Freescale Semiconductor
Application Note

Document Number: AN3781
Rev. 0, 06/2009

Utilizing Extra FC Credits for PCI Express Inbound Posted Memory Write Transactions in PowerQUICC III™ Devices

This application note explains the procedures to utilize the extra FC (Flow Control) credits for PCI Express inbound posted memory write transactions, which is currently a hidden feature for all of the ×8-capable Power QUICC III™ devices.

NOTE

This document is applicable for the following devices as well:

- MPC8610
- MPC8640
- MPC8640D
- MPC8641
- MPC8641D

Contents

1. Introduction 2
2. Developing the Desired code .2
3. Porting the Code to a Real World Application Environment 4
4. Revision History 4

1 Introduction

As described in the PCI Express chapter of the reference manual, current Power QUICC III™ devices feature Posted Header (PH) of 4 and Posted Data (PD) of 64 as FC credits initially advertised for PCI Express inbound memory write transactions, where 64 PD credits are equivalent to total data payloads of 4 transaction layer packets (TLPs) with 256-byte each as the maximum payload size.

These initial FC credits advertised are plenty for all PCI Express x1, x2 and x4 applications. When x8 link width is implemented, some performance demanding PCI Express application might need some extra FC credits (PH and PD) for inbound posted memory write transaction.

There is currently some hidden receiver buffer space available to satisfy this demand. When the SRIO interface of a Power QUICC III™ device is not enabled on the same SerDes port where x8 is selected as PCI Express link width, some hidden buffer space can be made available to the PCI Express receiver. This allows the x8 PCI Express interface to advertise and utilize 6 as PH and 96 as PD (equivalent to the total payload of 6 transaction layer packets (TLPs) with 256-byte each as the maximum payload size), therefore makes the aforesaid demand possible.

The following sections describe the detailed procedure to follow when enabling the extra receiver buffer space is desired. Two service registers are involved to override the default posted FC credit initially advertised.

2 Developing the Desired code

As a brief reference, the uBoot commands utilized in the following sample code include both “md (memory read)” and “mm (memory write)” with the syntax below:

- `md 12345678 // Memory Read to local space 0x12345678. Device returns result as 0xAABBCCDD.`
  `12345678: AABBCCDD // Blue indicates system’s console screen display`
- `mm 12345678 // Memory Write to local space 0x12345678 with write value of 0xFFFFFFFF`
  `12345678: AABBCCDD ? FFFFFFFF // Blue indicates system’s console screen display`

The purpose of the procedure below is to ensure that the device with the code implemented behaves correctly as intended such that the similar code could be ported to real world application environment later.

When SRIO is disabled, to enable the extra receiver buffer space for a x8 link width PCI Express controller, the following two events have to happen:

- The content of a service register located at the PCI Express controller's memory-mapped address offset 0xF10 needs to be changed from its reset default value so that 6 PH and 96 PD can be advertised and utilized for all inbound PCI Express posted memory write transactions.
- After the above modification, the other service register located at offset 0xF00 has to be tweaked to reset the corresponding PCI Express controller logic before the modification can take effect.

To determine the full memory-mapped address of the service register located at offset 0xF10 or 0xF00, the following guideline needs to be followed:

- First of all, the device's CCSRBAR has to be determined.
- Second, the correct PCI Express controller with x8 link width implemented needs to be identified.

Please note that for device with more than one PCI Express controllers that could be configured in x8 link width via the SerDes I/O Port Selection during POR configuration, the following rule applies when determining which PCI Express controller is allowed to utilize the extra receiver buffer space:

— If SRIO interface is disabled on the device, both x8 PCI Express controllers can utilize the extra receiver buffer space;

— If SRIO is enabled on SerDes 2 for example, the PCI Express controller not sharing the same SerDes port (the one mapped to SerDes 1 in this case) is allowed to utilize the extra receiver buffer space, as long as this PCI Express controller does not communicate with the enabled SRIO interface.

- Third, the memory-mapped block base address of the identified PCI Express controller needs to be determined.

Once all above is identified, the full address of the service register located at offset 0xF10 or 0xF00 can be determined by appending all these three addresses together, such as “CCSRBAR + PCI Express Controller's Block Base Address + 0xF10 or 0xF00”.

For example, the following example assumes that one of the Power QUICC III™ devices’ CCSRBAR is 0xF8000000 and the x8 link width PCI Express controller is PCI Express Controller 1 with block base address offset 0x0_A000. Therefore, the full address of the 0xF10 service register is 0xF800AF10. Users can figure out the correct address for their 0xF10 or 0xF00 service register in a similar fashion to reflect their actual application environment.

The following sample code sequence uses uBoot command to demonstrate how to alter the content of this 0xF10 offset service register and make the change take effect.

```
md F800AF10 // Memory Read to the PCI Express Controller’s memory-mapped address offset 0xF10
F800AF10: C8800000 // Device returns 0xC8800000 as the reset default of this register

mm F800AF10 // Memory Write to controller’s memory-mapped address offset 0xF10 with 0xE8806000
F800AF10: C8800000 ? E8806000 // Change 0xF10’s content to make it use 6 PH & corresponding PD.

md F800AF10 // Memory Read to 0xF10 to verify whether the previous write takes place
F800AF10: E8806000 // Device should returns 0xE8806000 on its console display

// The following code sequence assumes that the reset default value of the PCI Express
Controller memory-mapped offset 0xF00 is 0x80400080. In reality, the reset default of offset
0xF00 could be device-dependent. Please modify and implement the code accordingly for the
actual application.
md F800AF00 // Memory Read to the PCI Express Controller’s memory-mapped address offset 0xF00
F800AF00: 80400080 // Assume the device returns 0x80400080 as the reset default

mm F800AF00 // Memory Write to the PCI Express controller’s offset 0xF00. Set bit 4 only,
F800AF00: 80400080 ? 88400080 // while preserving the reset default values of all other bits.

Wait 1 msec;

mm F800AF00 // Memory Write to the PCI Express controller’s offset 0xF00. Clear bit 4 only,
F800AF00: 88400080 ? 80400080 // while preserving the reset default values of all other bits.
```

With the code sequence above, the related PCI Express Controller should be reset. The link will be re-trained with 6 PD and 96 PH shown up as the initial FC credits advertised for inbound Posted Memory Write TLPs.

3 Porting the Code to a Real World Application Environment

After the correct code is developed to reflect the correct x8 PCI Express controller identified with the associated memory-mapped address for service registers based on the application’s CCSRBAR, as well as the correct reset default value is determined for 0xF00 offset during the above code development, all the memory read steps in the above code sequence can be removed. In other words, only the correct memory writes are needed to be ported to a real world application’s initialization code.

If all the assumption in the previous section stays intact, the previous code sequence can be simplified into the following, which can then be easily ported to as part of an application’s initialization code:

```
mm F800AF10 // Memory Write to controller’s memory-mapped address offset 0xF10 with 0xE8806000
F800AF10: C8800000 ? E8806000 // Change 0xF10’s content to make it use 6 PH & corresponding PD.

mm F800AF00 // Memory Write to the PCI Express controller’s offset 0xF00. Set bit 4 only,
F800AF00: 80400080 ? 88400080 // while preserving the reset default values of all other bits.

Wait 1 msec;

mm F800AF00 // Memory Write to the PCI Express controller’s offset 0xF00. Clear bit 4 only,
F800AF00: 88400080 ? 80400080 // while preserving the reset default values of all other bits.
```

4 Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	06/2009	Initial public release.

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN3781
Rev. 0
06/2009