|  |
| --- |
| NetSpeed Debug and Trace Methodology  *Release Brief*  Version: 1.0  October 17, 2016 |

NetSpeed Debug and Trace Methodology

About This Document

This document describes NetSpeed product support for debugging and tracing system activity.

Audience

This document is intended for users of NocStudio:

* NoC Architects
* NoC Designers
* SoC Designers

Prerequisite

Before proceeding, you should generally understand:

* Basics of NetSpeed NoC IP

Related Documents

The following documents can be used as a reference to this document.

* NetSpeed NocStudio Orion AMBA User Manual
* NetSpeed Orion IP Integration Spec

Customer Support

For technical support about this product, please contact [support@netspeedsystems.com](mailto:support@netspeedsystems.com)

For general information about NetSpeed products refer to: [www.netspeedsystems.com](http://www.netspeedsystems.com)

**Contents**

[About This Document 2](#_Toc464477624)

[Audience 2](#_Toc464477625)

[Prerequisite 2](#_Toc464477626)

[Related Documents 2](#_Toc464477627)

[Customer Support 2](#_Toc464477628)

[1 Introduction 5](#_Toc464477629)

[2 NetSpeed Debug Architecture 6](#_Toc464477630)

[2.1 Debug APB Interface 6](#_Toc464477631)

[2.2 Tracing 6](#_Toc464477632)

[2.2.1 NS Trace Probes 7](#_Toc464477633)

Figures

[Figure 1 NetSpeed Debug Architecture 6](#_Toc464477634)

# Introduction

This document covers the set of features provided by NetSpeed NoCs that assist with error logging and handling and post silicon debug. There are several aspects to our solution:

1. Error event capture and reporting: generation and delivery of maskable interrupts, register access via regbus to extract details and to support SW intervention in error handling.
2. Status and statistics: regbus reporting of interesting status, filters and counters (outside of error events).
3. Packet tracing: non-invasive, real time capture of packet data entering and exiting the NoC.
4. Debug port access to regbus: a means of supporting connection of an external agent that can access the NoC register space.
5. ARM CoreSight Compliance: our solution meets the requirements of a “CoreSight reusable component” to allow seamless integration in a platform level CoreSight trace and debug implementation.

We provide a debug APB interface to the programming model (register set) an ATB interface which delivers the captured trace data, an event interface to support cross-triggering and a timestamp interface are provided.

Debug functionality is distributed throughout the elements of the NoC, each piece with its own local set of related registers formatted to meet the requirements of ARM’s specification of a visible component interface.

Access to this register space is provided via an APB port in our regbus tunnel.

MIPI has published a set of specifications related to trace and debug. MIPI STP is a “base protocol” that standardizes common features of the protocols implemented by each trace source, which each have their own specific definitions. Netspeed trace probes define their own protocols, and these are built upon (and are therefore compliant with) MIPI STP v2.

# NetSpeed Debug Architecture

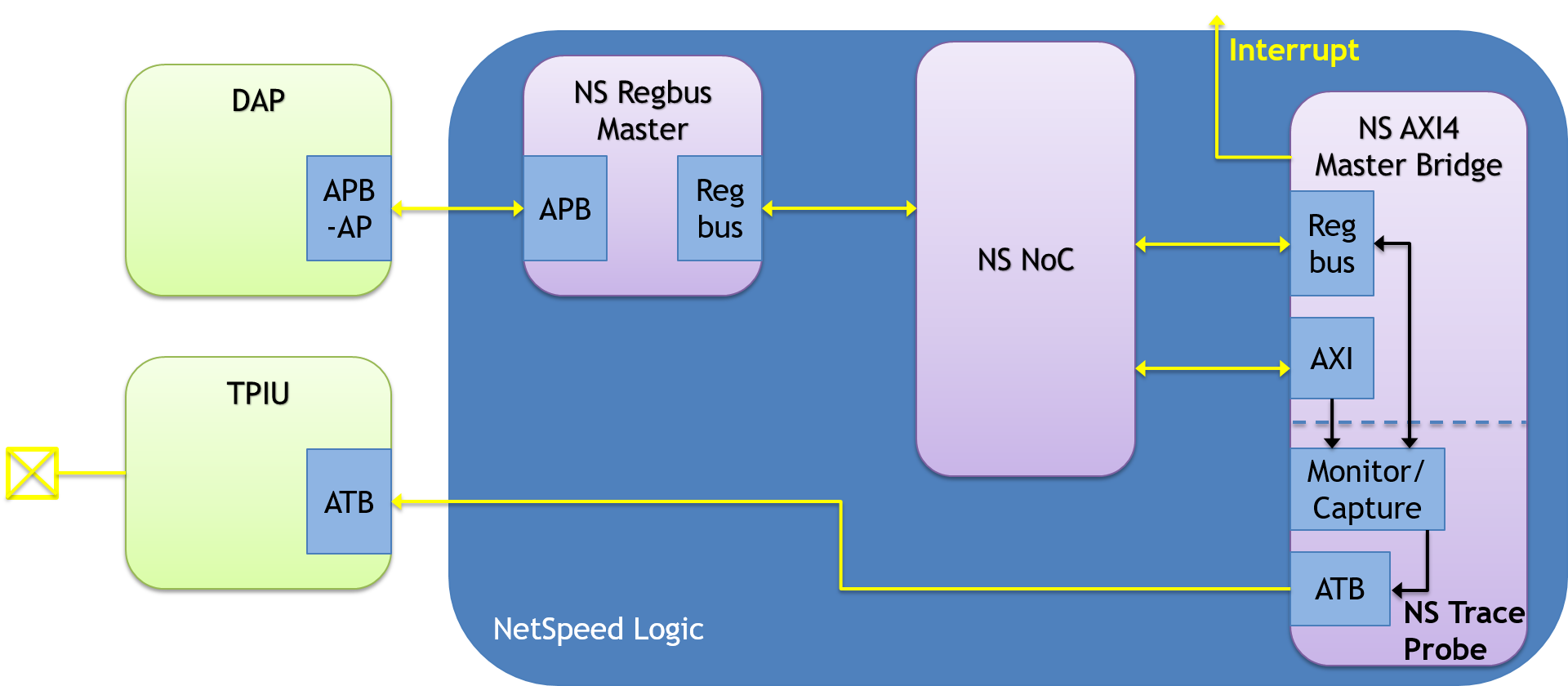


Figure 1 NetSpeed Debug Architecture

1. Debug APB Interface
2. Tracing - NS Trace Probes: instrumentation that is instantiated as companion modules at our master and slave bridges that can capture transactions in real time, filter and format the data into CoreSight compliant ATB streams.
3. Error capture and maskable interrupts
4. Status and statistics registers

## Debug APB Interface

The debug APB Interface: is a 32-bit APB bridge to regbus (as shown above, integrated into regbus master) that provides access to the register space that controls our debug features. The register map for debug features is compliant with CoreSight visible architecture, and it presents the full set of debug features as one or more CoreSight reusable components.

## Tracing

Our tracing features support non-invasive monitoring of transactions at the master and slave interfaces to the NoC. Due to the decentralized nature of NetSpeed NoCs, it is not generally possible to monitor at a single location to observe all or even a significant portion of network activity.

### NS Trace Probes

Our initial implementation supports trace probe modules, for the master and slave bridges of the following protocols: ACE, ACE-lite, CCC/LLC interfaces, AXI, AHB, APB. The functionality is largely common between them and is implemented in a modular fashion to maximize design reuse. Tracing non-invasively monitors the host interface of the bridge being probed and provide a programmable set of filtering and triggering features to give some selectivity in what transactions are captured in a trace and what portions of those transactions are included (i.e., address, control, data, etc.). The trace output is delivered over an ATB interface.

Following is a brief enumeration of filtering and event/trigger generation features.

* Address Comparitors
  + Masters: leverage address map logic for address range events, can filter/trigger per NocStudio defined address range.
  + Masters and Slaves: optional dynamically configurable address & address range match registers
* Control Comparitors: set of configurable control field registers
* Error Status
* External trigger inputs, event outputs (CTI)

#### NS Trace Probes Probes Pin Lists

Bridges that are enabled for tracing will bring out an ATB master (TX) interface as well as some other trace related interfaces.

|  |  |  |  |
| --- | --- | --- | --- |
| **Signal** | **Direction** | **Width** | **Description** |
| atclk | input | 1 | Clock input for ATB interface. Interface operates synchronously w/respect to this clock. |
| atbytes | output | log2(ATB\_DATA\_WIDTH) - 4 | Number of valid bytes in ATDATA. |
| atdata | output | 32, 64 or 128 | Trace data. |
| atid | output | 7 | ID that uniquely identifies source of trace. |
| atready | input | 1 | Downstream ATB slave is ready to accept data. |
| atvalid | output | 1 | Transfer valid this cycle. |
| afvalid | input | 1 | Flush request. |
| afready | output | 1 | Flush acknowledgement. |
| syncreq | input | 1 | Synchronization request. |

**Table 1** ATB Interface Signals

There are some additional interface signals that will be added to support tracing.

|  |  |  |  |
| --- | --- | --- | --- |
| **Signal** | **Direction** | **Width** | **Description** |
| tsvalue | input | 64 | External timestamp value, encoded as natural binary number. Synchronous to noc\_clk. |
| hwevent | input | 1 | CTI input. |
| trigout | output | 1 | CTI output. |
| asyncout | output | 1 | CTI output, asserted for one cycle when ASYNC-VERSION goes out on ATB interface. |
| dbgen | input | 1 | Invasive debug enable. |
| niden | input | 1 | Non-invasive debug enable. |
| spiden | input | 1 | Secure invasive debug enable. |
| spniden | Input | 1 | Secure non-invasive debug enable. |

**Table 2** Other Trace-Related Signals

2670 Seely Ave

Building 11

San Jose, CA 95134

(408) 914-6962

<http://www.netspeedsystems.com>