENSE IS AVAILABLE FROM MPEG LA, L.L.C., 6312 S. FIDDLERS GREEN CIRCLE, SUITE 400E, GREENWOOD VILLAGE, COLORADO 80111.

List of Chapters

Volume 1:

- 1** [Overview](#)
- 2** [Core Complex \(CCX\)](#)

Volume 2:

- 3** [Reliability, Availability, and Serviceability \(RAS\) Features](#)

Volume 3:

- 4** [System Management Unit \(SMU\)](#)

Volume 4:

- 5** [Advanced Platform Management Link \(APML\)](#)
- 6** [SB Temperature Sensor Interface \(SB-TSI\)](#)
- 7** [Host System Management Port \(HSMP\)](#)
- 8** [Data Fabric \(DF\)](#)
- 9** [Unified Memory Controller \(UMC\)](#)

Volume 5:

- 10** [Northbridge IO \(NBIO\)](#)

Volume 6:

- 11** [Fusion Controller Hub \(FCH\)](#)

Volume 7:

- 12** [Reserved](#)

[List of Namespaces](#)

[List of Definitions](#)

Table of Contents

10	Northbridge IO (NBIO)
10.1	Definitions
10.2	IOHUB Core (IOHC)
10.2.1	Peer-to-Peer Support
10.2.1.1	Peer-to-Peer Synchronization
10.2.1.1.1	Peer-to-Peer Status Polling
10.2.1.1.2	Writing Data and Flag to a Peer
10.2.1.1.3	Signaling via the CPU
10.2.1.1.4	Unsupported Peer-to-Peer Synchronization Methods
10.2.1.2	Peer-to-Peer Interaction with PCIe® ACS
10.2.2	Registers
10.2.2.1	IOHC Registers
10.3	I/O Memory Management Unit (IOMMU)

List of Figures

List of Tables

Table 176: Link Definitions

10 Northbridge IO (NBIO)

10.1 Definitions

Table 176: Link Definitions

Term	Description
IOHC	IOHUB Core; the I/O crossbar.
IOMMU	I/O Memory Management Unit.
NBIF	Northbridge Bus Interface, including PCIe® bridges and controllers for internal endpoints.
NTB	Non-transparent bridge. A device that links the memory space of two separate systems together. The processor implements a NTB that connects two systems together using the PCIe interface.

10.2 IOHUB Core (IOHC)

10.2.1 Peer-to-Peer Support

The processor supports passing high bandwidth peer-to-peer memory read and write operations between I/O devices. Any [PCIe](#) device may be a source or target of a P2P operation. Additionally, internal peripherals such as the CCP or NTB may also be the source or target of a P2P operation.

To generate a peer-to-peer request, an I/O device generates a DMA read or write operation containing an [MMIO](#) address rather than a DRAM address.

10.2.1.1 Peer-to-Peer Synchronization

In order to optimize DMA performance to DRAM, the processor limits the methods peer I/O devices may use to synchronize between each other in order to achieve producer/consumer ordering. These methods are a subset of what is allowed for in the [PCIe](#) specification.

10.2.1.1.1 Peer-to-Peer Status Polling

After issuing a set of P2P write operations, a P2P source may ensure that they are flushed to the target by performing a subsequent P2P read operation to the same target. The P2P source may then signal that the peer writes have been completed by setting an internal status register that is polled by the P2P target.

If the P2P source does not have read permissions to access the target, as enforced by the IOMMU, a zero-byte read operation may still be used.

10.2.1.1.2 Writing Data and Flag to a Peer

After issuing a set of P2P write operations, a P2P source may signal the target that the writes have completed by writing a status flag to the target also using a P2P write with RO=0. The target may either locally poll the status flag or the P2P write may trigger hardware that then operates on the received P2P write data.

10.2.1.1.3 Signaling via the CPU

After issuing a set of P2P write operations, a P2P source may signal the target indirectly via the CPU. It may generate an interrupt or write a flag into DRAM that is polled by the CPU. The CPU would then do an [MMIO](#) write to the P2P target to indicate that the P2P writes have been delivered from the P2P source.

10.2.1.1.4 Unsupported Peer-to-Peer Synchronization Methods

The processor does not support the below peer-to-peer synchronization methods.

Case 1:

Any synchronization scheme that relies on completions pushing posted writes in the downstream direction away from the processor is not supported.

In the following example, the read response of the status register in step 3 may pass the P2P writes in step 1.

1. Device A does P2P writes to device B.
2. Device A sets an internal status register to indicate the P2P writes are complete.
3. In parallel, device B polls the status register in device A.

In order to make the case pass, device A may use a flushing read to device B before setting the internal status register in step 2.

Case 2:

Any synchronization scheme that relies on ordering between the writes of one device and the reads and writes of another device is not supported, regardless of whether or not the operations target DRAM or a 3rd I/O device.

In the following example, it is possible that the DMA operations from device B in step 3 may pass the DMA writes in step 1.

1. Device A does DMA writes to DRAM.
2. Device A does a P2P write to device B to indicate that data has been written to DRAM.
3. Once device B observes the write in step 2, device B does DMA reads or writes to the same DRAM locations accessed by device A and requires that these operations will be ordered after the writes in step 1.

In order to make this case pass, device A should perform a flushing read to DRAM before issuing the P2P write to device B in step 2.

Similarly, for the following example it is possible that the P2P operations in step 3 may reach device C before the P2P writes in step 1.

1. Device A does P2P writes to device C.
2. Device A does a P2P write to device B to indicate that data has been written to device C.
3. Once device B observes the write in step 2, device B does P2P reads or writes to device C to operate on the data sent by device A or otherwise assumes that these operations will be ordered after the writes in step 1.

In order to make this case pass, device A should perform a flushing read to device C before issuing the P2P write to device B in step 2.

10.2.1.2 Peer-to-Peer Interaction with PCIe® ACS

The processor [PCIe](#) root ports treat a received completion that does not match a corresponding non-posted request previously issued by the same root port as an unexpected completion. Configurations that generate non-matching completions to the root ports are not supported. Non-matching completions may be sent to a root port during specific combinations of peer-to-peer traffic in conjunction with an external PCIe switch and specific settings of the PCIe ACS

features in the switch. In these cases, completions to peer-to-peer requests do not follow the reverse of the request path back to the requester through the PCIe topology.

Similarly, the processor root ports will trigger a timeout event if the root port issues downstream non-posted request without the corresponding completion returning to the same root port. This can occur whenever PCIe ACS features are enabled such that a function located behind a switch generates a P2P read or atomic targeting an endpoint located behind the same switch, where the request is routed through the PCIe root port, but the completion is directly routed back to the originator without going back through the PCIe root port.

To minimize the number of cases that trigger the above conditions, by default P2P requests issued by the root port have a RequesterID matching that of the root complex. For most cases, this will force completions to be routed back through the root port if its associated request first came from the root port, even if RO is set in the request.

In general, if IOMMU is enabled, software should enable ACS Request Redirection, enable ACS Upstream Forwarding, and disable ACS Completion Redirection in root ports, external switches and multi-function endpoints to ensure that DMA requests are forced up towards the IOMMU for translation. ACS Direct Translated P2P may be enabled in switches to maximize P2P performance by allowing translated requests and their corresponding completions to be routed directly by the switch without first going up to the processor root port.

10.2.2 Registers

10.2.2.1 IOHC Registers

BXXD00F0x0C4 (IOHC::NB_SMN_INDEX_3)	
Read-write. Reset: 0000_0000h.	
_instIOHC0_iohub0_nbio0_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC0_iohub0_nbio0_aliasSMN[NB_BUS_NUM]	
_instIOHC0_iohub0_nbio1_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC2_iohub0_nbio1_aliasSMN[NB_BUS_NUM]	
_instIOHC1_iohub1_nbio0_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC0_iohub1_nbio0_aliasSMN[NB_BUS_NUM]	
_instIOHC1_iohub1_nbio1_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC2_iohub1_nbio1_aliasSMN[NB_BUS_NUM]	
_instIOHC2_iohub2_nbio0_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC1_iohub2_nbio0_aliasSMN[NB_BUS_NUM]	
_instIOHC2_iohub2_nbio1_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC3_iohub2_nbio1_aliasSMN[NB_BUS_NUM]	
_instIOHC3_iohub3_nbio0_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC1_iohub3_nbio0_aliasSMN[NB_BUS_NUM]	
_instIOHC3_iohub3_nbio1_aliasHOST; BXXD00F0x0C4; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC3_iohub3_nbio1_aliasSMN[NB_BUS_NUM]	
_instIOHC0_iohub0_nbio0_aliasSMN; NBCFG0x000000C4; NBCFG0=13B0_0000h	
_instIOHC2_iohub2_nbio0_aliasSMN; NBCFG1x000000C4; NBCFG1=13C0_0000h	
_instIOHC0_iohub0_nbio1_aliasSMN; NBCFG2x000000C4; NBCFG2=13D0_0000h	
_instIOHC2_iohub2_nbio1_aliasSMN; NBCFG3x000000C4; NBCFG3=13E0_0000h	
_instIOHC1_iohub1_nbio0_aliasSMN; NBCFG4x000000C4; NBCFG4=1D40_0000h	
_instIOHC3_iohub3_nbio0_aliasSMN; NBCFG5x000000C4; NBCFG5=1D50_0000h	
_instIOHC1_iohub1_nbio1_aliasSMN; NBCFG6x000000C4; NBCFG6=1D60_0000h	
_instIOHC3_iohub3_nbio1_aliasSMN; NBCFG7x000000C4; NBCFG7=1D70_0000h	
Bits	Description
31:0	NB_SMN_INDEX_3. Read-write. Reset: 0000_0000h. Index value for SMN Index/Data pair access. Register access using these reg pairs will have security level 7.

BXXD00F0x0C8 (IOHC::NB_SMN_DATA_3)

Read-write. Reset: 0000_0000h.

_instIOHC0_iohub0_nbio0_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC0_iohub0_nbio0_aliasSMN[NB_BUS_NUM]
 _instIOHC0_iohub0_nbio1_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC2_iohub0_nbio1_aliasSMN[NB_BUS_NUM]
 _instIOHC1_iohub1_nbio0_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC0_iohub1_nbio0_aliasSMN[NB_BUS_NUM]
 _instIOHC1_iohub1_nbio1_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC2_iohub1_nbio1_aliasSMN[NB_BUS_NUM]
 _instIOHC2_iohub2_nbio0_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC1_iohub2_nbio0_aliasSMN[NB_BUS_NUM]
 _instIOHC2_iohub2_nbio1_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC3_iohub2_nbio1_aliasSMN[NB_BUS_NUM]
 _instIOHC3_iohub3_nbio0_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC1_iohub3_nbio0_aliasSMN[NB_BUS_NUM]
 _instIOHC3_iohub3_nbio1_aliasHOST; BXXD00F0x0C8; BXX=IOHC::NB_BUS_NUM_CNTL_instIOHC3_iohub3_nbio1_aliasSMN[NB_BUS_NUM]
 _instIOHC0_iohub0_nbio0_aliasSMN; NBCFG0x000000C8; NBCFG0=13B0_0000h
 _instIOHC2_iohub2_nbio0_aliasSMN; NBCFG1x000000C8; NBCFG1=13C0_0000h
 _instIOHC0_iohub0_nbio1_aliasSMN; NBCFG2x000000C8; NBCFG2=13D0_0000h
 _instIOHC2_iohub2_nbio1_aliasSMN; NBCFG3x000000C8; NBCFG3=13E0_0000h
 _instIOHC1_iohub1_nbio0_aliasSMN; NBCFG4x000000C8; NBCFG4=1D40_0000h
 _instIOHC3_iohub3_nbio0_aliasSMN; NBCFG5x000000C8; NBCFG5=1D50_0000h
 _instIOHC1_iohub1_nbio1_aliasSMN; NBCFG6x000000C8; NBCFG6=1D60_0000h
 _instIOHC3_iohub3_nbio1_aliasSMN; NBCFG7x000000C8; NBCFG7=1D70_0000h

Bits Description

31:0 **NB_SMN_DATA_3**. Read-write. Reset: 0000_0000h. Data value for [SMN](#) Index/Data pair access. Register access using these reg pairs will have security level 7.

IOHCMISC[0...7]x00000044 (IOHC::NB_BUS_NUM_CNTL)

Read-write. Reset: 0000_0000h.

GNB Bus Number Control.

_instIOHC0_iohub0_nbio0_aliasSMN; IOHCMISC0x00000044; IOHCMISC0=13B1_0000h
 _instIOHC2_iohub2_nbio0_aliasSMN; IOHCMISC1x00000044; IOHCMISC1=13C1_0000h
 _instIOHC0_iohub0_nbio1_aliasSMN; IOHCMISC2x00000044; IOHCMISC2=13D1_0000h
 _instIOHC2_iohub2_nbio1_aliasSMN; IOHCMISC3x00000044; IOHCMISC3=13E1_0000h
 _instIOHC1_iohub1_nbio0_aliasSMN; IOHCMISC4x00000044; IOHCMISC4=1D41_0000h
 _instIOHC3_iohub3_nbio0_aliasSMN; IOHCMISC5x00000044; IOHCMISC5=1D51_0000h
 _instIOHC1_iohub1_nbio1_aliasSMN; IOHCMISC6x00000044; IOHCMISC6=1D61_0000h
 _instIOHC3_iohub3_nbio1_aliasSMN; IOHCMISC7x00000044; IOHCMISC7=1D71_0000h

Bits Description

31:24 Reserved.

23:16 **NB_SEGMENT**. Read-write. Reset: 00h. Specifies the number of the NBIO segment in a multi-segmented system.

15:9 Reserved.

8 **NB_BUS_LAT_Mode**. Read-write. Reset: 0.

Description: NBIO bus number is specified by NB_BUS_NUM.

0 = Local bus number of NBIO is capture from any type 0 configuration request.

1= Use the NB_BUS_NUM to decode for configuration cycles targeting the NBIO bus.

7:0 **NB_BUS_NUM**. Read-write. Reset: 00h. Specifies the number of the NBIO local bus when NB_BUS_LAT_mode is set.

10.3 I/O Memory Management Unit (IOMMU)

The I/O Memory Management Unit (IOMMU) extends the AMD64 system architecture by adding support for address translation and system memory access protection on DMA transfers from peripheral devices.