ARM® CoreLink™ QoS-400 Network Interconnect Advanced Quality of Service

Revision: r0p2

Supplement to ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual

ARM®

ARM DSU 0026C (ID121113)

ARM CoreLink QoS-400 Network Interconnect Advanced Quality of Service

Supplement to ARM CoreLink NIC-400 Network Interconnect Technical Reference Manual

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
02 July 2012	A	Non-Confidential	First release for r0p0
30 April 2013	B	Non-Confidential	First release for r0p1
11 December 2013	C	Non-Confidential	First release for r0p2

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Product Status

The information in this document is final, that is for a developed product.

Web Address

`http://www.arm.com`

Contents

ARM CoreLink QoS-400 Network Interconnect Advanced Quality of Service Supplement to ARM CoreLink NIC-400 Network Interconnect Technical Reference Manual

Chapter 1
Introduction

This chapter introduces the CoreLink QoS Network Interconnect Advanced Quality of Service. It contains the following sections:

1.1 About the product

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service is an extension to the CoreLink NIC-400 Network Interconnect base product and provides programmable QoS facilities for attached AMBA masters. See Figure 2-1 on page 2-2 for a block diagram that shows a NIC-400 design that contains QoS-400 regulators.

1.1.1 Interconnect QoS and AMBA Designer

AMBA Designer contains QoS options for *Interface Block* (IB) and *AXI Slave Interface Block* (ASIB) nodes. Selecting an option causes the corresponding QoS regulator to be rendered as part of the IB or ASIB. The default is for no regulators to be rendered.

1.2 Compliance

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service is a component that works with the CoreLink NIC-400 Network Interconnect. It complies with AMBA® AXI4, AXI3, AHB-Lite, and APB protocols. The QoS-400 is compatible with all versions of NIC-400.

See the *ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual* for information on the CoreLink™ NIC-400 Network Interconnect.

1.3 Key features

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service has the following features:

- Programmable maxima for outstanding transactions as follows:
  - Separate maxima for read and write requests.
  - Combined maximum for all requests.
  - Fractional value to provide finer control.
- Regulation of read and write request-issuing rates to meet programmed traffic specifications as follows:
  - Separate regulation for read and write requests.
  - Combined regulation for all requests.
- Regulation of read and write request QoS values to target a programmed transaction latency.
- Regulation of read and write request QoS values to target a programmed address request latency:
  - The targeted latency indirectly sets the period.
- Low gate count:
  - You can configure the QoS facilities individually for each ASIB and IB.
  - Efficient measurement of transaction latency.
- Low power consumption, with no dynamic power consumed when the regulators are disabled, except for clock power.
- No cycles of latency added to requests when inactive.

1.4 Interfaces

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service is not a stand-alone product, but integrates with the ASIB or IB modules of the CoreLink NIC-400 Network Interconnect. The CoreLink QoS-400 Network Interconnect Advanced Quality of Service uses the **awqos** and **arqos** signals that you can configure on the ASIB or *AXI Master Interface Block* (AMIB) of the CoreLink NIC-400 Network Interconnect.

Note

The QoS-400 programmers interface is integrated into the *Global Programmers View* (GPV) for the NIC-400 base product. See the *ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual* for additional information.

1.5 Configurable options

You can enable the following QoS-400 options in AMBA Designer, using the AMBA Designer *Graphical User Interface* (GUI):

- Transaction rate regulation.
- Outstanding transaction regulation.
- Transaction or address latency regulation.

1.6 Test features

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service contains no test features.

1.7 Product documentation, design flow, and architecture

The product documentation, design flow, and architecture for the CoreLink QoS-400 Network Interconnect Advanced Quality of Service are the same as for the CoreLink NIC-400 Network Interconnect. The QoS-400 is integrated into the NIC-400 architecture.

See the CoreLink NIC-400 Network Interconnect documentation for more information.

1.8 Product revisions

This section describes the differences in functionality between product revisions:

r0p0 First release.

r0p1 Second release. Wording changed to reflect period, and address request period.

r0p2 Third release. No technical updates.

Chapter 2
Functional Description

This chapter describes the major interfaces and components of the CoreLink QoS-400 Network Interconnect Advanced Quality of Service, and how it operates. It contains the following sections:

2.1 About the functions

Figure 2-1 shows the CoreLink QoS-400 Network Interconnect Advanced Quality of Service in position in the CoreLink NIC-400 Network Interconnect.

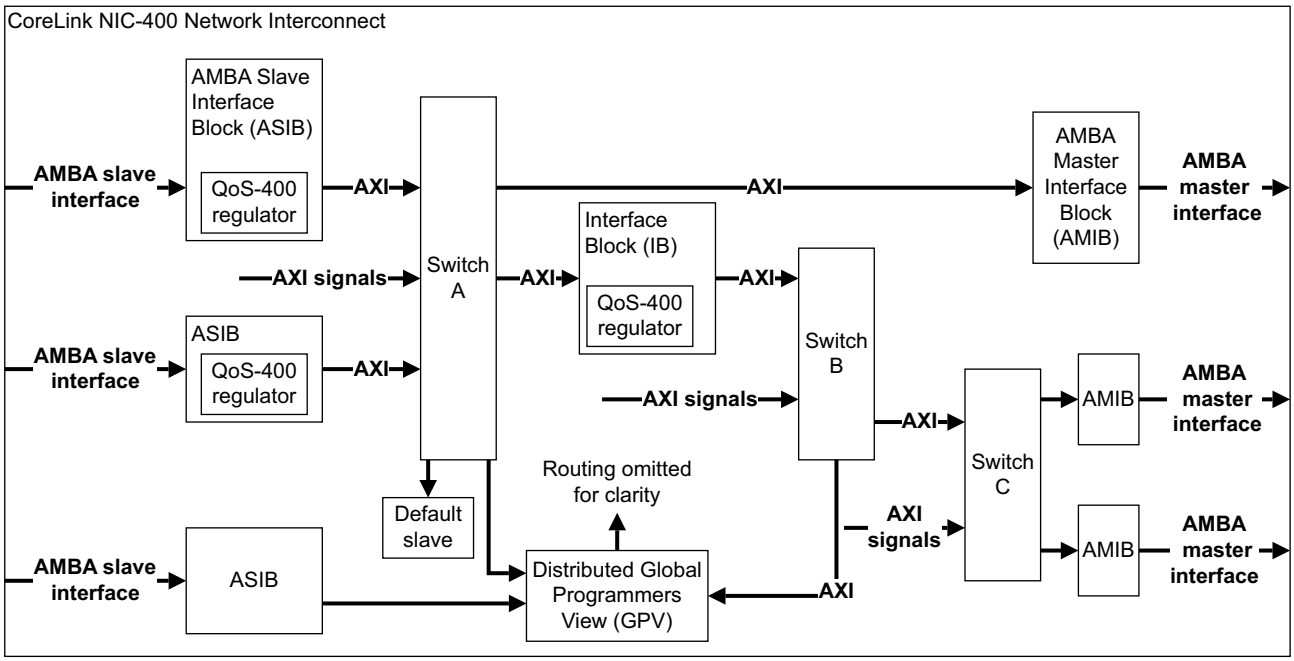

Figure 2-1 CoreLink NIC-400 Network Interconnect with CoreLink QoS-400 Advanced Quality of Service top-level diagram

Example 2-1 on page 2-3 contains an animation that shows the following masters, on the left-hand side of a NIC-400 interconnect:

- DMA controller.
- Graphics processor.
- ARM processor.
- LCD controller.

These masters send random requests to the *Dynamic Memory Controller* (DMC) slave on the right-hand side of the interconnect through the switches inside the interconnect. The DMC must process the requests and send responses back to the masters to service them. When a master sends a request, the number of outstanding transactions in the counter on the right-hand side of the animation increases by one. When the DMC slave services one of these transactions by sending a response, the number of outstanding transactions on the DMC decreases by one. In a system that does not use QoS-400, the number of outstanding transactions can increase up to the limit of the memory controller, at that point, the last master to send a request is not serviced. In Example 2-1 on page 2-3, the memory controller can handle a maximum of 12 transactions.

The animation in Example 2-1 on page 2-3 shows a situation in which the DMC slave cannot process the requests quickly enough, and the number of outstanding transactions reaches the critical limit of the maximum number of outstanding transactions.

Note

In the following two animation examples, you might have to save this document to your desktop before answering a security question to trust the source of this document.

Example 2-1 Animation showing transactions and responses in a NIC-400 interconnect that does not use QoS-400

To run the animation for the first time after opening the PDF file, left-click on the diagram. Acrobat gives you the following choices for how to run the animation:

Play the multimedia content this one time

If you select this option, Acrobat plays the animation once, but the next time you open the PDF file, Acrobat prompts you with the same question and does not consider the document to be trusted. Acrobat prompts you with the same question each time you open the PDF file from new.

Play the multimedia content and add this document to my list of trusted documents

If you select this option, Acrobat plays the animation once, and the next time you open the PDF file, it remembers that this is a trusted document, and does not prompt you with the same question again.

To run the animation again after the first time, right-click on the following animation and select **Play**. If you right-click and select **Loop**, the next time you run the animation, Acrobat repeats the animation continuously from start to end.

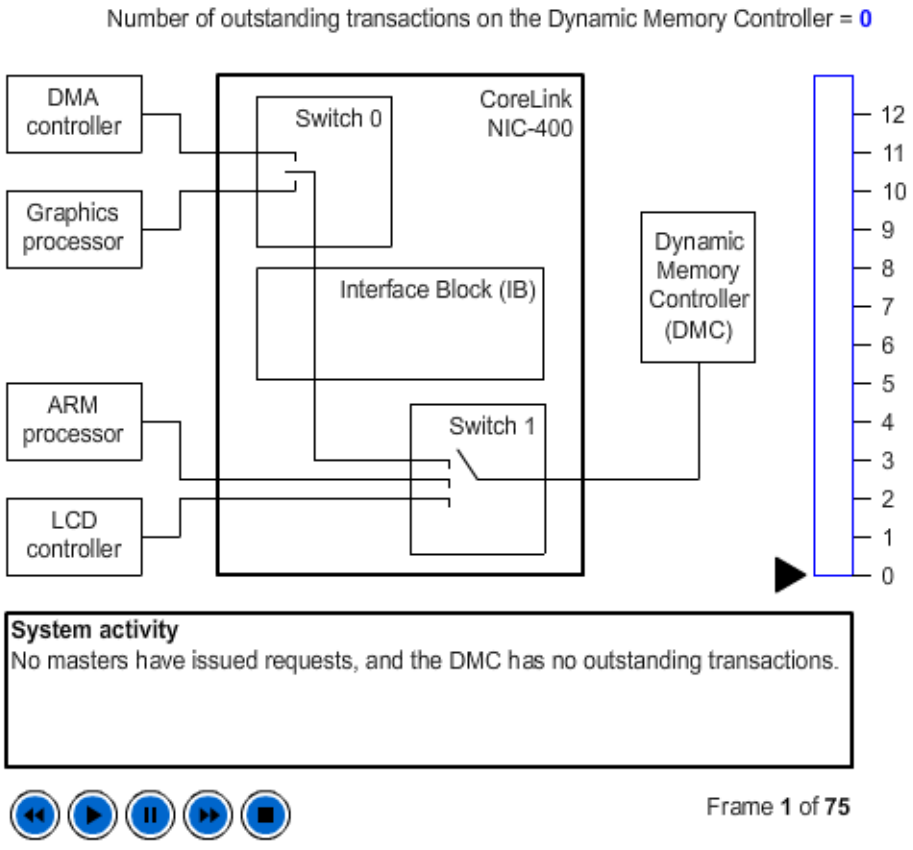

Diagram showing transactions and responses in a NIC-400 interconnect that does not use QoS-400

Example 2-2 Animation showing transactions and responses in a NIC-400 interconnect that uses QoS-400

The masters send random requests to the DMC slave on the right-hand side of the interconnect through the QoS-400 regulators and switches. The QoS-400 regulators for the DMA controller and graphics processor each have a maximum number of outstanding transactions that you specify, in this example, four and five respectively. In the same way as in Example 2-1 on page 2-3, the DMC must service the requests by sending responses back to the masters. With QoS-400 implemented, the DMC services the requests it receives for each master, until the QoS regulator for that master reaches its limit, the maximum number of outstanding transactions. This means that real-time-critical masters such as the LCD controller never fail to have their requests serviced, and no visible disruption occurs. By not adding a QoS-400 regulator for the LCD controller, as is the case in Example 2-2, whenever the LCD controller issues a request, the DMC services it immediately. The ARM processor also has no regulator, so its requests gain priority over the DMA controller and the graphics processor.

The animation in Example 2-2 shows the situation where the DMC responds to requests for the LCD controller and ARM processor when the other channels, for the DMA controller and graphics processor, are blocked because they have reached their individual limits for the maximum number of outstanding transactions.

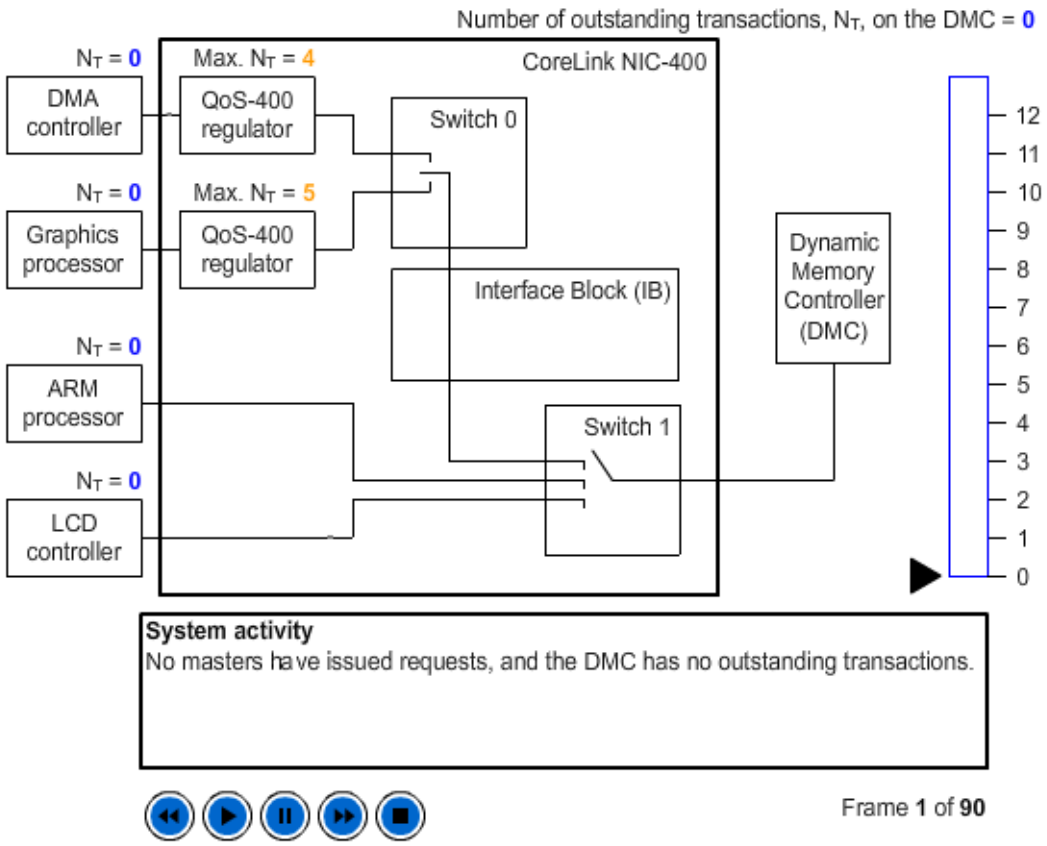

Diagram showing transactions and responses in a NIC-400 interconnect that uses QoS-40

QoS configuration options are available on:

- ASIBs that connect masters to the interconnect.
- IBs that connect between switches.

Note

You program the QoS-400 blocks from the NIC-400 GPV.

2.2 Operation

This section contains the following subsections:

2.2.1 Relationship with the CoreLink NIC-400 Network Interconnect

The CoreLink NIC-400 Network Interconnect base product includes an optional QoS value for each address, **awqos** for writes and **arqos** for reads. For the base product, you can configure the QoS value to one of the following options:

- Set to a fixed value at RTL configuration time.
- Set to a default value that you can change by programing at run-time.
- Input from an external master that has QoS signals.

Within the interconnect, the QoS values control arbitration. Externally, the QoS values connect to slaves, such as dynamic memory controllers, to arbitrate and prioritize traffic. The QoS-400 supplies additional hardware that can both regulate the read and write requests and control the QoS value dynamically.

You can also program QoS-400 to append a QoS value to every address request that acts as an arbitration priority value within the NIC-400 interconnect. You can forward the QoS value to the addressed slave so that it prioritizes the request and reduces its latency. You can also configure the slave interface to pass on a QoS value that the attached master supplies. This is part of the NIC-400 configuration and QoS-400 also uses it.

TrustZone technology and security

You implement the CoreLink QoS-400 Network Interconnect Advanced Quality of Service completely within the CoreLink NIC-400 Network Interconnect. It only extends the Programmers View of the NIC-400 within the 4KB blocks already allocated to the *Interface Block* (IB) and *AXI Slave Interface Block* (ASIB) nodes. It does not add any signals to the master or slave interfaces. The CoreLink QoS-400 Network Interconnect Advanced Quality of Service does not change the TrustZone® properties of the CoreLink Network Interconnect (NIC-400).

2.2.2 QoS regulators

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service provides facilities to regulate transactions based on the following inter-related measures:

R_T Issuing rate.

L_T Latency.

N_T Number of outstanding transactions.

Assuming that the master is always trying to issue transaction requests, then R_T, L_T, and N_T are related by the following formula:

$R_T = N_T \times 1/L_T$

The formula is a variation on Little's Law, that relates queue length to arrival rate and time in the system. A three-dimensional surface plotted in a graph represents the relationship between the transaction issuing rate $\mathbf{R}_T$, latency $\mathbf{L}_T$, and the number of outstanding transactions $\mathbf{N}_T$. See Figure 2-2.

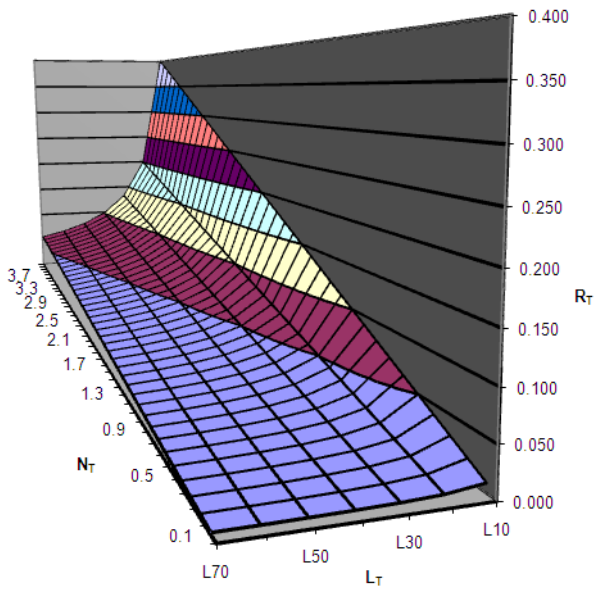

Figure 2-2 Relationship between transaction rate, latency, and outstanding transactions

QoS-400 supports three regulation mechanisms based on the $\mathbf{R}_T$, $\mathbf{L}_T$, and $\mathbf{R}_T$, measures. All three regulate the aggregate of the traffic through the ASIB or IB regardless of the eventual destination. This works best when the majority of the traffic is to a single shared resource, such as a DMC.

Note

A shared resource, such as a DMC, is often the most heavily loaded resource in a system.

The transaction rate regulators and the number of outstanding transactions set an upper bound on these measures. They prevent the measures from exceeding a limit that you program. They do this by holding back transaction requests whenever the limits are reached.

The transaction latency regulator adds a QoS value to the request, or modifies an existing one, and the interconnect uses that value as an arbitration priority. You can also configure the interconnect to forward this QoS value to a QoS value-sensitive slave. This slave can then prioritize the request based on its QoS value. The latency regulator increases the QoS value when it observes an increase in latency. QoS-400 introduces address request latency as an alternative to the transaction latency in QoS-301.

Transaction latency specifies a target bandwidth in combination with a number of outstanding transactions using the formula from Little's Law. This might be easier to use where the number of outstanding transactions is limited.

Address request latency regulation directly sets a target period for address requests, that is equivalent to setting a target maximum bandwidth. The number of outstanding transactions can vary according to the transaction latency in accordance with Little's Law. Consequently, as the transaction latency increases, the number of outstanding transactions also increases to maintain the bandwidth.

The CoreLink QoS-400 Network Interconnect Advanced Quality of Service regulators provide mechanisms that control how the NIC-400 shares the resources of a slave. You can use them either individually, or in combination, to control this sharing, and to prevent overloading the slave.

One measure of resource loading is the number of requests in the queue waiting to be served. When most of the requests from a number of masters all go to the same slave, the sum of the number of outstanding transactions at each master corresponds to the number of requests waiting to be served at that slave.

Regulating the number of outstanding transactions from each of the masters therefore provides a direct means to distribute the resources of a slave without overloading it. If the loading changes for any reason, for example when a master completes its task early, then:

- The load on the slave is reduced.
- The queue length decreases.
- The average latency that the remaining masters observe decreases.

Figure 2-2 on page 2-7 shows that for a constant number of outstanding transactions, a decrease in latency corresponds to an increase in transaction issuing rate. In effect, some of the spare capacity has been distributed to the other masters.

Another measure of the resource that a slave provides is the rate at which it processes transactions. The transaction rate regulator enables you to limit the rate at which a master issues transactions, and therefore sets the proportion of the resource of the slave that is requested. Figure 2-2 on page 2-7 shows that for a constant issuing rate, a master can compensate for any increase in latency by increasing the number of outstanding transactions.

The third measure of latency is not under the direct control of a master because it depends on many factors in the interconnect and the slave. The more heavily loaded a slave is, the longer a transaction must wait for service, and the higher the latency. You can regulate the latency by adding a QoS value to transaction requests to indicate to the system and slave when a master requires the latency to be reduced. The higher the value, the lower the latency required.

All three regulators can be active at the same time. For example, if the rate regulator is limiting a master to a particular rate, and the latency increases, then the number of outstanding transactions tends to increase to compensate. If you permit this to continue unchecked, then the master takes more than its share of the resource from the slave. You can program the outstanding transaction regulator to prevent this. At the same time, the latency regulator increases the priority to decrease the latency, enabling the rate to recover, and the number of outstanding transactions to decrease.

A control register permits you to enable or disable any combination of the three regulators in one programming step. See Chapter 3 *Programmers Model*.

You can also configure internal IBs to contain QoS regulators that provide intermediate regulation points between switches in an interconnect. This could, for example, enable a composite flow of requests from many sources to be constrained without having to over-constrain the individual flows.

2.2.3 Transaction rate regulation

A variant of the standard internet *Traffic SPECification* (TSPEC) specifies transaction rate regulation using the following parameters:

p Peak rate.

b Burstiness allowance.

r Average rate.

You can independently program and enable the regulation of the read and write address channels with their own control bits. Alternatively, you can select combined regulation of the read and write address channels using another control bit. See Chapter 3 *Programmers Model*.

The request arrival curve, that graph (A) TSPEC traffic upper bound shows, in Figure 2-3 represents the characteristics imposed on the request flow, or flows.

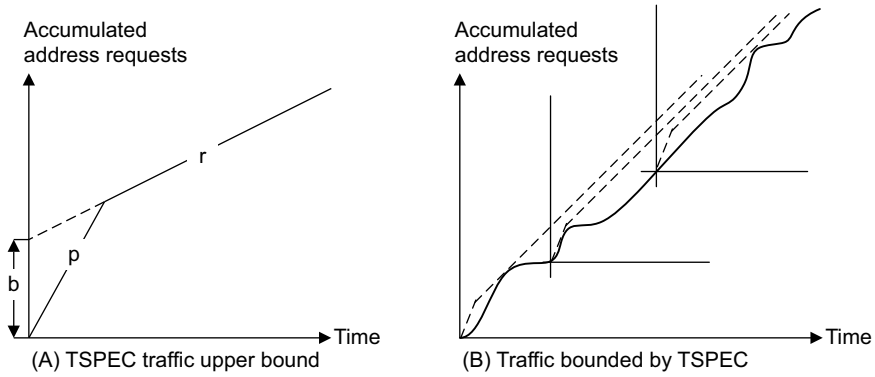

Figure 2-3 TSPEC traffic specification

The TSPEC parameters define an upper bound that applies over any time window.

(B) Traffic bounded by TSPEC in Figure 2-3 illustrates this, with the accumulated data curve bounded by the TSPEC curve from any point in the data sequence. You program the TSPEC parameter values for AW and AR request rates in separate sets of registers. See Chapter 3 *Programmers Model*.

The regulators are disabled after reset.

If you program the regulators while they are enabled, the new values take effect immediately. Alternatively, you can disable the regulators, update the values, and then re-enable the regulators.

Binary fractions in transfers per cycle provide the values for peak and average rates.

So, for example, a value of `0x800`, that is, 0.5 in decimal, sets a rate of one transfer every two cycles.

A value of `0x100`, that is, 0.0625 in decimal, sets a rate of one transfer every 16 cycles.

A value of `0x000` sets a rate of one transfer per cycle, that is, no regulation. If you set either the burstiness to 0, or the average rate to 0, this disables the regulation of burstiness and average rate, (b,r).

In the same way, if you set the peak rate to 0, this disables the peak rate regulation. You can set:

- Peak rate only.
- Burstiness and average rate, without the peak rate.

Example 2-3 How to program the TSPEC regulator registers

This example describes a case where you set the following requirements:

- Transaction rate to use 4% of the available bandwidth.
- Each transaction to be a 16-beat burst.
- Peak rate, p.
- Burstiness allowance, b.

- Average rate registers, r.

It must be noted that the burstline allowance is not related to the AXI burst length. It is a measure of traffic burstiness, in this case the burstiness of transfers on the AW or AR channels.

Transactions are regulated on AW and AR transfers, so you can use the burst length to calculate the AR or AW transfer rate from the required bandwidth as follows:

- The required data bandwidth is 4% that is equal to four data transfers in every 100 cycles.
- One transaction has 16 beats of data, so requires one transaction every 16/4 * 100 cycles. This gives one transaction every 400 cycles.

One transfer every 400 clock cycles is equivalent to 0.0025 transfers per cycle. You must convert this into a 12 bit binary fraction to give the value to program the average rate register with:

`0b000000001010`

This value is an approximation limited in accuracy by the number of bits available in the average rate register.

This value is an approximation limited in accuracy by the number of bits available in the average rate register. See *AW Channel Average Rate Register* on page 3-8 and *AR Channel Average Rate Register* on page 3-10. In this example, the nearest approximation gives a rate of 1 transaction every 409.6 clock cycles or 16/409.6*100 = 3.9% of available bandwidth. If you do not set all three TSPEC values, you can set either the peak rate only, or the burstiness and average rate, without the peak rate.

The peak rate register, p is only 8 bits, not 12 bits as in the average rate register, r, so the previously calculated value does not fit. See *AW Channel Average Rate Register* on page 3-8 and *AR Channel Average Rate Register* on page 3-10. Therefore, you can achieve hard regulation at one transaction in 409 clock cycles by setting the following:

p `0b00000000`. See *AW Channel Peak Rate Register* on page 3-7 and *AR Channel Peak Rate Register* on page 3-9.

r `0b000000001010`. See *AW Channel Average Rate Register* on page 3-8 and *AR Channel Average Rate Register* on page 3-10.

b `0b0000000000000001`. See *AW Channel Burstiness Allowance Register* on page 3-8 and *AR Channel Burstiness Allowance Register* on page 3-9.

The burstiness allowance, combined with the peak rate and average rate, enables variance in the issuing rate from that master during different system loadings. For example, set the values as follows:

p `0b00000001`, that is, 1 in 256. See *AW Channel Peak Rate Register* on page 3-7 and *AR Channel Peak Rate Register* on page 3-9.

r `0b000000001010`, that is, 1 in 409.6. See *AW Channel Average Rate Register* on page 3-8 and *AR Channel Average Rate Register* on page 3-10.

b `0b0000000000000101`. See *AW Channel Burstiness Allowance Register* on page 3-8 and *AR Channel Burstiness Allowance Register* on page 3-9.

This permits a maximum issuing rate of one request every 256 clock cycles until the burstiness allowance, b, has been used up, and then, a maximum issuing rate, r, of one request every 409.6 clock cycles.

The regulator keeps track of the burstiness allowance to control whether the issuing rate is limited to the peak rate, p, or to the average rate, r. The burstiness allowance is initialized from the burstiness allowance register, b. When regulation is enabled it is decremented on every transfer and incremented at the average rate. While the allowance is non-zero the peak rate is allowed. In this example, the burstiness allowance starts at 5, so, at the peak issuing rate, the allowance reaches 0 after:

b * (p/(p-r)) = 5 * (O.0039/(0.0039 - 0.00244) = 13 transfers

When the allowance has reached zero the issuing rate is limited to the average rate. If the issuing rate from the master drops below the average rate, then the burstiness allowance increases until it reaches the programmed burstiness allowance, b.

Note

To calculate the 12-bit binary from the decimal fraction use the following equation:

<12-bit binary fraction> = 12^12 * <decimal fraction> = 4096 * <decimal fraction>

For example, converting the decimal fraction 0.0025, would give 4096 * 0.0025 or 10.24. Rounding this to the nearest whole number gives 10 or `0b0000 0000 1010` as a 12-bit binary number.

Combined AW and AR transaction rate regulation

You can regulate the combined transaction rate from the AW and AR channels. When you select this mode, QoS ignores the individual channel rates. QoS takes the TSPEC parameter values for the AW and AR channels from the values programmed in the AW registers. See Chapter 3 *Programmers Model*.

Because two channels support twice the rate of a single channel, QoS scales the TSPEC parameters, for combined regulation, by a factor of two. For example, to specify a combined average rate of one transfer every eight cycles, set the value to `0x100`. This is equal to two transfers every 16 cycles. That is, the rate you program is half the required combined rate.

When the combined AW and AR channel traffic is so close to the TSPEC boundary that only one transfer is permitted, but both channels are requesting, then the regulator admits the AW channel and AR channel alternately.

2.2.4 Outstanding transaction regulation

The regulator enables you to program, at run-time, values for the number of AW and AR requests that it issues. See Chapter 3 *Programmers Model*.

At design time, you can configure the ASIB, or internal IBs, using maximum values for the number of AW and AR requests that the master can issue at any one time. You normally set these limits to match the characteristics of the attached master, in the case of an ASIB, but in either case, you must always observe these limits because they size the downstream components. If you program the QoS regulator with larger values than the configuration limits, it has no effect.

Fractional outstanding transactions

You can characterize a sequence of transactions, with periods when there are no outstanding transactions, by using a fractional outstanding transaction number. For example, if requests occur every 100 cycles, but it only takes 50 cycles for the last response to arrive, then this would correspond to an average of 0.5 outstanding transactions. This generalizes to give an average value for any sequence of transactions where the number of outstanding transactions varies.

Combined AW and AR outstanding transaction regulation

You can program a maximum outstanding transaction capability for the combined AW and AR channels. This is in addition to the individual channel maxima, so that it is possible to set a combined limit that is lower than the sum of the individual channel limits. A combined value of zero, or greater than or equal to the sum of the individual configuration limits, has no effect.

For example, if the AW and AR configuration issuing capabilities are four and four, the default issuing capability is eight. Setting a combined value of six lowers the combined issuing capability, but leaves the individual channel configuration limits unaltered. Therefore, if there are four outstanding AW requests, then only two AR requests are permitted. Alternatively, if there are only two outstanding AW requests, then four AR requests can be made.

2.2.5 Transaction latency regulation

The feedback control regulator achieves transaction latency regulation by modifying the **axqos** value of each transaction request. This overrides any **axqos** value that the NIC-400 base product might have specified. If the interconnect and the addressed slave treat this as a priority value, then it has the required regulatory effect. For example, if a transaction is given an **axqos** value that gives it a higher priority, then this tends to reduce the transaction latency of that transaction. In this way, a feedback loop is set up so that when the actual latency is higher than the target transaction latency, the **axqos** value is proportionately raised, and the larger the transaction latency discrepancy, the higher the priority.

Note

- The regulator measures the time it takes for a transaction to complete.
- **axqos** is either **arqos** or **awqos**.

Transaction latency regulation is useful for masters that have performance that is directly dependent on transaction latency. For example, a processor might be stalled while it waits for data after a cache miss.

You program the target transaction latency separately for writes and reads. You enable transaction latency regulation by setting the appropriate control bits in the QoS control register. See *QoS Control Register* on page 3-4.

When you enable transaction latency regulation for reads or writes, the base product **axqos** values are not used.

You set the range of **axqos** values used for transaction latency regulation by programming the minimum and maximum values for writes and reads.

You program a scaling factor to give control over how quickly the **axqos** values change. The smaller the scaling factor, the more slowly the **axqos** values change in response to changes in transaction latency. The scaling factor is specified in powers of two. See *Feedback Controlled Scale Register* on page 3-11.

2.2.6 Address request latency regulation

The feedback control regulator achieves address request latency regulation by modifying the **axqos** value of each transaction request. This overrides any **axqos** value that the base product has specified. If the interconnect and the addressed slave treat this as a priority value, then it has the required regulatory effect. For example, if a transaction is given an **axqos** value that gives it a higher priority, then this reduces the latency between address requests. In this way, a feedback loop is set up so that when the actual latency is higher than the target latency, the **axqos** value is proportionately raised, and the larger the latency discrepancy, the higher the priority.

Note

- The regulator measures the time between address requests.
- **axqos** is either **arqos** or **awqos**.

Address request period regulation is useful for masters that have minimum bandwidth requirements, for example, a GPU.

You program the target period separately for writes and reads. You enable address request period regulation by setting the appropriate control bits in the QoS control register. See *QoS Control Register* on page 3-4.

When you enable address request period regulation for reads or writes, the base product **axqos** values are not used. When you disable address request period regulation, the base product **axqos** values are passed through unmodified.

You set the range of **axqos** values used for address request period regulation by programming the minimum and maximum values for writes and reads.

You program a scaling factor to give control over how quickly the **axqos** values change. The smaller the scaling factor, the more slowly the **axqos** values change in response to changes in latency. The scaling factor is specified in powers of two. See *Feedback Controlled Scale Register* on page 3-11.

Chapter 3
Programmers Model

This chapter describes the programmers model. It contains the following sections:

3.1 About this programmers model

The following information applies to the QoS-400 registers:

- The base address is not fixed, and can be different for any particular system implementation. The offset of each register from the base address is fixed.
- Unless otherwise stated in the accompanying text:
  - Do not modify undefined register bits.
  - Ignore undefined register bits on reads.
  - All register bits are reset to a logic 0 by a system reset, or a powerup reset.
- Access type in Table 3-1 on page 3-3 is described as follows:

  RW Read and write.

  RO Read only.

  WO Write only.

3.2 Register summary

Table 3-1 shows the registers in offset order from the base memory address.

Table 3-1 Register summary

Offset	Name	Type	Reset	Width	Description
0x000-0x0FF	-	-	-	-	Reserved.
0x100	read_qos	RW	0	4	CoreLink NIC-400 Network Interconnect read_qos register. See the *ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual* for information about this register.
0x104	write_qos	RW	0	4	CoreLink NIC-400 Network Interconnect write_qos register. See the *ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual* for information about this register.
0x108	fn_mod	RW	0	2	CoreLink NIC-400 Network Interconnect fn_mod register. See the *ARM® CoreLink™ NIC-400 Network Interconnect Technical Reference Manual* for information about this register.
0x10C	qos_cntl	RW	0	2,8	*QoS Control Register* on page 3-4.
0x110	max_ot	RW	0	6, 8, 6, 8	*Maximum Number of Outstanding Transactions Register* on page 3-5.
0x114	max_comb_ot	RW	0	7, 8	*Maximum Combined Outstanding Transactions Register* on page 3-6.
0x118	aw_p	RW	0	8	*AW Channel Peak Rate Register* on page 3-7.
0x11C	aw_b	RW	0	16	*AW Channel Burstiness Allowance Register* on page 3-8.
0x120	aw_r	RW	0	12	*AW Channel Average Rate Register* on page 3-8.
0x124	ar_p	RW	0	8	*AR Channel Peak Rate Register* on page 3-9.
0x128	ar_b	RW	0	16	*AR Channel Burstiness Allowance Register* on page 3-9.
0x12C	ar_r	RW	0	12	*AR Channel Average Rate Register* on page 3-10.
0x130	target_fc	RW	0	12, 12	*Feedback Controlled Target Register* on page 3-11.
0x134	ki_fc	RW	0	3, 3	*Feedback Controlled Scale Register* on page 3-11.
0x138	qos_range	RW	0	4, 4, 4, 4	*QoS Range Register* on page 3-12.
0x13C-0xFFF	-	-	-	-	Reserved.

3.3 Register descriptions

This section describes the CoreLink QoS-400 Network Interconnect Advanced Quality of Service registers. Table 3-1 on page 3-3 provides cross-references to individual registers.

3.3.1 QoS Control Register

The qos_cntl Register characteristics are:

Purpose This register contains the enable bits for all the regulators. By default, all of the bits are set to 0, and no regulation is enabled.

Usage constraints Regulation only takes place when both the enable bit is set, and its corresponding regulation value is non-zero. This enables you to perform an integration test without activating the regulation.

The QoS regulators are reset whenever they are re-enabled. The transaction and address latency regulator enables **en_ar_fc** and **en_aw_fc** can be used to provide a soft reset of these regulators. If the values in the qos_range register are changed to make the range narrower than either the previous upper or lower bounds then you must reset the regulator through a soft reset by disabling and re-enabling it. If the values in the **qos_range** register are changed to make the range wider than either the previous upper or lower bounds then the regulator does not require a soft reset.

Configurations Available in all QoS-400 configurations.

Attributes See Table 3-1 on page 3-3.

Figure 3-1 shows the bit assignments.

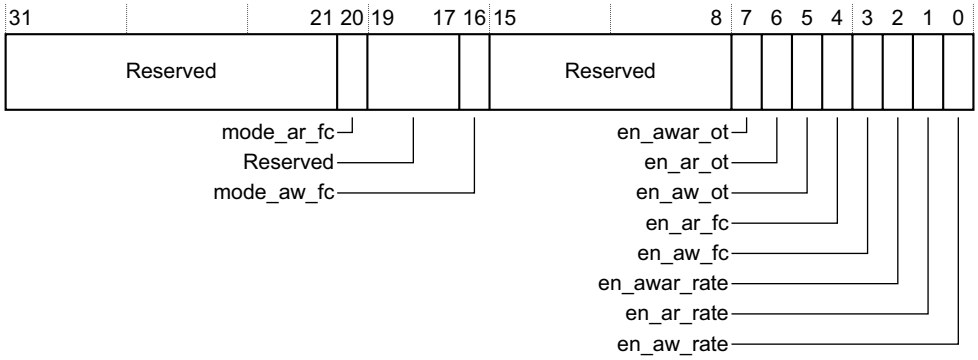

Figure 3-1 qos_cntl Register bit assignments

Table 3-2 shows the bit assignments.

Table 3-2 qos_cntl Register bit assignments

Bits	Name	Function
[31:21]	-	Reserved. Do not modify. Read as zero.
[20]	mode_ar_fc	Select feedback control regulation for AR of either transaction or address latency as follows: `0b0` Transaction latency. `0b1` Address latency.
[19:17]	-	Reserved. Do not modify. Read as zero.

Table 3-2 qos_cntl Register bit assignments (continued)

Bits	Name	Function
[16]	mode_aw_fc	Select feedback control regulation for AW of either transaction or address latency as follows: `0b0` Transaction latency. `0b1` Address latency.
[15:8]	-	Reserved. Do not modify. Read as zero.
[7]	en_awar_ot[a]	Enable combined regulation of outstanding transactions.
[6]	en_ar_ot[a]	Enable regulation of outstanding read transactions.
[5]	en_aw_ot[a]	Enable regulation of outstanding write transactions.
[4]	en_ar_fc[b]	Enable regulation of AR transaction or address latency using feedback control, depending on the value to which you set mode_ar_fc.
[3]	en_aw_fc[b]	Enable regulation of AW transaction or address latency using feedback control, depending on the value to which you set mode_aw_fc.
[2]	en_awar_rate[c]	Enable combined AW and AR rate regulation.
[1]	en_ar_rate[c]	Enable AR rate regulation.
[0]	en_aw_rate[c]	Enable AW rate regulation.

a. If you include outstanding transaction regulation, you can configure en_awar_ot, en_ar_ot, and en_aw_ot. Otherwise, bits [7:5] are reserved, read as zero, and you cannot modify them.

b. If you include transaction or address latency regulation, you can configure en_ar_fc and en_aw_fc. Otherwise, bits [4:3] are reserved, read as zero, and you cannot modify them.

c. If you include transaction rate regulation, you can configure en_awar_rate, en_ar_rate, and en_aw_rate. Otherwise, bits [2:0] are reserved, read as zero, and you cannot modify them.

3.3.2 Maximum Number of Outstanding Transactions Register

The max_ot Register characteristics are:

Purpose This register enables you to program the maximum number of address requests for the AR and AW channels. See *Outstanding transaction regulation* on page 2-11 for more information.

The outstanding transaction limits have an integer part and a fractional part as follows:

ar_max_oti
Corresponds to the integer part of the maximum outstanding AR addresses.

aw_max_oti
Corresponds to the integer part of the maximum outstanding AW addresses.

ar_max_otf
Corresponds to the fractional part of the maximum outstanding AR addresses.

aw_max_otf
Corresponds to the fractional part of the maximum outstanding AW addresses.

A value of 0 for both the integer and fractional parts disables the programmable regulation so that the NIC-400 base product configuration limits apply.

The AW and AR outstanding transaction limits are enabled when you set the corresponding en_aw_ot or en_ar_ot control bits of the QoS control register. See *QoS Control Register* on page 3-4 and Table 3-2 on page 3-4.

Usage constraints You cannot increase the following limits:

- The configuration limits you set at design time.
- The CoreLink NIC-400 Network Interconnect limit of 32.

Configurations Only available when you select outstanding transaction regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-2 shows the bit assignments.

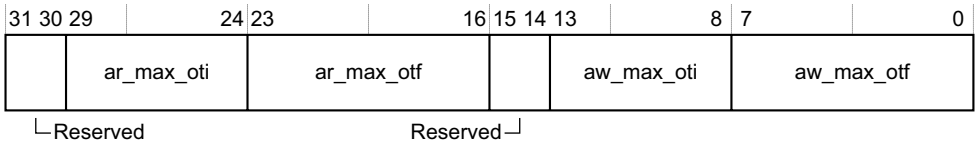

Figure 3-2 max_ot Register bit assignments

Table 3-3 shows the bit assignments.

Table 3-3 max_ot Register bit assignments

Bits	Name	Function
[31:30]	-	Reserved. Do not modify. Read as zero.
[29:24]	ar_max_oti	Integer part of the maximum outstanding AR addresses.
[23:16]	ar_max_otf	Fractional part of the maximum outstanding AR addresses.
[15:14]	-	Reserved. Do not modify. Read as zero.
[13:8]	aw_max_oti	Integer part of the maximum outstanding AW addresses.
[7:0]	aw_max_otf	Fractional part of the maximum outstanding AW addresses.

3.3.3 Maximum Combined Outstanding Transactions Register

The max_comb_ot Register characteristics are:

Purpose This register enables you to program the maximum number of address requests for the AR and AW channels. The combined limit is applied after any individual channel limits. See *Outstanding transaction regulation* on page 2-11 for more information.

The outstanding transaction limits have an integer part and a fractional part as follows:

awar_max_oti

Corresponds to the integer part.

awar_max_otf

Corresponds to the binary fraction.

A value of 0 for both the integer and fractional parts disables the programmable regulation so that the configuration limits apply.

The regulation of the combined outstanding transaction limit also requires that you set the en_awar_ot control bit of the QoS control register. See *QoS Control Register* on page 3-4 and Table 3-2 on page 3-4.

Usage constraints You cannot increase the following limits:

- The configuration limits you set at design time.
- The CoreLink NIC-400 Network Interconnect limit of 64.

Configurations Only available when you select outstanding transaction regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-3 shows the bit assignments.

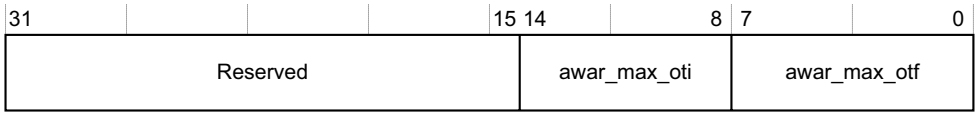

Figure 3-3 max_comb_ot Register bit assignments

Table 3-4 shows the bit assignments.

Table 3-4 max_comb_ot Register bit assignments

Bits	Name	Function
[31:15]	-	Reserved. Do not modify. Read as zero.
[14:8]	awar_max_oti	Integer part of the maximum combined outstanding AW and AR addresses.
[7:0]	awar_max_otf	Fractional part of the maximum combined outstanding AW and AR addresses.

3.3.4 AW Channel Peak Rate Register

The aw_p Register characteristics are:

Purpose This register enables you to program a binary fraction of the peak number of transfers per cycle. See *Transaction rate regulation* on page 2-8 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select transaction rate regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-4 shows the bit assignments.

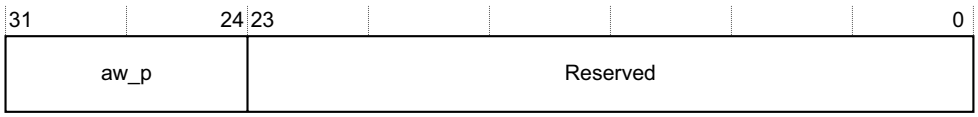

Figure 3-4 aw_p Register bit assignments

Table 3-5 shows the bit assignments.

Table 3-5 aw_p Register bit assignments

Bits	Name	Function
[31:24]	aw_p	AW channel peak rate.
[23:0]	-	Reserved. Do not modify. Read as zero.

3.3.5 AW Channel Burstiness Allowance Register

The aw_b Register characteristics are:

Purpose This register enables you to program a burstiness allowance, in number of transfers. See *Transaction rate regulation* on page 2-8 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select transaction rate regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-5 shows the bit assignments.

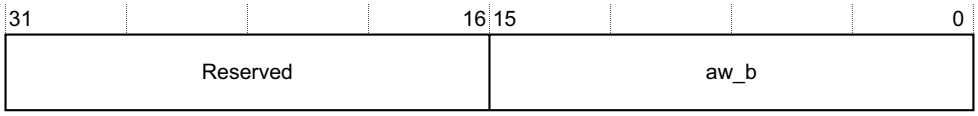

Figure 3-5 aw_b Register bit assignments

Table 3-6 shows the bit assignments.

Table 3-6 aw_b Register bit assignments

Bits	Name	Function
[31:16]	-	Reserved. Do not modify. Read as zero.
[15:0]	aw_b	AW channel burstiness.

3.3.6 AW Channel Average Rate Register

The aw_r Register characteristics are:

Purpose This register enables you to program a binary fraction of the average number of transfers per cycle. See *Transaction rate regulation* on page 2-8 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select transaction rate regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-6 on page 3-9 shows the bit assignments.

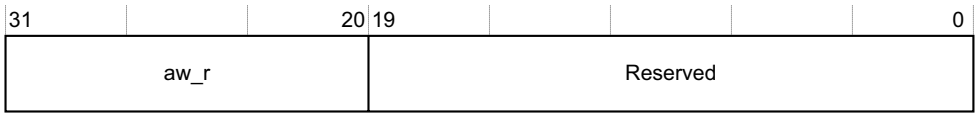

Figure 3-6 aw_r Register bit assignments

Table 3-7 shows the bit assignments.

Table 3-7 aw_r Register bit assignments

Bits	Name	Function
[31:20]	aw_r	AW channel average rate.
[19:0]	-	Reserved. Do not modify. Read as zero.

3.3.7 AR Channel Peak Rate Register

The ar_p Register characteristics are:

Purpose This register enables you to program a binary fraction of the peak number of transfers per cycle. See *Transaction rate regulation* on page 2-8 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select transaction rate regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-7 shows the bit assignments.

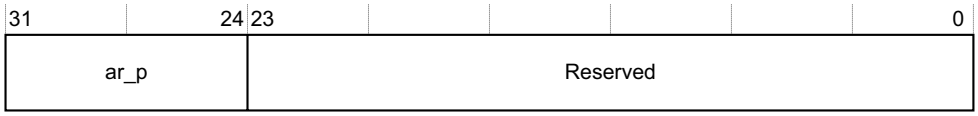

Figure 3-7 ar_p Register bit assignments

Table 3-8 shows the bit assignments.

Table 3-8 ar_p Register bit assignments

Bits	Name	Function
[31:24]	ar_p	AR channel average rate.
[23:0]	-	Reserved. Do not modify. Read as zero.

3.3.8 AR Channel Burstiness Allowance Register

The ar_b Register characteristics are:

Purpose This register enables you to program a burstiness allowance, in number of transfers. See *Transaction rate regulation* on page 2-8 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select transaction rate regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-8 shows the bit assignments.

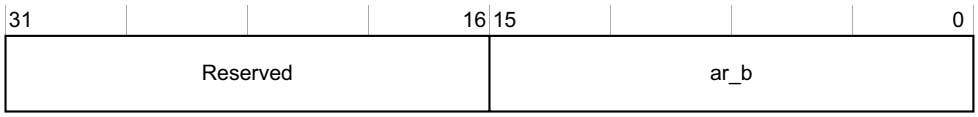

Figure 3-8 ar_b Register bit assignments

Table 3-9 shows the bit assignments.

Table 3-9 ar_b Register bit assignments

Bits	Name	Function
[31:16]	-	Reserved. Do not modify. Read as zero.
[15:0]	ar_b	AR channel burstiness.

3.3.9 AR Channel Average Rate Register

The ar_r Register characteristics are:

Purpose This register enables you to program a binary fraction of the average number of transfers per cycle. See *Transaction rate regulation* on page 2-8 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select transaction rate regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-9 shows the bit assignments.

Figure 3-9 ar_r Register bit assignments

Table 3-10 shows the bit assignments.

Table 3-10 ar_r Register bit assignments

Bits	Name	Function
[31:20]	ar_r	AR channel average rate.
[19:0]	-	Reserved. Do not modify. Read as zero.

3.3.10 Feedback Controlled Target Register

The target_fc Register characteristics are:

Purpose This register enables you to program a target latency, in cycles, for the regulation of reads and writes. A value of 0 corresponds to no regulation. See *Transaction latency regulation* on page 2-12 and *Address request latency regulation* on page 2-12 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select latency regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-10 shows the bit assignments.

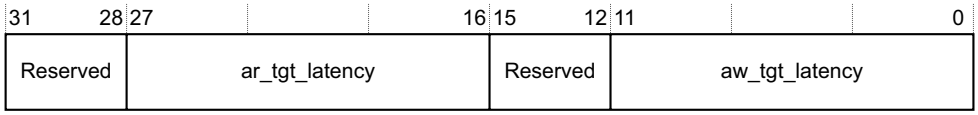

Figure 3-10 target_fc Register bit assignments

Table 3-11 shows the bit assignments.

Table 3-11 target_fc Register bit assignments

Bits	Name	Function
[31:28]	-	Reserved. Do not modify. Read as zero.
[27:16]	ar_tgt_latency	AR channel target latency.
[15:12]	-	Reserved. Do not modify. Read as zero.
[11:0]	aw_tgt_latency	AW channel target latency.

3.3.11 Feedback Controlled Scale Register

The ki_fc Register characteristics are:

Purpose This register enables you to program a latency regulation value, **awqos** or **arqos**, scale factor coded for powers of 2 in the range 2^{-3} to 2^{-10}, to match a 16-bit integrator. See *Transaction latency regulation* on page 2-12 and *Address request latency regulation* on page 2-12 for more information.

Usage constraints There are no usage constraints.

Configurations Only available when you select latency regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-11 shows the bit assignments.

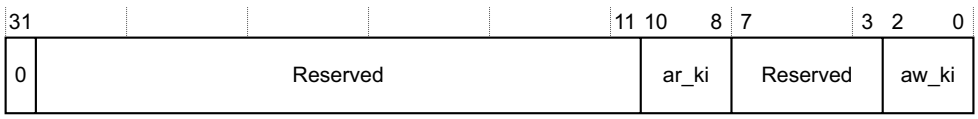

Figure 3-11 ki_fc Register bit assignments

Table 3-12 shows the bit assignments.

Table 3-12 ki Register bit assignments

Bits	Name	Function
[31:11]	-	Reserved. Do not modify. Read as zero.
[10:8]	ar_ki	**arqos** scale factor, power of 2 in the range 2^{-3} to 2^{-10}.
[7:3]	-	Reserved. Do not modify. Read as zero.
[2:0]	aw_ki	**awqos** scale factor, power of 2 in the range 2^{-3} to 2^{-10}.

Table 3-13 defines the translation from the programmed value to the scale factor used to derive the QoS value from the latency integrator. See *Transaction latency regulation* on page 2-12 and *Address request latency regulation* on page 2-12 for more information.

Table 3-13 Latency regulation scale factor translation

Latency regulation value	Latency regulation scale factor
0x0	2^{-3}
0x1	2^{-4}
0x2	2^{-5}
0x3	2^{-6}
0x4	2^{-7}
0x5	2^{-8}
0x6	2^{-9}
0x7	2^{-10}

3.3.12 QoS Range Register

The qos_range Register characteristics are:

Purpose This register enables you to program the minimum and maximum values for the **arqos** and **awqos** signals that the latency regulators generate. See *Transaction latency regulation* on page 2-12 and *Address request latency regulation* on page 2-12 for more information.

Usage constraints Do not set:

- `ar_min_qos` to a value greater than `ar_max_qos` because this causes unpredictable behavior.
- `aw_min_qos` to a value greater than `aw_max_qos` because this causes unpredictable behavior.

Configurations Only available when you select latency regulation in AMBA Designer.

Attributes See Table 3-1 on page 3-3.

Figure 3-12 on page 3-13 shows the bit assignments.

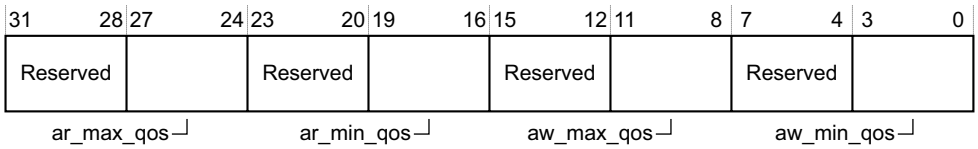

Figure 3-12 qos_range Register bit assignments

Table 3-14 shows the bit assignments.

Table 3-14 qos_range Register bit assignments

Bits	Name	Function
[31:28]	-	Reserved. Do not modify. Read as zero.
[27:24]	ar_max_qos	Maximum **arqos**.
[23:20]	-	Reserved. Do not modify. Read as zero.
[19:16]	ar_min_qos	Minimum **arqos**.
[15:12]	-	Reserved. Do not modify. Read as zero.
[11:8]	aw_max_qos	Maximum **awqos**.
[7:4]	-	Reserved. Do not modify. Read as zero.
[3:0]	aw_min_qos	Minimum **awqos**.

Appendix A
Signal Descriptions

This chapter describes the QoS-400 signals. It contains the following sections:

A.1 Introduction

You can configure the CoreLink QoS-400 Network Interconnect Advanced Quality of Service signals in the CoreLink NIC-400 Network Interconnect. The CoreLink QoS-400 Network Interconnect Advanced Quality of Service also uses these signals.

The following subsections describe the CoreLink QoS-400 Network Interconnect Advanced Quality of Service signals:

- *AXI Slave Interface Block (ASIB) signals* on page A-3.
- *AXI Master Interface Block (AMIB) signals* on page A-4.

A.2 AXI Slave Interface Block (ASIB) signals

You can configure the ASIB to have the QoS signals that Table A-1 shows.

Table A-1 ASIB signals

Name	Direction	Width	Description
awqos[a]	Input	[3:0]	The QoS value for this write transaction. This acts as a priority and the higher the value, the higher the priority.
arqos[a]	Input	[3:0]	The QoS value for this read transaction. This acts as a priority and the higher the value, the higher the priority.

a. This signal is an optional signal in the NIC-400 base product. This signal is overridden if you use QoS-400, and you use latency regulation.

A.3 AXI Master Interface Block (AMIB) signals

You can configure the AMIB to have the QoS signals that Table A-2 shows. A connected slave can use these signals to arbitrate between transactions based on the priority value.

Table A-2 AMIB signals

Name	Direction	Width	Description
awqos	Output	[3:0]	The QoS value for this write transaction. This acts as a priority and the higher the value, the higher the priority.
arqos	Output	[3:0]	The QoS value for this read transaction. This acts as a priority and the higher the value, the higher the priority.

Appendix B
Revisions

This appendix describes the technical changes between released issues of this book.

Table B-1 Issue A

Change	Location	Affects
First release	-	-

Table B-2 Differences between issue A and issue B

Change	Location	Affects
Wording changed to reflect period and address request period rather than latency throughout section	*Address request latency regulation* on page 2-12	All revisions

Table B-3 Differences between Issue B and issue C

Change	Location	Affects
Updated the example on how to program the TSPEC regulator registers and added a note	Example 2-3 on page 2-9	All revisions