OmniXtend version 0.1 specification

Version 20181204

Table of Contents

TileLink is an interconnect standard providing multiple masters with coherent memory-mapped access to a collection of memories and slave devices. TileLink has primarily been used to connect tightly coupled components on a single chip. This document describes how the TileLink protocol is serialized for transmission over chip-to-chip interconnects, and should be read in conjunction with the TileLink specification (currently version 1.7-draft).

1. TileLink Physical Links

TileLink can be carried over a variety of chip-to-chip physical links, from simple parallel busses to bundles of high-speed serial lanes. The physical link must support bidirectional communication between chips.

1.1. TileLink Parallel Link

The SiFive FU540 includes a simple parallel synchronous bus to connect to an expansion FPGA. The bus link is 32 bits wide in both directions.

1.2. TileLink Ethernet Link

The TileLink protocol can be carried over an Ethernet link. This serialization forms the basis of the OmniXtend v0.1 specification. Future evolution of the serialized OmniXtend protocol will likely cause it to diverge from the on-chip TileLink and become either a superset, or require

translation at the point of de/serialization.

2. TileLink Serializer

Each end of the physical link is connected to a TileLink serializer. The two TileLink serializers cooperate to multiplex two TileLink master-slave links, one in each direction, across the bidirectional physical link.

2.1. TileLink Serialization Parcels

Each serialized TileLink message is encoded as a sequence of fixed-width words, or parcels. The parcels are either 32-bit, 64-bit, or 128-bit wide, depending on the physical link parameters.

> **NOTE** Larger parcels simplify hardware design and support higher speed communication, but some link bandwidth is lost to internal fragmentation at larger parcel sizes.

This document version only discusses 32-bit parcel sizes.

> **NOTE** First prototype will be 32-bit parcels, but we're likely to want to move up to 64-bit or 128-bit parcels for most links.

2.2. TileLink Message Encodings

Each TileLink message on Channel A—E is encoded as a single-parcel header followed by zero or more additional parcels.

2.2.1. TileLink Header Format

Position	Width	Name	Description
31:16	16	Source	TileLink transaction ID
15:13	3	Domain	Ordering domain ID
12: 9	4	Size	Logarithm of number of bytes in transaction
8: 6	3	Param	Parameter
5: 3	3	Opcode	Operation code, identifies type of message
2: 0	3	Format	Channel number 0-5 (A-F)

The Source, Size, Param, and Opcode fields follow the TileLink specification.

NOTE

The source field is probably too small at 16 bits, but will grow with larger parcel sizes.

The Ordering Domain ID is used to provide message-ordering guarantees within different subsets of messages. A domain ID of 0 indicates that no ordering guarantees are required. A non-zero domain ID guarantees that the message is transmitted in FIFO order with respect to earlier messages with the same Domain ID.

The Format field indicates the message channel (A-F):

```
Format  Message Format
    0   Channel A
    1   Channel B
    2   Channel C
    3   Channel D
    4   Channel E
    5   Channel F (Credits)
    6   Reserved (User)
    7   Reserved (Escape)
```

2.2.2. TileLink Channel A Encoding

Channel A messages contain the following components:

```
Channel A Format

Data-full messages (PutFullData, ArithmeticData, LogicalData)
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)
    3     Data parcel 0
    ...
    N+2   Data parcel N-1

The number of data parcels, N, is ceil(2^Size/Parcel Bytes).

Data-less message (Get, Hint, AcquireBlock, AcquirePerm)
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)

PutPartial message
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)
    ...
    N+2   Rotated data parcel N-1

PutPartial messages must include the byte mask in addition to the data.
To ensure that the protocol can stream a transaction from the on-chip
bus, the data and mask must be available at the same time. For this
reason, the data parcels are encoded as 0=mask, 1-8=data, 9=mask,
10-17=data, ...

The number of data parcels, N, is ceil(9*2^Size/Parcel Bits).
```

2.2.3. TileLink Channel B Encoding

```
Same as A, but Domain=Source=0 is required.
```

2.2.4. TileLink Channel C Encoding

```
Same as A, but Domain=0.  If not a Release message, also Source=0.
Data-full messages (AccessAckData, ProbeAckData, ReleaseData)
Data-less messages (AccessAck, HintAck, ProbeAck, Release)
```

2.2.5. TileLink Channel D Encoding

```
Data messages (AccessAckData)
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)
    3     Data parcel 0
    ...
    N+2   Data parcel N-1


Data-less messages (AccessAck, HintAck, ReleaseAck)
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)
    3     Data parcel 0
    ...
    N+2   Data parcel N-1


Grant messages (Grant)
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)
    3     31:16 reserved, 15:0 Sink


GrantData messages (GrantData)
Parcel
    0     Header
    1     Hi32(address)
    2     Lo32(address)
    3     31:16 reserved, 15:0 Sink
    4     Data parcel 0
    ...
    N+3   Data parcel N-1


In all cases, N is ceil(2^Size/Parcel Bytes).
```

2.2.6. TileLink Channel E Encoding

```
All fields should be 0, except Source and Format.
The Source field is used to carry the TileLink Sink.
```

2.2.7. TileLink Serializer Flow Control and F Channel

The TileLink Serializer uses credit-based flow control to ensure every parcel will be received by the other side. A transaction is only transmitted over the physical link once the sender has enough credit forr buffer space on the receiving side sufficient to fit all of the parcels in the transaction. A sixth channel, the F channel, is added to the original five TileLink channels (A—E) of each link, to carry credit information for each of the original five channels from the slave side back to the sending master TileLink Serializer.

Links start with no credits for any channel.

Due to guaranteed buffer space and independent buffering of each channel, deadlock cannot occur. Therefore, any work-conserving policy may be used to select the channel to transmit.

Channel F Single-Parcel Header Format

Position	Width	Name	Description
31:27	5	ECredit	Credit for E channel
26:22	5	DCredit	Credit for D channel
21:17	5	CCredit	Credit for C channel
16:12	5	BCredit	Credit for B channel
11: 7	5	ACredit	Credit for A channel
6: 3	4	Zero	0
2: 0	3	Format=5	Channel number 0-5 (A-F)

The 5-bit credit fields use a logarithmic encoding of credit values, where a credit field value of x encodes a value of $2^{(x-1)}$ credits for $x > 0$.

Credit	Value
0	0
1	1
2	2
3	4
4	8
5	16
6	32
7	64
8	128
9	256
10	512
11	1024
12	2048
13	4096
...	

3. TileLink Ethernet Frame Control

The TileLink messages are encoded as Layer 1 Ethernet packets, with a standard preamble followed by a start frame delimiter. The TileLink message header parcel is transmitted next, followed by remaining parcels, then finally the CRC checksum and end of frame marker.

Last updated 2018-12-03 23:57:39 PDT