SYNOPSYS®

DesignWare® Cores

PCIe eDMA Channel Grouping Application Note

PCI EXPRESS ENDPOINT (EP)
PCI EXPRESS ROOT PORT (RC)
PCI EXPRESS DUAL MODE (DM)

DesignWare Intellectual Property Core

1 Introduction

This application note describes how to use the PCIe eDMA Channel Grouping feature.

Motivation for this enhancement is related to an usecase where there is bandwidth competition between the fetch of a linked list descriptor and the eDMA data transfer. For a system where fetching a descriptor has a high latency and the eDMA data transfer size is small this issue is aggravated since eDMA executes 1 transfer in sequence.
For a single channel implementation, the current eDMA executes the task of fetching descriptor and executing the data transfer in sequency originating wait cycles between fetching and waiting for descriptor and reading data and waiting for data. The following figure illustrate a single write channel sequences to accomplish data transfer.

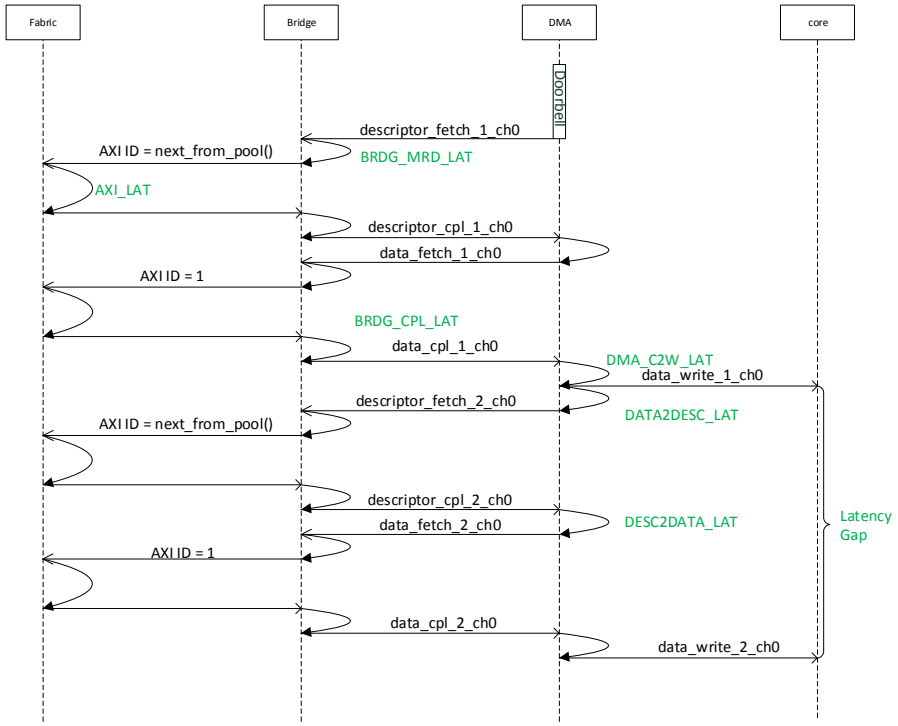

Figure 1 - Single Write Channel sequence

This feature introduces a technique called ‘Channel Grouping’ that uses the existing eDMA architecture, but groups several channels to execute a single transfere. that would enable eDMA to utilize the maximum bandwidth available even for short transfer sizes and huge application latencies.

The concept is to categorize eDMA transfers on schedule as high B/W and low B/W transfers. ‘N’ channels would be grouped together to handle high B/W transfers and a single channel for low B/W transfers. The number of channels to be grouped ‘N’, depends on the application latency, element sizes and B/W target. A data transfer ordering is assured for the grouped channels. The implementation of this concept involves changes in both hardware and software.

2 Registers

2.1 DMA_WR_EN_CHGROUP_OFF

- **Description:** DMA Write Engine Channel Gouping register
- **Size:** 32 bits
- **Offset:** B+80100

BITS	NAME	MEMORY ACCESS	DESCRIPTION
[31:17]	Reserved	R	Reserved for future use. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)
[16]	DMA_WR_ENG_CHGRP_EN	R/W	Enable Write Engine Channel Grouping. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)
[15:8]	DMA_WR_ENG_CHGRP_END	R/W	Ending channel number for Write Engine Channel Group. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)
[7:0]	DMA_WR_ENG_CHGRP_START	R/W	Starting channel number for Write Engine Channel Group. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)

2.2 DMA_RD_EN_CHGROUP_OFF

- **Description:** DMA Read Engine Channel Gouping register
- **Size:** 32 bits
- **Offset:** B+80104

BITS	NAME	MEMORY ACCESS	DESCRIPTION
[31:17]	Reserved	R	Reserved for future use. **Value After Reset:** 0x0

			Exists: (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)
[16]	DMA_RD_ENG_CHGRP_EN	R/W	Enable Read Engine Channel Grouping. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)
[15:8]	DMA_RD_ENG_CHGRP_END	R/W	Ending channel number for Read Engine Channel Group. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)
[7:0]	DMA_RD_ENG_CHGRP_START	R/W	Starting channel number for Read Engine Channel Group. **Value After Reset:** 0x0 **Exists:** (CC_DMA_ENABLE && (CC_NUM_DMA_WR_CHAN>1 || CC_NUM_DMA_RD_CHAN>1) && AMBA_INTERFACE>1)

3 Features / Limitations

The eDMA Channel Grouping includes the following features and limitations:

- This feature can be used only when you enable Linked List operation, that is, set Linked List Enable field of DMA Channel Control 1 Register (`DMA_CH_CONTROL1_OFF_WRCH_0`) to 1'b1.
- Only continuous channels can be grouped and `DMA_WR_ENG_CHGRP_END` > `DMA_WR_ENG_CHGRP_START` always. For example,
  - Gouping Channels 0 to 3 – (Channel 0, 1, 2 and 3) is a valid combination. This can be archieved by setting `DMA_WR_ENG_CHGRP_START` to 0 and `DMA_WR_ENG_CHGRP_END` to 3;
  - Grouping Channels 7 to 2 (Channel 7, 0, 1, 2) is not a valid combination;
  - Grouping Channels 2, 3, and 5 is not allowed.
- If you want to split a DMA transfer across 'N' channels, ensure that the Linked List Size is a multiple of the 'N'. If the Linked List is not a multiple of 'N' you can:
  - Add Data Elements with transfer size zero to the Linked List;
  - Re-arrange the Data Element transfer size to obtain a linked list which is a multiple of 'N'.
- In a Linked List used by channels grouped the following rules apply:
  - Data Element transfer size can differ;
  - The DMA controller supports last Data Elements with transfer size 0;
  - Each Linked List pointer (`DMA_LLP_[LOW|HIGH]_OFF_[WR|RD]CH_[0..7]`) of group channels must be initialized with the first Data Element address;
  - The element pointer address is always incremented by 'N' elements to fetch the next element for the grouped channels while processing, this is automatically performed by the DMA controller;
  - The Cycle bit (cb) should be the same for all elements of the Linked List;
  - The end-of-linked list condition (ccs!=cb) should be at the same position in all the elements.
- Cycle bit for n^{th} element of `DMA_WR_ENG_CHGRP_START` determines if the n^{th} element of all the other channels in the group are valid. The recycling of elements should always be in the order `DMA_WR_ENG_CHGRP_END` to `DMA_WR_ENG_CHGRP_START` to avoid race conditions between the producer and consumer.
- The channel weight programmed for all channels in the group needs to be the same.
- The channel that is grouped and in operation can not be doorbelled individually.
- The `DMA_[WR|RD]_EN_CHGROUP_OFF` register or context registers of any of the channels grouped should not be written while grouped channels are in operation.
- The Enable field in `DMA_[WR|RD]_EN_CHGROUP_OFF` should be set to zero before any of the channels between ChGrpStart & ChGrpEnd are to be used individually.

4 Software Programming

When the application needs a high bandwidth transfer, it splits the M elements on the linked list into 'N' symmetric Linked Lists each with M/N Elements. Where 'N' is the number of channels grouped and M the number of elements on the original list. The following example explains your application can use the Channel Grouping feature.

Basic Example

Consider a Linked List with 7 Data Elements + 1 Link Element (cycles back to 1st Data Element) and a requirement to group 3 channels to satisfy target bandwidth. The Linked List would have to be built as shown in Figure 2.

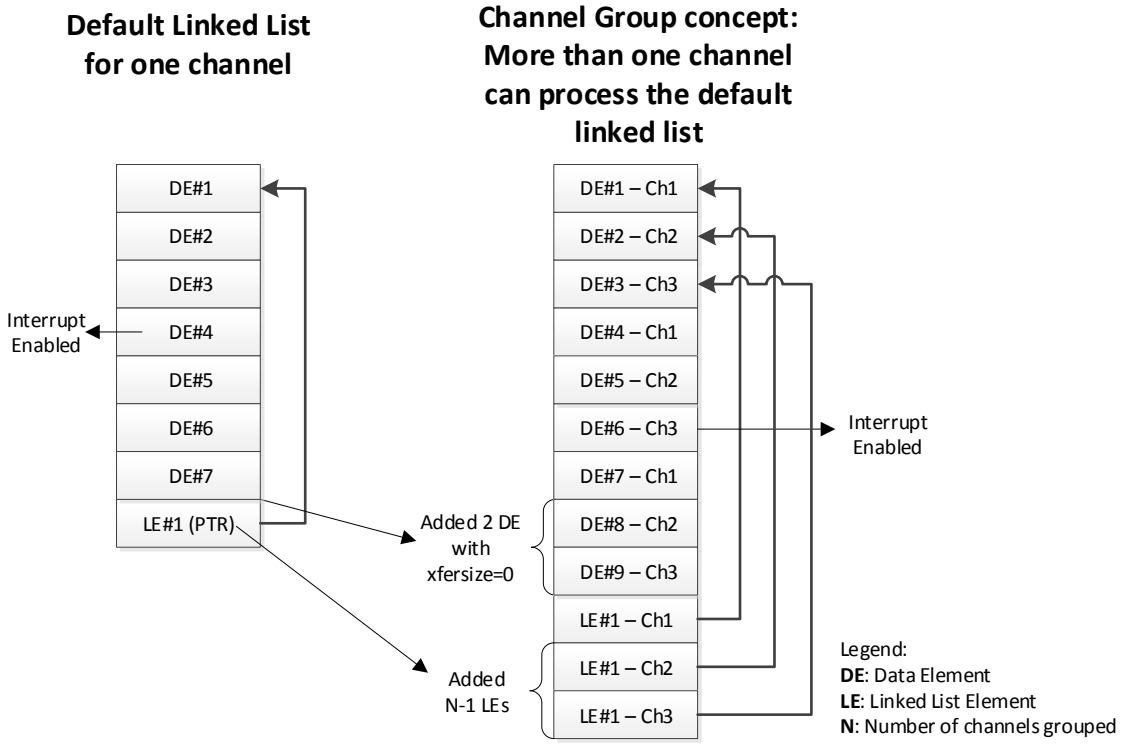

Figure 2 - Channel Grouping Linked List

It's recommended to enable the interrupt only for the last Data Element of the grouped channel. In Figure 2 corresponds to the DE#6 – Ch3.

Your application software is responsible to build and recycle the Linked List as described in Figure 2. Your application must adhere to following sequence to ensure proper functioning of the channel grouping feature:

- The number of Data Elements in the Linked List must be a multiple of 'N' channels. You can add new Data Elements with transfer size zero to the Linked List to make the data elements a multiple of N, or split the transfer size Data Element.
- The original Linked List must be expanded by adding the required number of Les, one for each channel, in this case, adding two more Linked Elements, LE#1 – Ch2 and LE#1 – Ch3.
- Application sets up N (ex: 3) channels by writing the context pointer (`DMA_LLP_LOW|HIGH_OFF_[WR|RD]CH_x`) of the Linked List first Data Element location for each channel. Example: Ch1 → DE#1, Ch2 → DE#2, Ch3 → DE#3.
- Application sets up the Channel Group register so that N channels are sequenced, say Channel 1 to 3.

- You application doorbells the DMA controller on the lowest of the 3 channels, which is Channel 1 in this case.

Considering the eDMA Write Engine, the initial set up sequence would look like Figure 3.

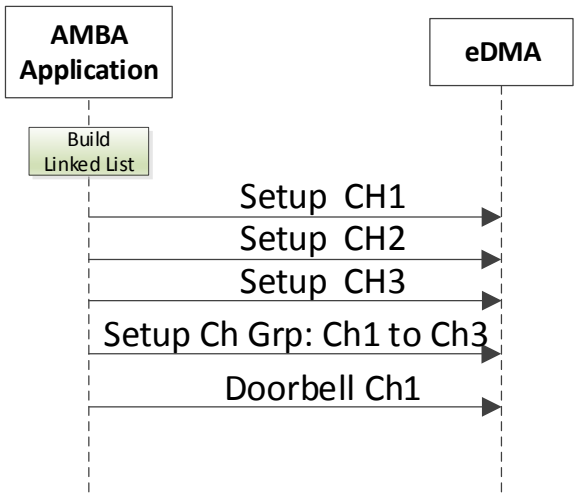

Figure 3 – Setup Channel Grouping

```
   DMA_WRITE_ENGINE_EN_OFF     = 32'h0000_0001
Setup Channel 1:
   DMA_CH_CONTROL1_OFF_WRCH_0 = 32'h0000_0200
   DMA_CH_CONTROL2_OFF_WRCH_0 = 32'h0000_0000
   DMA_LLP_LOW_OFF_WRCH_0      = 32'h0000_8000
   DMA_LLP_HIGH_OFF_WRCH_0     = 32'h0000_0000
Setup Channel 2:
   DMA_CH_CONTROL1_OFF_WRCH_1 = 32'h0000_0200
   DMA_CH_CONTROL2_OFF_WRCH_1 = 32'h0000_0000
   DMA_LLP_LOW_OFF_WRCH_1      = 32'h0000_8018
   DMA_LLP_HIGH_OFF_WRCH_1     = 32'h0000_0000
Setup Channel 3:
   DMA_CH_CONTROL1_OFF_WRCH_2 = 32'h0000_0200
   DMA_CH_CONTROL2_OFF_WRCH_2 = 32'h0000_0000
   DMA_LLP_LOW_OFF_WRCH_2      = 32'h0000_8030
   DMA_LLP_HIGH_OFF_WRCH_2     = 32'h0000_0000
Setup Channel Group:
   DMA_WRITE_ENGINE_CHGROUP_OFF = 32'h0001_0300
Doorbell Channel 1:
   DMA_WRITE_DOORBELL_OFF      = 32'h0000_0000
```

5 System Hardware

When the application performs the Channel Group doorbell, considering the basic ecample, the eDMA proceeds with the following sequence events:

- After Channel 1 doorbell, the eDMA automatically doorbells the other channel/s (2,3)
- eDMA issues Descriptor Memory Reads for Channel 1, 2 and 3 respectively.
  eDMA starts performing Data Memory Reads for Channel 1 and channel 2, 3 are hold.
- After all the Data Memory Reads for the Channel 1
  - eDMA reads the next descriptor for Channel 1 and,
  - start issueing Data Memory Reads for Channel 2.
- As completions for Channel 1 are received the respective Memory Writes are issued on the Wire.
- The Memory Reads and Completions always follow the sequence:

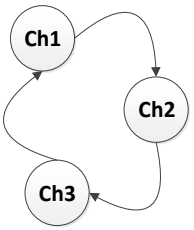

Figure 4 - Channel Grouping sequence

- The traffic would look like Figure 5.

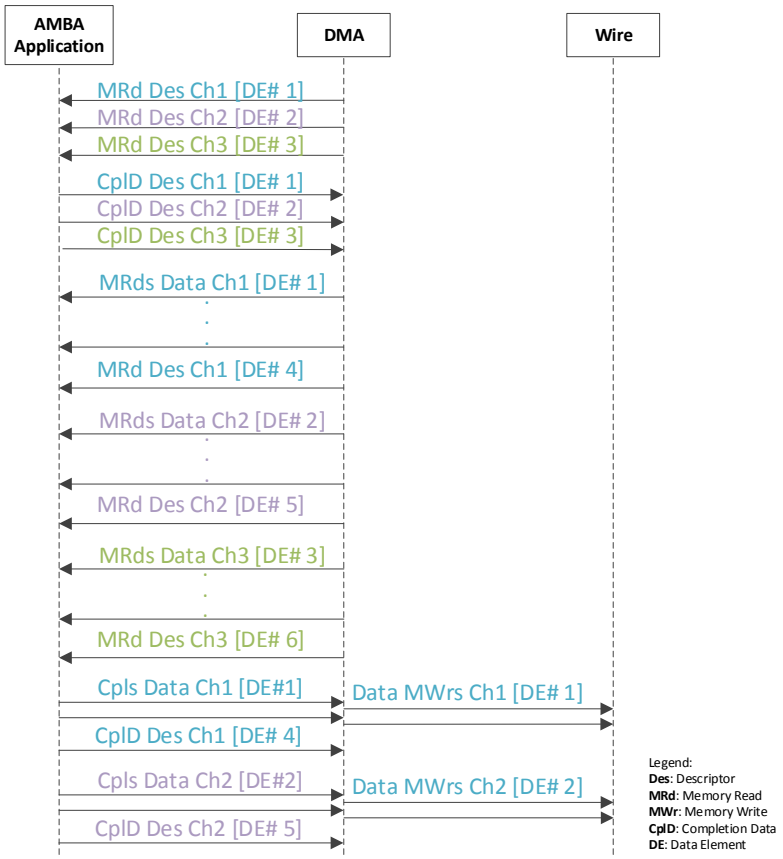

Figure 5 - Grouped Channel Traffic Overview

Each of the channels grouped stops on the Channel Stop condition (ccs!=cb).

- For successful operation, the done interrupt status bit corresponding to the channel that processed the element is set. (`DMA_[WRITE|READ]_INT_STATUS_OFF` status register)
- In the case of the done interrupt being a 'watermark' interrupt, the application can read the DMA Linked List Pointer of the channel whose interrupt bit is set and recycle the elements until this location. Considering the example, if Data Element #6 has the watermark interrupt, `DMA_[WRITE|READ]_INT_STATUS_OFF[3]` will be set. So, the software can read Channel 1's DMA Linked List Pointer and recycle Data Elements: DE#1-Ch1, DE#2-Ch2, DE#3-Ch3, DE#4-Ch1, DE#5-Ch2 and DE#6-Ch3.

6 Recommendations

- To have maximum performance possible, the weights of all the channels in the group should be set to 31.
- Enabling done interrupt only for n^{th} element of `DMA_[WR|RD]_ENG_CHGRP_END` will ensure that the application is triggered with interrupt only after n^{th} element of all the channels in the group has been processed successfully.