

CFG Crux

Technical User Manual

Version: CRUX-20.Q1

Revision: 0.0

Intel Confidential

CFG Crux Technical Reference Manual

About This Document

This document is written for system designers, system integrators, and programmers who are designing or programming a System-on-Chip (SoC) using Crux NoC IP.

Audience

This document is intended for users of NocStudio:

* NoC Architects
* NoC Designers
* SoC Architects

Prerequisite

Before proceeding, you should generally understand:

* Basics of Network on Chip technology

Related Documents

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

* CFG NocStudio Crux User Manual
* CFG Crux IP Integration Spec

Customer Support

For technical support about this product and general information, contact CFG Support.

Revision History

|  |  |  |
| --- | --- | --- |
| Revision | Date | Updates |
| 0.0 | Jan 29, 2020 | Initial Version |

Contents

[About This Document 2](#_Toc496629820)

[Audience 2](#_Toc496629821)

[Prerequisite 2](#_Toc496629822)

[Related Documents 2](#_Toc496629823)

[Customer Support 2](#_Toc496629824)

[1 Introduction 8](#_Toc496629825)

[1.1 CFG Crux Overview 8](#_Toc496629826)

[1.2 Configurability Options 8](#_Toc496629827)

[2 Functional Description: System Interconnect 9](#_Toc496629828)

[2.1 NoC Topology 9](#_Toc496629829)

[2.2 Routers 12](#_Toc496629830)

[2.3 Information Transport in the NoC 13](#_Toc496629831)

[2.4 Clocking 18](#_Toc496629832)

[3 Functional Description: Streaming Protocol 23](#_Toc496629833)

[3.1 Bridging from Host to NoC 23](#_Toc496629834)

[4 Deadlock Avoidance 26](#_Toc496629835)

[4.1 Quick Primer on Deadlocks 26](#_Toc496629836)

[4.2 Constructing Deadlock-Free Interconnects 28](#_Toc496629837)

[4.3 Protection against Slave dependency behavior 29](#_Toc496629838)

[5 Quality of Service Support 31](#_Toc496629839)

[5.1 Traffic Classes 32](#_Toc496629840)

[5.2 Strict priority based allocation 34](#_Toc496629841)

[5.3 Weighted bandwidth allocation 37](#_Toc496629842)

[5.4 Rate Limiting Hosts 41](#_Toc496629843)

[5.5 Dynamic Priority Support for Isochronous Traffic 46](#_Toc496629844)

[6 NoC Serviceability: Regbus Layer 47](#_Toc496629845)

[6.1 The Register Bus 47](#_Toc496629846)

[6.2 NoC Registers 52](#_Toc496629847)

[6.3 Error Responses To Register Accesses 53](#_Toc496629848)

[6.4 User Register Bus Access 54](#_Toc496629849)

[6.5 Register Bus Master Interface 55](#_Toc496629850)

[6.6 Expected Usage of Register Bus Master 58](#_Toc496629851)

[6.7 Ring Slave to Host Interface 58](#_Toc496629852)

[6.8 Atomic Operations 59](#_Toc496629853)

[6.9 Restrictions 62](#_Toc496629854)

[7 Programmers Model 64](#_Toc496629855)

[7.1 Router Registers 64](#_Toc496629856)

[7.2 Streaming Bridge registers 75](#_Toc496629857)

[7.3 Regbus Master/Slave Bridge Registers 95](#_Toc496629858)

[8 Appendix A: NetSpeed Streaming Interface Protocol 101](#_Toc496629859)

[8.1 NSIP Transmitter 101](#_Toc496629860)

[8.2 NSIP Receiver 104](#_Toc496629861)

[8.3 Credit Based Flow Control at TX and RX Interfaces 106](#_Toc496629862)

[8.4 Width Ratios and Conversion Summary 107](#_Toc496629863)

[8.5 Ordering Requirements 109](#_Toc496629864)

[8.6 CFG Streaming Bridge Overview 109](#_Toc496629865)

[8.7 Adding Streaming Bridge and Traffic in NocStudio 111](#_Toc496629866)

[8.8 Streaming Bridge Specification 112](#_Toc496629867)

[8.9 QoS 113](#_Toc496629868)

Figures

[Figure 1. Bridge and Router Functions in the NoC 9](#_Toc496629869)

[Figure 2. A 4x4 Homogeneous Grid or Mesh Interconnect 10](#_Toc496629870)

[Figure 3 A 4x4 heterogeneous grid or mesh interconnect 11](#_Toc496629871)

[Figure 4. Schematic Symbol of a NocStudio RTL-Library Router Component 13](#_Toc496629872)

[Figure 5. Information Transport in the NoC 14](#_Toc496629873)

[Figure 6. NoC Packet Organized in Two Different Flit Sizes 15](#_Toc496629874)

[Figure 7. Merging and Dividing Flits with Various Channel Widths 17](#_Toc496629875)

[Figure 8. Combinations of Flit Merging and Division at Router Ports 17](#_Toc496629876)

[Figure 9. Multiple Clock Domains and Clock Crossings 18](#_Toc496629877)

[Figure 10 In-Link Domain crossing structure 20](#_Toc496629878)

[Figure 11. Clock Gating as Implemented in CFG NoC 21](#_Toc496629879)

[Figure 12: CFG Streaming Bridge functions 23](#_Toc496629880)

[Figure 13. Simple Deadlock Graph with Two Resources 27](#_Toc496629881)

[Figure 14. Simple Interconnect with Deadlock 27](#_Toc496629882)

[Figure 15 - System Level Dependency Graph 29](#_Toc496629883)

[Figure 16: System with 2 masters and one slave 35](#_Toc496629884)

[Figure 17: Class/Priority mapping 35](#_Toc496629885)

[Figure 18: Snapshot of Performance Simulation with same class, same priority 36](#_Toc496629886)

[Figure 19:Snapshot of performance simulation with 2 classes, different priority 36](#_Toc496629887)

[Figure 20:Snapshot of performance simulation with 2 classes, same priority 37](#_Toc496629888)

[Figure 21: 3 master, 1 slave system 38](#_Toc496629889)

[Figure 22: Performance results with ratios 3:2:1 39](#_Toc496629890)

[Figure 23 - Token Count increases at specified rate. Packet transmit decrements count. 42](#_Toc496629891)

[Figure 24 - Token Bucket 43](#_Toc496629892)

[Figure 25. Regbus Master Bridge 48](#_Toc496629893)

[Figure 26. Regbus Layer Communication 49](#_Toc496629894)

[Figure 27. Regbus Tunnel Connects Primary NoC Layer to Regbus Layer 50](#_Toc496629895)

[Figure 28: Regbus Address Map 52](#_Toc496629896)

[Figure 29: Register bus master bridge 55](#_Toc496629897)

[Figure 30: Waveform showing read requests and responses at the register bus master interface 58](#_Toc496629898)

[Figure 31: Waveform showing write requests and responses at the register bus master interface 58](#_Toc496629899)

[Figure 32 : Waveform showing ring slave read requests and responses (4B and 8B) 62](#_Toc496629900)

[Figure 33 : Waveform showing ring slave write requests and responses (4B and 8B) 62](#_Toc496629901)

[Figure 34: Timing of Single beat transactions at TX Bridge 103](#_Toc496629902)

[Figure 35: Timing of Multi beat transactions at TX Bridge 104](#_Toc496629903)

[Figure 36: Timing of Single beat transactions at RX Bridge 105](#_Toc496629904)

[Figure 37: Timing of Multi beat transactions at RX Bridge 106](#_Toc496629905)

[Figure 38: Credit based flow control at TX and RX interfaces of NSIP agents. 106](#_Toc496629906)

[Figure 39: Timing of Multi beat transaction with credit flow control at TX Bridge 107](#_Toc496629907)

[Figure 40: Timing of Multi beat transaction with credit flow control at RX Bridge 107](#_Toc496629908)

[Figure 41 Four TX interfaces (left) and four RX interface (right) that may and may not communicate with each other based on interface widths and single beat prop 108](#_Toc496629909)

[Figure 42 Illustration of width conversion of data beats of a message in NSIP protocol from a narrow TX interface to a wide RX interface 109](#_Toc496629910)

[Figure 43 NetSpeed streaming bridge Interfaces to host port and to NoC 110](#_Toc496629911)

[Figure 44 Illustration of a NoC with streaming host ports and traffic between them 111](#_Toc496629912)

Tables

[Table 1 Host and NetSpeed Streaming Bridge Interface 24](#_Toc496629913)

[Table 2 - Rate limiter Configuration Register 44](#_Toc496629914)

[Table 3: Register attribute table 52](#_Toc496629915)

[Table 4: Register bit attribute table 53](#_Toc496629916)

[Table 5: Response table for NoC Register Accesses 53](#_Toc496629917)

[Table 6: Response table for User Register Bus Accesses 54](#_Toc496629918)

[Table 7: Register Bus Master Interface signals 55](#_Toc496629919)

[Table 8: Register slave to host interface 59](#_Toc496629920)

[Table 9 ID register. 65](#_Toc496629921)

[Table 10 RPERR register. 66](#_Toc496629922)

[Table 11 RPERRM register. 67](#_Toc496629923)

[Table 12 RE register. 68](#_Toc496629924)

[Table 13 REC register. 68](#_Toc496629925)

[Table 14 RECC register. 69](#_Toc496629926)

[Table 15 REM register. 70](#_Toc496629927)

[Table 16 RIVCS register. 72](#_Toc496629928)

[Table 17 ROEC register. 73](#_Toc496629929)

[Table 18 ROECC register. 73](#_Toc496629930)

[Table 19 ROVCS register. 75](#_Toc496629931)

[Table 20 BRHST\_CNTR1 register. 76](#_Toc496629932)

[Table 21 BRHST\_CNTR1\_MASK register. 77](#_Toc496629933)

[Table 22 BRPERR0 register. 78](#_Toc496629934)

[Table 23 BRPERR1 register. 79](#_Toc496629935)

[Table 24 BRPERRM0 register. 79](#_Toc496629936)

[Table 25BRPERRM1 register. 80](#_Toc496629937)

[Table 26 BRS register. 81](#_Toc496629938)

[Table 27 BRUS register. 81](#_Toc496629939)

[Table 28 BTPERR register. 82](#_Toc496629940)

[Table 29 BTPERRM register. 83](#_Toc496629941)

[Table 30 BTRL register. 83](#_Toc496629942)

[Table 31 BTS register. 84](#_Toc496629943)

[Table 32 BTUS\_0 register. 85](#_Toc496629944)

[Table 33 BTUS\_1 register. 87](#_Toc496629945)

[Table 34 R register. 88](#_Toc496629946)

[Table 35 RXE register. 89](#_Toc496629947)

[Table 36 RXEM register. 90](#_Toc496629948)

[Table 37 RXID register. 90](#_Toc496629949)

[Table 38 T\_0 register. 91](#_Toc496629950)

[Table 39 T\_1 register. 92](#_Toc496629951)

[Table 40 T\_2 register. 93](#_Toc496629952)

[Table 41 TXE register. 94](#_Toc496629953)

[Table 42 TXEM register. 94](#_Toc496629954)

[Table 43 TXID register. 95](#_Toc496629955)

[Table 44 AM\_BRIDGE\_ID register. 96](#_Toc496629956)

[Table 45AM\_ERR register. 97](#_Toc496629957)

[Table 46 AM\_INTM register. 99](#_Toc496629958)

[Table 47AM\_NOCVER\_ID register. 99](#_Toc496629959)

[Table 48 AM\_STS register. 100](#_Toc496629960)

[Table 49 NoC Streaming Bridge TX signals from Streaming Host port to NoC 102](#_Toc496629961)

[Table 50 NoC Streaming Bridge RX signals from NoC to Streaming Host port 105](#_Toc496629962)

# Introduction

## CFG Crux Overview

CFG Crux is a scalable, high-performance Network-on-chip (NoC) IP that is used for rapidly designing and analyzing highly efficient and scalable interconnects for a wide variety of SoCs. To quickly produce efficient, high-performance NoC IPs, Crux uses a requirements-driven design approach. Crux uses number of state-of-the-art algorithms to provide robust end-to-end QoS and application-level deadlock avoidance. The solution can scale from low- to medium-end SoCs with 10s of IP blocks to high-end SoCs with 100s of IP blocks and provides bandwidth scaling, optimal latency and clock frequencies of up to 3 GHz. Crux is built upon following the following fundamental design principles.

### Requirements driven approach

Crux is configured and optimized using NocStudio - a NoC architecture exploration platform and interconnect synthesizer. NocStudio enables architects to design, configure and simulate CFG’s NoC IP as well as evaluate multiple SoC architectures. The interconnect can be designed and customized based on system requirements such as bandwidth, latency, traffic profiles, as well as fine-grained requirements such as total and per-flow system bandwidth and chip layout.

### Physically aware latency optimized design

Crux design is physically aware of the layout of the on-chip system components producing an interconnect topology that is customized for the SoC layout. Being physically aware ensures that wiring congestions does not occur late in the design cycle and appropriate number of buffers and pipeline stages are present at various fabric channels to enable smooth backend design. Latency sensitive traffic can use dedicated connections to reduce arbitration and congestions, and 16 levels of QoS are supported for fine-grained bandwidth allocation and prioritization. Based on the system traffic specification and SoC physical layout, NoC topology, fabric components, and their placements are automatically computed using machine-learning and graph theory algorithms to optimize the design for area and power.

## Configurability Options

CFG Crux provides user configurability and flexibility across multiple design dimensions. In addition to providing flexibility of number of ports, interface widths, layout portioning, power and voltage partitioning, etc., significant configurability is also provided to the architect in defining the NoC topology as well as other system level characteristics.