CFG UFI Bridges

Functional Specifications

Joji Philip

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Author | Description |
| 0.1 | 02/23/2020 | Joji Philip | Initial draft of UFI microarchitecture specification.  This contains details on block partitioning and implementation consideration of UFI channels. It also contains details of credit management and initialization state machines. |
| 0.2 | 04/06/2020 | Om Prakash Hari | Updated block diagrams to reflect implemented partitioning |
|  |  |  |  |

CFG UFI Bridges

[1 Introduction 2](#_Toc37091599)

[1.1 Block diagram 3](#_Toc37091600)

[1.2 Functional description 5](#_Toc37091601)

# Introduction

Broadly UFI can be described as an encapsulation protocol which allows a set of Intel standard protocols supported by an agent to share wires on its interface with the fabric. Currently UFI supports IDI, IDI\_SA, UPI\_COH, UPI\_NC and CXL.MEM protocols. A specific set of combinations of these protocols are supported by different agents.

Figure 1 From the UFI spec, shows the set of physical channels supported by the UFI protocol.



Figure 1 UFI Physical channels

Refer to UFI specification for further details.

From CFG fabric support perspective, NocStudio will now allow agents with native UFI interfaces to be added. A UFI bridge will be instantiated to convert the UFI interface to CFG NoC protocol. UFI bridge forms the fabric side of a UFI interface. Bridge’s A2F interface receives transactions form the agent and its F2A interface transmits transactions from the fabric to the agent.

UFI bridge will be configurable through parameters for

* Profile of supported protocols
* Number of physical instances of each A2F and F2A channels

## Block diagram



Figure 2 High level block diagram

### A2F REQ channel



Figure A2F REQ channel

### A2F RSP channel



Figure A2F RSP channel

### A2F DATA channel



Figure A2F DAT channel

### F2A channels

Each instance of F2A REQ, RSP and DATA channels use a virtualized RX ready-valid interface of RSSB. Interface host VC credit available status is provided to RSSB RX interface. This credit status is utilized to arbitrated among the NoC virtual channels mapping to the host VCs and post a transaction on the RX interface. Since credits have already been checked, the transaction makes progress on the interface in a timely manner without blocking other virtual channels.

For IDI protocol support, exiting IDI master bridge U2C REQ, U2C RSP and U2C DATA channels will be reused. IDI protocol doesn’t use UFI virtual channels on F2A physical channels.



Figure 6 High-level F2A physical channels

## Functional description

### Maximizing resource sharing among protocols

A2F REQ, RSP and DAT have dedicated channel modules. Each channel module connects to a single virtualized TX RSSB interface. For bridge configs with multiple A2F channels of a type, corresponding module and TX interfaces are instantiated multiple times. Packets from the UFI interface first undergo address decode and route lookup after extracting the required fields from the interface transaction. Results of the lookup operation, including the route and target NoC layer, VC information is put into the interface VC queues. Arbitration is performed among the head of these queues which have NoC side credit available for the targeted layer and VC. Selected transaction is then processed by the appropriate protocol handling block to frame NoC packet and request the TX interface of RSSB. A given bridge instance will have a specific set of protocol processing blocks enabled, based on the agent profile.

### Dedicating resources per protocol



Figure 7 Dedicated processing resource per protocol

In some cases the protocols on a channel may significantly differ in the address lookup process, the size of their transaction packet formats and hence queue size etc. In such cases, it might be required to have a completely dedicated path for each protocol. Note however that since a single protocol channel transaction arrives on the interface every cycle, the BW speedup provided by the parallel path is not required or utilized.

### UFI interface block

This block is responsible with the initialization and shut down of a UFI interface. After reset or a power wakeup event the two end points must initialize credit and reach a fully connected state before transactions can be sent. Similarly, prior to a power shutdown on the interface, the two end points have to co-ordinate entering a clean quiescent state.

Initialization and disconnect are coordinated through the following set of signals on both A2F and F2A

|  |  |  |
| --- | --- | --- |
| Signal | Direction (for A2F physical channel) | description |
| \*\_txconn\_req | Agent to fabric | Con/discon request |
| \*\_rxconn\_ack | Fabric to agent | Con/discon Ack |
| \*\_rxconn\_pchange\_req | Fabric to agent | Fabric initiating a state change |
| \*\_rx\_empty | Fabric to agent | Fabrics idle status |

#### 

#### Initialization and shut-down

* After exit from reset the RX holds credits for all the virtual channels and these must be passed to TX before flit transfer can begin
* During normal operation, flits and credits are exchanged between the RX and TX ends
* Shared credit is sent from RX to TX and is not associated with any VC
* Dedicated credits are sent on a multiplexed credit link



* A 4-way handshake mechanism is used to bring the link up and down
* There are 4 states :
  + Disconnected
  + Connecting (Transient state)
  + Connected
  + Disconnecting (Transient state)
* After exit from reset, the link is in “disconnected” state.
* Sequence of behavior at the end points in the states are described below

|  |  |  |
| --- | --- | --- |
| State | TX | RX |
| Disconnected | Must not send transactions  Will not receive credits | Will not receive transactions  Must not send credits |
| Connecting | Must not send transactions  Must accept credits (race) | Will not receive transactions  Must not send credits |
| Connected | Can send transactions  Must accept credits | Must accept transactions  Can send credits |
| Disconnecting | Must not send transaction  Must accept credits | Must accept transactions (race)  Must stop sending credits |

RX state transition

|  |  |  |
| --- | --- | --- |
| Present State | Next state | Condition for transition |
| Disconnected | Connecting | txconn\_req is sampled high |
| Connecting | Connected | RX end is ready to receive transactions and send credits |
| Connected | Disconnecting | txconn\_req is sample low |
| Disconnecting | Disconnected | Wait for sufficient duration for outstanding transaction from Tx to be received and link to go idle |

TX state transition

|  |  |  |
| --- | --- | --- |
| Present State | Next state | Condition for transition |
| Disconnected | Connecting | Options:   1. Link is required to send new transactions 2. On exit from reset 3. When the RX path enters connecting state |
| Connecting | Connected | rxconn\_ack is sampled high |
| Connected | Disconnecting | Options:   1. No new transactions requiring the link (idle + hysteresis) 2. Qchannel initiating a power down event 3. RX path enters disconnecting state |
| Disconnecting | Disconnected | rxconn\_ack is sampled low |

### Route lookup

#### A2F REQ Address decoding

* Programmable {base address, mask} or {high limit, low limit} per range
* Programmable system address reloc and slave address reloc per range
* Programmable SAI check for read and write access permission per range
* Ranges specifiable as IMR or security filter
* Intel hash function or simple hash function specifiable per range
* Root space check

##### IMR or security filter

* Special ranges defined IMRs. Not associated with any slave or destination
* A match against this range causes the transaction to be rejected
* This has highest priority and overrides exiting 2 levels of priority (foreground/Hi, Background/Lo)

##### Intel hash

* Each range can either support simple hash or Intel hash function

#### A2F DATA and A2F RSP ID lookup

For IDI protocol support, exiting C2U RSP and C2U DATA ID based route lookup will be used. Associative table contains destination logical IDs and corresponding physical route and NoC VC.

A more generic ID based lookup module will be required for supporting UPI and other protocols

* Primary table will comprise of logical ID of destinations
* Secondary table will comprise of key factoring {host VC, opcode} and value providing the physical route and NoC VC information

### Input VC queues

A2F channels can support multiple protocols and multiple virtual channels per protocol. Each virtual channel has a credit-based flow control with the transmitter on the A2F interface. An input queue is needed for every virtual channel of every protocol on the interface. These queues must be able to drain into the NoC independent of each other. Each queue will be structured for FIFO ordering of transactions within a virtual channel.

Address decode/route look up operation identifies the route to NoC destination and the NoC layer and VC to be taken. This information is written into the Q along with the original transaction fields. Head of all the VC buffers and the targeted NoC VC information is visible for selecting and reading transactions from the queue and sending it on to the NoC.

* Number enabled queues should be parameter controlled
* Queue ID is {\*\_protocol\_id, \*\_vc\_id}. Single protocol bridge should be supported

#### Flip-flop based dedicated FIFO queues for VCs

In the simplest instance, each A2F virtual channel will be an independent flip-flop based FIFO buffer. Based on the number of protocols and VCs, these FIFO buffers must be instantiated under parameter control.



* Incoming transaction’s *protocol\_id[1:0]* and *vc\_id[2:0]* are decoded to demux the writes to the corresponding FIFO buffer
* FIFOs have a prefetched registered output
* This is only suitable if a relatively small number of VCs are present on the interface. The credit loop latency also must be low to allow these VC FIFO buffers to be shallow.

#### RF based storage with fixed partitions for VC FIFOs



* If a large number of VC queues are required and the credit latencies are large requiring several entries for each VC, than an RF based storage is recommended
* An RF with 1 WR and 1 RD port is required
* 1 or 2 deep flip-flop based prefetch stages are required for each VC. These stages maintain prefetched and registered head entries of the VC queues
* When empty, the RF is bypassed to directly write incoming transactions into the corresponding prefetch stage.
* When prefetch stage are full, new entries are written into the incoming VC’s partition in the RF
* A prefetch filler logic continuously reads data from the RF and fills the prefetch stages. It monitors credits from the prefetch stages, occupancy of VC queues in the RF and selects a VC to read and fill its prefetch stage.

#### RF based storage with shared dynamic FIFO queues for VCs

In a statically partitioned scheme, if a VC runs out of space in its partition, it cannot use entries from another VC even if they are unused. Further, to allows all VCs to independently maintain full bandwidth on the link, each VCs space will need to be sized to the full credit loop latency.

To improve utilization of the storage space, a shared buffer with dynamic allocation can be built. This will look like the scheme explained in the previous section. But the RF structure won’t have fixed static regions reserved for each VC. Instead it will be considered a pool of free entries. Entries from the pool will be allocated dynamically to linked-list queues per VC.



### Protocol processing

These blocks will perform protocol specific features such as standard packetization, format compression, etc. IDI processing channels will be reused from the IDI master bridge.

### Switch interface

A2F REQ and A2F RSP have single flit transactions and will use a single virtualized TX ready-valid interface to RSSB. With single flit transactions, there are no blocking issues related to multiple host VCs sharing a common TX interface. Any host VC transaction placed on the interface is granted by the NoC side in a timely manner as NoC credit availability has already been checked for.

A2F Data channel does not interleave flits from multiple host side VCs. At a time, complete packets are sent and 64B or 32B data packets are possible. Interface is 32B, so each packet is at most 2-flits. To make non-blocking progress, destination NoC VC must have at least 2 flit credits before a host VC can participate in arbitration for the TX interface. Note however that the 2 flits from the host VC can arrive with empty bubbles in between, thus unfairly consuming the bandwidth of the TX interface shared with other VCs. Optionally, host VC can be configured for store and forward operation such that they can send both flits on back to back cycles on the TX interface.

Note on UPI A2F data: TBD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UFI Channel | Type of interface | Host interface | Credit check | RSSB interface |
| A2F REQ | TX Virtualized RDY/VLD | Single flit | Flit credit from NoC | HDR/SB only |
| A2F RSP | TX Virtualized RDY/VLD | Single flit | Flit credit from NoC | HDR/SB only |
| A2F DATA | TX Virtualized RDY/VLD | 2x 32B flits | Packet credit 64B from NoC | HDR/SB and PAYLD/DATA.  HDR-DATA separation possible |
| F2A REQ | RX Virtualized RDY/VLD | Single flit | Flit credit from host VC to NoC | HDR/SB only |
| F2A RSP | RX Virtualized RDY/VLD | Single flit | Flit credit from host VC to NoC | HDR/SB only |
| F2A DATA | RX Virtualized RDY/VLD | 2x 32B (non-Intlv)  1x 32B (intlv) | Flit or packet credit from host VC to NoC | HDR/SB and PAYLD/DATA.  HDR-DATA separation possible |

### Programmable registers

* Programmable address range registers with base, mask, reloc, SAI etc
* Status registers, interrupts, mask etc
* Performance monitoring events

### Clock, power, voltage domain crossing

* UFI bridge is a single clock synchronous design
* Clock, power, voltage domain boundary can be on the UFI link to the agent
* Clock, power, voltage domain boundary using ILDC on the NoC link to routers

### Low power

* Tracking structures inherited from IDI master bridge

### FuSa

* Address parity error checked in the input stage and passed through as decode error
* Enhanced parity of IDI (can we use interface fatal)
* Interleaved F2A DATA : IDI channel always assumes interleaved (Rob)

Parameters: from the doc, number of VCs

Header data sep

F2A interleave

Link block signal