

**IN THE UNITED STATES DISTRICT COURT  
FOR THE NORTHERN DISTRICT OF ILLINOIS  
EASTERN DIVISION**

HFT SOLUTIONS, LLC,

*Plaintiff,*

v.

Citadel Securities LLC

*Defendant.*

Case No. 1:24-cv-13213

**JURY TRIAL DEMANDED**

**PLAINTIFF HFT SOLUTIONS, LLC’S PRELIMINARY DISCLOSURE OF ASSERTED  
CLAIMS AND INFRINGEMENT CONTENTIONS (LPR 2.2)**

Pursuant to LPR 2.2, Plaintiff HFT Solutions, LLC (“HFT”) submits the following Preliminary Disclosure of Asserted Claims and Infringement Contentions. This disclosure is based on the information available to HFT as of the date of this disclosure, and HFT reserves the right to amend this disclosure to the full extent permitted, consistent with the Court’s Rules and Orders.

**I. LPR 2.2(a): Asserted Claims**

HFT asserts that Defendant Citadel Securities LLC (“Citadel”) infringes the following claims (collectively, “Asserted Claims”):

- (1) U.S. Patent No. 10,931,286 (the “286 patent”), claims 1-4, 6-8, 10-13, 16-22;
- (2) U.S. Patent No. 11,128,305 (the “305 patent”), claims 1-11 and 14-22; and
- (3) U.S. Patent No. 11,575,381 (the “381 patent”), claims 1-12.

For each Asserted Claim, HFT asserts infringement under 35 U.S.C. §§ 271(a), 271(b), 271(c), 271(f), and 271(g).

**II. LPR 2.2(b): Accused Instrumentalities of which HFT is aware**

HFT asserts that the Asserted Claims are infringed by the various instrumentalities used,

made, sold, offered for sale, or imported into the United States by Citadel, including GGPA systems and platforms, including trading systems and platforms that include FGPA boards such as the Bittware XUP-VV8 and AMD Alveo UL3524 and UL3422 FPGA boards. On information and belief, Xilinx, maker of the FPGAs in the Bittware XUP-VV8 board, was acquired by AMD in June 2022.

Citadel's Accused Instrumentalities of which HFT is presently aware are described in more detail in the accompanying preliminary infringement contention charts, Exhibits A-C.

HFT reserves the right to accuse additional products and/or systems from Citadel to the extent HFT becomes aware of additional products and/or systems during the discovery process. Unless otherwise stated, HFT's assertions of infringement apply to all variations, versions, and applications of each of the Accused Instrumentalities, on information and belief, that different variations, versions, and applications of each of the Accused Instrumentalities are substantially the same for purposes of infringement of the Asserted Claims.

### **III. LPR 2.2(c): Claim Charts**

HFT's analysis of Citadel's products is based upon limited information that is publicly available, and based on HFT's own investigation prior to any discovery in these actions. Specifically, HFT's analysis is based on certain limited resources that evidence certain products made, sold, used, or imported into the United States by Citadel.

HFT reserves the right to amend or supplement these disclosures for any of the following reasons:

- (1) Citadel and/or third parties provide evidence relating to the Accused Instrumentalities;
- (2) HFT's position on infringement of specific claims may depend on the claim constructions adopted by the Court, which has not yet occurred; and

(3) HFT's investigation and analysis of Citadel's Accused Instrumentalities is based upon public information and HFT's own investigations. HFT reserves the right to amend these contentions based upon discovery of non-public information that HFT anticipates receiving during discovery.

Attached as Exhibits A-C, and incorporated herein in their entirety, are charts identifying where each element of the Asserted Claims are found in the Accused Instrumentalities.

Unless otherwise indicated, the information provided that corresponds to each claim element is considered to indicate that each claim element is found within each of the different variations, versions, and applications of each of the respective Accused Instrumentalities described above.

#### **IV. LPR 2.2(d): Doctrine of Equivalents**

With respect to the patents at issue, each element of each Asserted Claim is considered to be literally present. HFT also contends that each Asserted Claim is infringed or has been infringed under the doctrine of equivalents in Citadel's Accused Instrumentalities. Further information within the scope of LPR 2.2(d) is included in the attached claim charts.

#### **V. LPR 2.2(f): Priority Dates**

The Asserted Claims of the '286 patent are entitled to a priority date at least as early as November 5, 2018 the filing date of Provisional Application No. 62/755,804.

The Asserted Claims of the '305 patent are entitled to a priority date at least as early as November 5, 2018 the filing date of Provisional Application No. 62/755,804.

The Asserted Claims of the '381 patent are entitled to a priority date at least as early as November 5, 2018 the filing date of Provisional Application No. 62/755,804.

A diligent search continues for additional responsive information and HFT reserves the right to supplement this response.

**VI. LPR 2.2(g): Willful Infringement**

HFT alleges willful infringement by Citadel. Pursuant to paragraph 2(e) of the Case Management Order, Dkt. No. 25, HFT reserves the right to pursue discovery regarding willful infringement.

**VII. LPR 2.2(h): Identification of Instrumentalities Practicing the Claimed Invention**

At this time, HFT does not identify any of its instrumentalities as practicing the Asserted Claims. A diligent search continues for additional responsive information and HFT reserves the right to supplement this response.

Dated: June 6, 2025

Respectfully submitted,

/s/ Dale Chang

Dale Chang

Marc A. Fenster (pro hac vice)

Brian D. Ledahl (pro hac vice)

Paul A. Kroeger (pro hac vice)

Joshua Scheufler (pro hac vice)

RUSS AUGUST & KABAT

12424 Wilshire Blvd. 12th Floor

Los Angeles, CA 90025

Tel: 310-826-7474

Fax: 310-826-6991

ATTORNEYS FOR PLAINTIFF,  
HFT WIRELESS, LLC

**CERTIFICATE OF SERVICE**

I certify that this document is being served upon counsel of record for Citadel on  
June 6, 2025 via e-mail.

/s/ Dale Chang  
Dale Chang

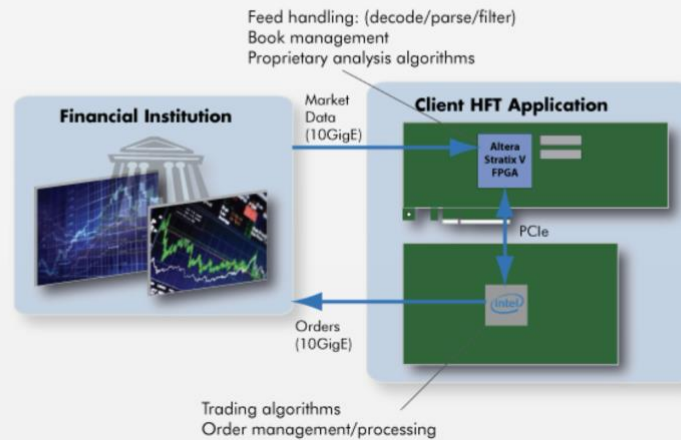
### Exhibit 3

Defendant’s use of the Accused Instrumentalities—including, for example, FPGA systems that use Bittware XUP-VV8 or AMD Alveo UL 3524 or UL3422 boards—infringes at least the following claims of the ’381 patent.<sup>1</sup>

1. A method for processing a first serial data stream, using a field programmable gate array system, to generate a second serial data stream, wherein the method comprises the steps of:

The Accused Instrumentalities perform “A method for processing a first serial data stream, using a field programmable gate array system, to generate a second serial data stream,” e.g.:

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer’s high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.

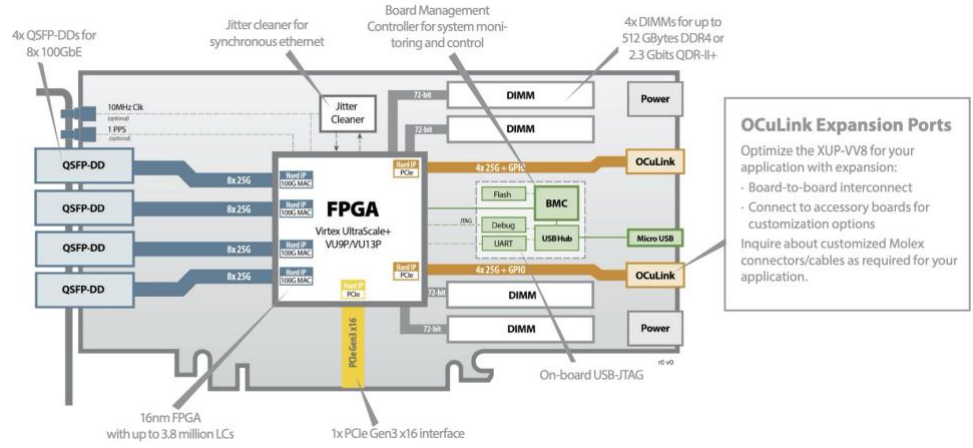


[\(https://www.bittware.com/about/financial/\)](https://www.bittware.com/about/financial/)

At Citadel Securities, a leading global market maker, our team of FPGA Engineers create next generation FPGA solutions to support ultra-low latency trading systems across the firm. FPGA Engineers work closely with business leaders to design and deploy FPGA solutions that help optimize trading system performance.

[\(https://www.citadelsecurities.com/careers/details/fpga-engineer/\)](https://www.citadelsecurities.com/careers/details/fpga-engineer/)

<sup>1</sup> On information and belief, Defendant configures all Accused Instrumentalities to operate in a similar manner for purposes of infringement such that the Bittware XUP-VV8 is representative of all Accused Instrumentalities, including AMD Alveo boards.

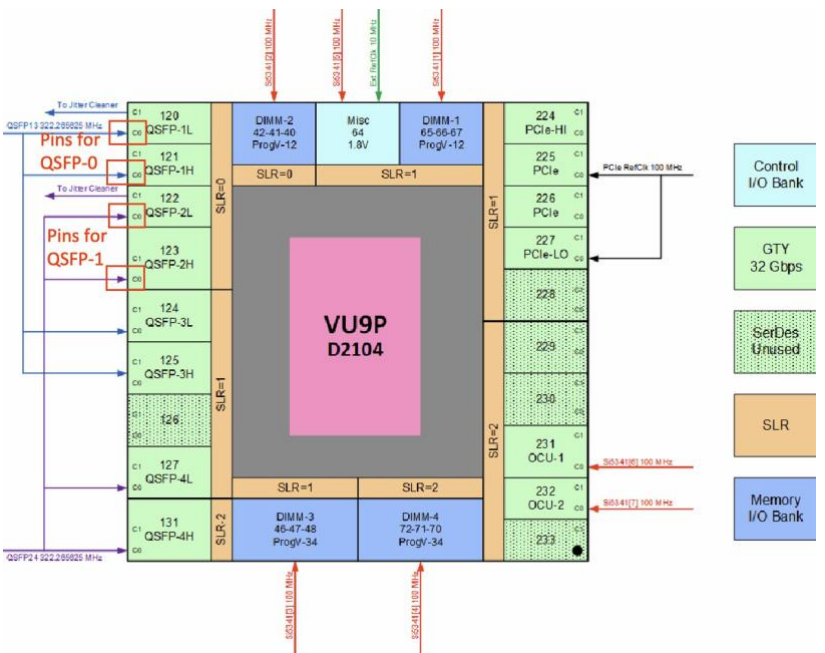


**OCuLink Expansion Ports**  
 Optimize the XUP-VV8 for your application with expansion:  
 - Board-to-board interconnect  
 - Connect to accessory boards for customization options  
 Inquire about customized Molex connectors/cables as required for your application.

[https://www.bittware.com/files/XUP-VV8\\_datasheet\\_r0v0.pdf](https://www.bittware.com/files/XUP-VV8_datasheet_r0v0.pdf)

(a) receiving, by a deserializer in a field programmable array, a clock signal;

The Accused Instrumentalities perform “receiving, by a deserializer in a field programmable array, a clock signal,” e.g.:





(b) receiving, by the deserializer, the first serial data stream;

The Accused Instrumentalities perform “(b) receiving, by the deserializer, the first serial data stream,” e.g.:

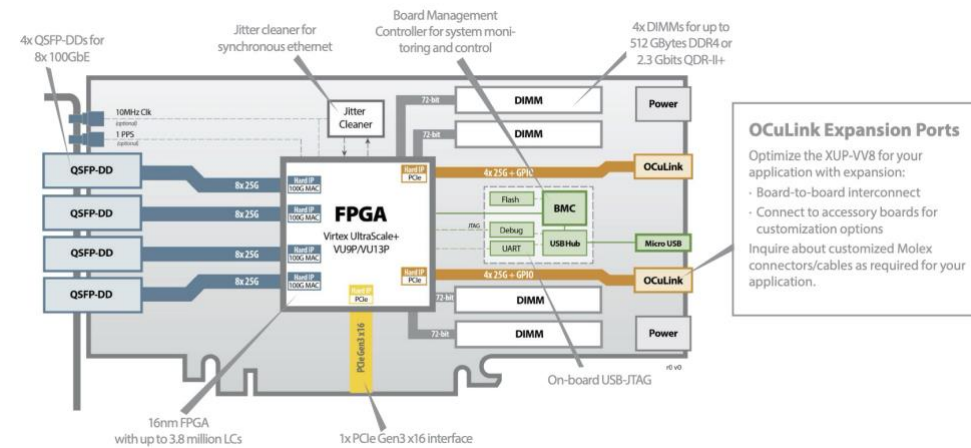
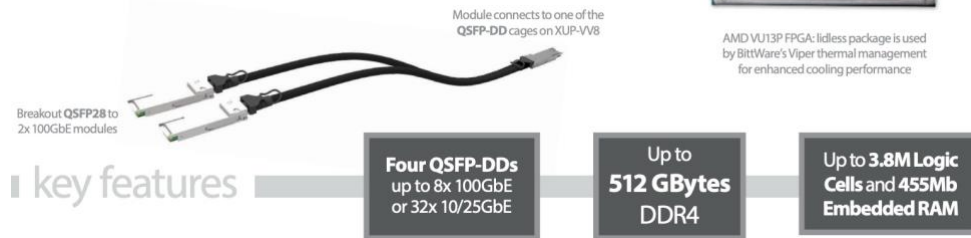
### 8x 100GbE Network Ports and VU9P/13P FPGA

The XUP-VV8 offers a large AMD FPGA in a 3/4-length PCIe board featuring QSFP-DD (double-density) cages for maximum port density. Using the Virtex UltraScale+ VU13P or VU9P FPGA, the board supports up to 8x 100GbE or 32x 10/25GbE.

The FPGA provides large logic and memory resources—up to 3.8M logic cells and 455Mb embedded memory. The board also provides a jitter cleaner to support synchronous ethernet. The board can be configured as single width for users who don't need external memory on the DIMMs.

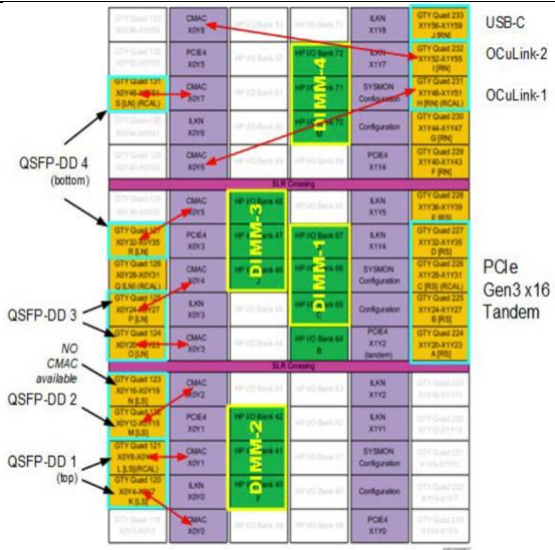


AMD VU13P FPGA: lidless package is used by BittWare's Viper thermal management for enhanced cooling performance



**OCuLink Expansion Ports**  
 Optimize the XUP-VV8 for your application with expansion:  
 • Board-to-board interconnect  
 • Connect to accessory boards for customization options  
 Inquire about customized Molex connectors/cables as required for your application.

([https://www.bittware.com/files/XUP-VV8\\_datasheet\\_r0v0.pdf](https://www.bittware.com/files/XUP-VV8_datasheet_r0v0.pdf))



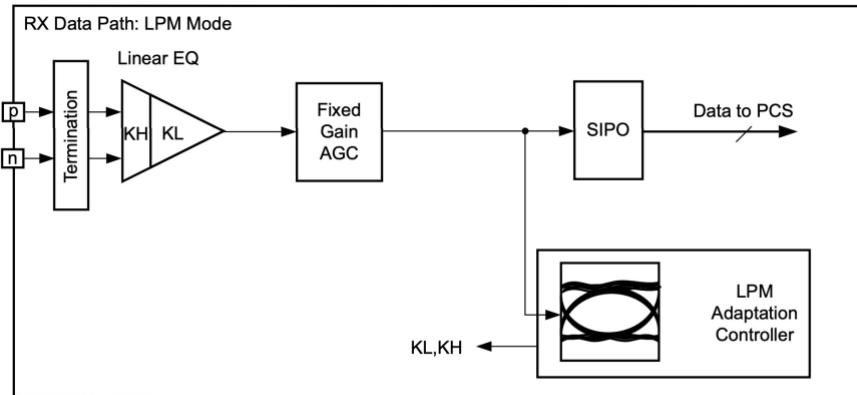
“The FPGA is used to implement the PCIe Gen3, DDR4, QSFP-DD, and Board Management Controller interfaces”

(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

## RX Overview

### Functional Description

This section shows how to configure and use each of the functional blocks inside the receiver (RX). Each GTY transceiver includes an independent receiver, made up of a PCS and a PMA. Figure 4-1 shows the blocks of the GTY transceiver RX. High-speed serial data flows from traces on the board into the PMA of the GTY transceiver RX, into the PCS, and finally into the interconnect logic. Refer to Figure 2-11, page 44 for the description of the channel clocking architecture, which provides clocks to the RX and TX clock dividers.



X19659-082217

Figure 4-12: LPM Mode

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

(c) generating, by the deserializer, a receiver side clock signal;

The Accused Instrumentalities perform “generating, by the deserializer, a receiver side clock signal,” e.g.:

## RX Fabric Clock Output Control

### Functional Description

The RX clock divider control block has two main components: serial clock divider control and parallel clock divider and selector control. The clock divider and selector details are illustrated in Figure 4-16.

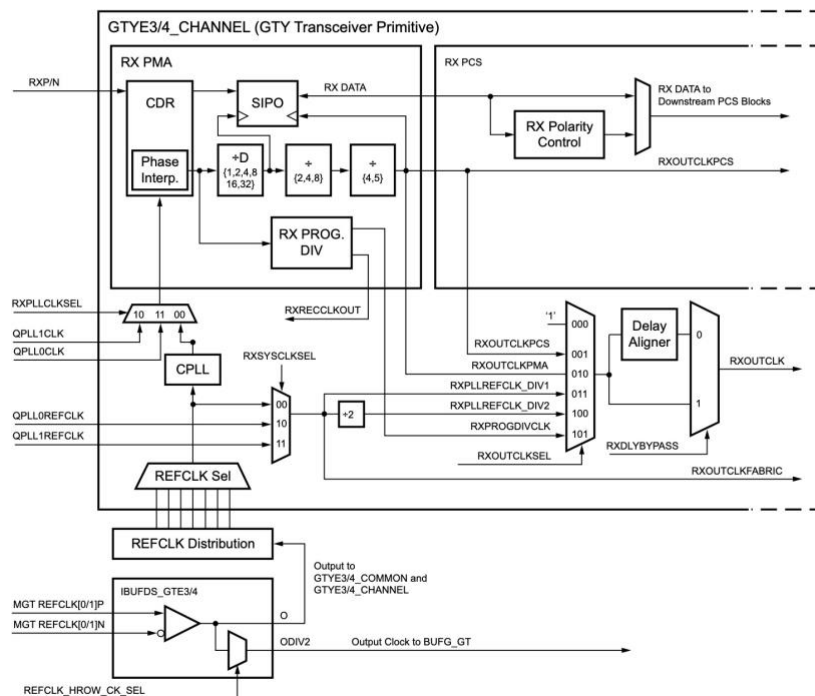


Figure 4-16: RX Serial and Parallel Clock Divider

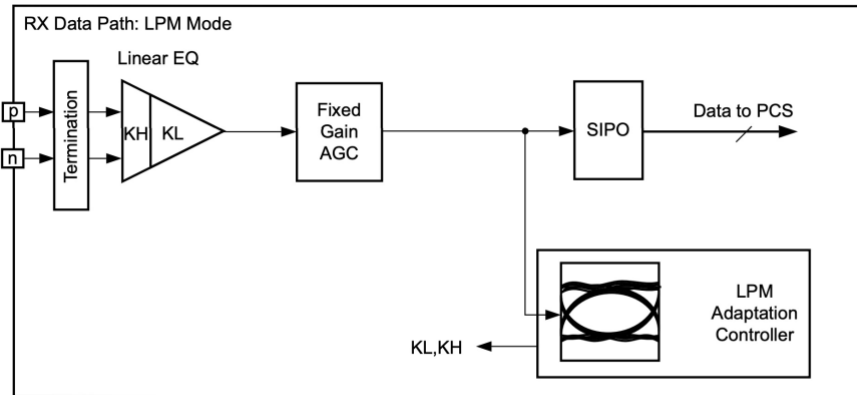
### RX Programmable Divider

The RX programmable divider shown in Figure 4-16 uses the recovered clock from the CDR to generate a parallel output clock. By using the recovered clock, RX programmable divider,

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

(d) converting, by the deserializer, the first serial data stream into a first plurality of parallel data streams;

The Accused Instrumentalities perform “converting, by the deserializer, the first serial data stream into a first plurality of parallel data streams,” e.g.:



X19659-082217

Figure 4-12: LPM Mode

## RX Fabric Clock Output Control

### Functional Description

The RX clock divider control block has two main components: serial clock divider control and parallel clock divider and selector control. The clock divider and selector details are illustrated in Figure 4-16.

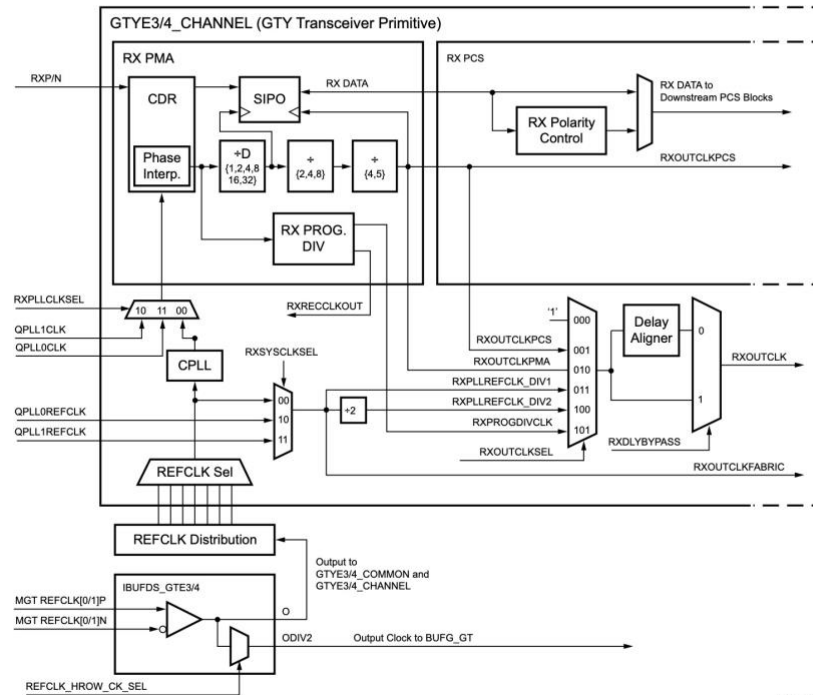


Figure 4-16: RX Serial and Parallel Clock Divider

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

(e) transmitting, from the deserializer to computational circuitry in the field programmable gate array, the first plurality of parallel data streams;

The Accused Instrumentalities perform “transmitting, from the deserializer to computational circuitry in the field programmable gate array, the first plurality of parallel data streams,” e.g.:

## RX Interface

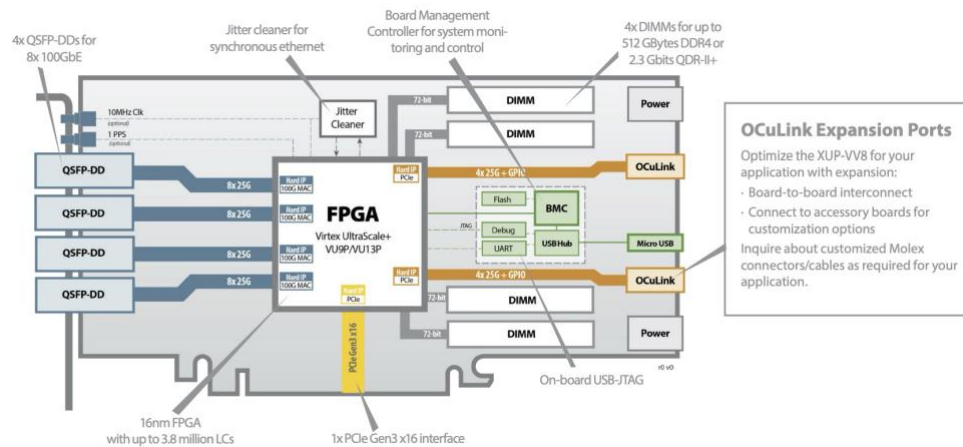
### Functional Description

The RX interface is the gateway to the RX datapath of the GTY transceiver. Applications receive data through the GTY transceiver by receiving data from the RXDATA port on the positive edge of RXUSRCLK2. The width of the port can be configured to be two, four, or

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

(f) transmitting, from the deserializer to a phase lock loop of the field programmable gate array system that is not within the field programmable gate array, the receiver side clock signal;

The Accused Instrumentalities perform “transmitting, from the deserializer to a phase lock loop of the field programmable gate array system that is not within the field programmable gate array, the receiver side clock signal,” e.g.:



([https://www.bittware.com/files/XUP-VV8\\_datasheet\\_r0v0.pdf](https://www.bittware.com/files/XUP-VV8_datasheet_r0v0.pdf))

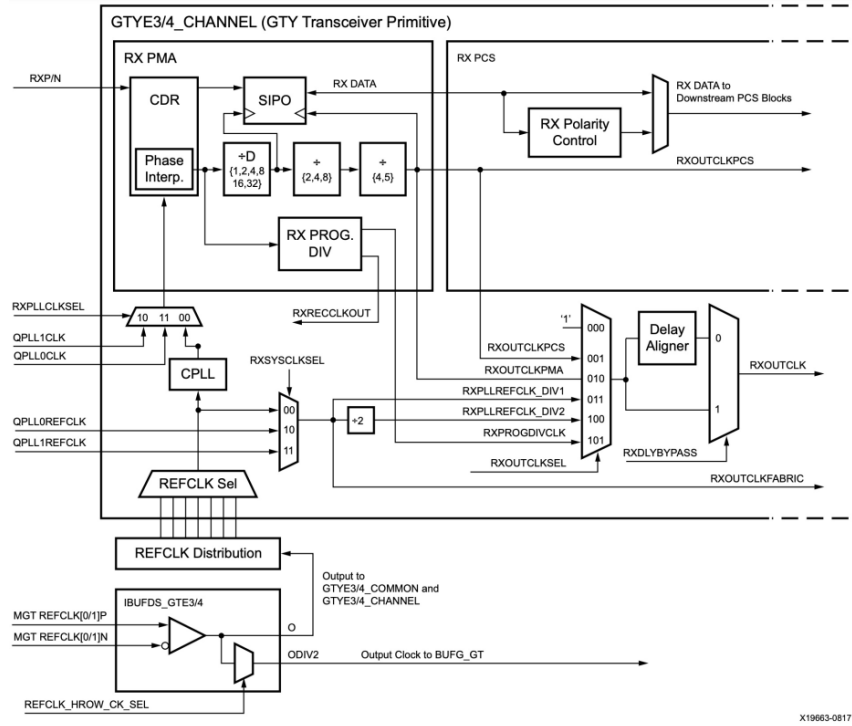


Figure 4-16: RX Serial and Parallel Clock Divider

The output clock of the programmable divider can also be bought out to the transceiver reference clock pin configured as an output. The supported divider values are 4, 5, 8, 10, 16,

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)



“Two jitter cleaners remove unwanted noise from the QSFP-DD clock inputs.”

(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

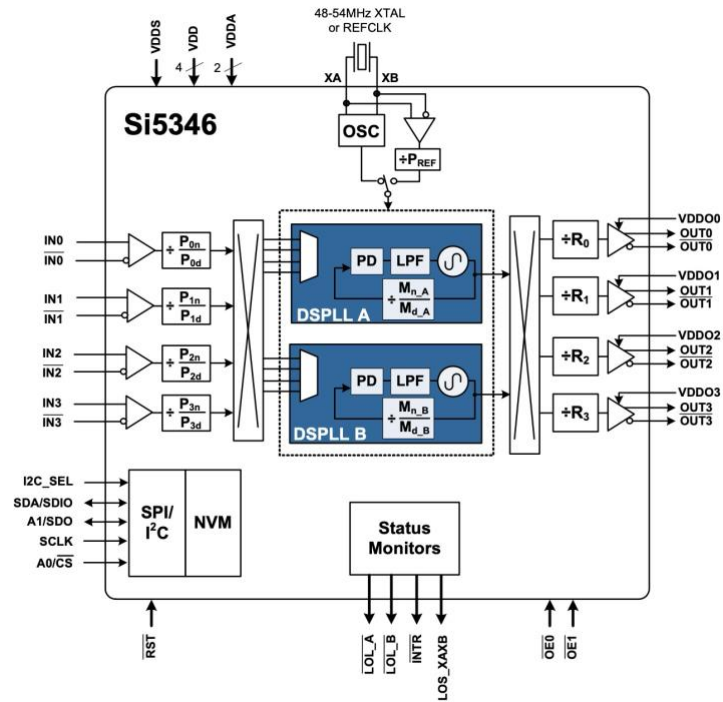


Figure 10. Si5346 Detailed Block Diagram

### 5.6. Inputs (IN0, IN1, IN2, IN3)

There are four inputs that can be used to synchronize any of the DSPLLs. The inputs accept both differential and single-ended clocks. A crosspoint between the inputs and the DSPLLs allows any of the inputs to connect to any of the DSPLLs as shown in Figure 14.

### 5.3.3. Lock Acquisition Mode

Each of the DSPLLs independently monitors its configured inputs for a valid clock. If at least one valid clock is available for synchronization, a DSPLL will automatically start the lock acquisition process.

If the fast lock feature is enabled, a DSPLL will acquire lock using the Fastlock Loop Bandwidth setting and then transition to the DSPLL Loop Bandwidth setting when lock acquisition is complete. During lock acquisition the outputs will generate a clock that follows the VCO frequency change as it pulls-in to the input clock frequency.

(<https://docs.rs-online.com/9a98/0900766b81411ff1.pdf>)

(g) generating, using the phase lock loop, a second clock signal;  
a second clock signal;

The Accused Instrumentalities perform “generating, using the phase lock loop, a second clock signal,”  
e.g.:

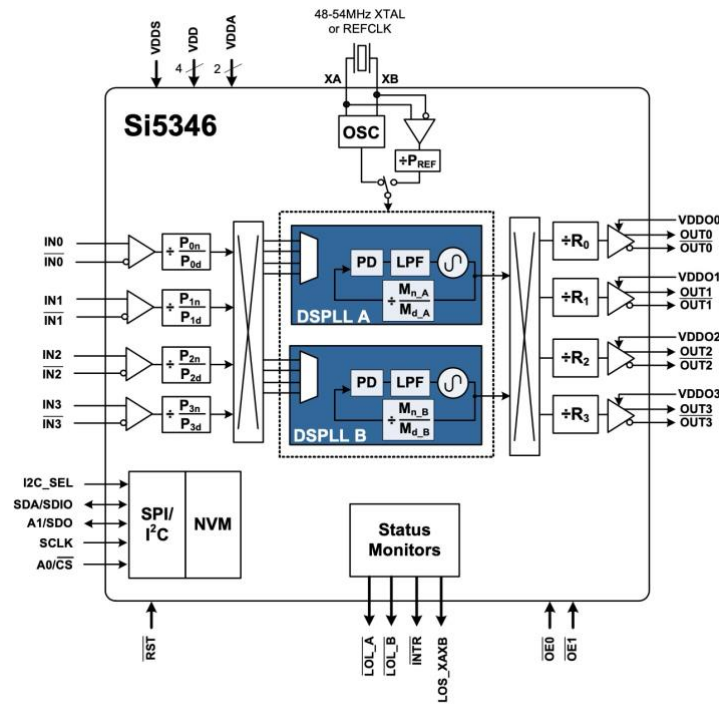


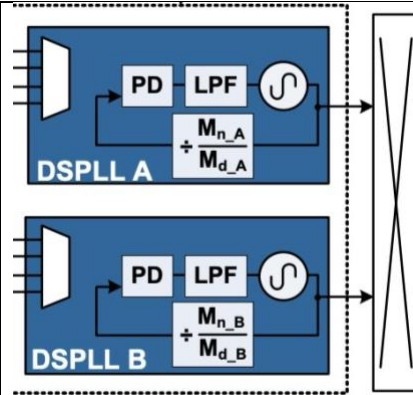
Figure 10. Si5346 Detailed Block Diagram

### Description

The Si5347 is a high performance jitter attenuating clock multiplier which integrates four any-frequency DSPLLs for applications that require maximum integration and independent timing paths. The Si5346 is a dual DSPLL version in a smaller package. Each DSPLL has

### 5.6. Inputs (IN0, IN1, IN2, IN3)

There are four inputs that can be used to synchronize any of the DSPLLs. The inputs accept both differential and single-ended clocks. A crosspoint between the inputs and the DSPLLs allows any of the inputs to connect to any of the DSPLLs as shown in Figure 14.



#### 5.3.4. Locked Mode

Once locked, a DSPLL will generate output clocks that are both frequency and phase locked to their selected input clocks. At this point, any XTAL frequency drift will not affect the output frequency. Each DSPLL has its own  $\overline{\text{LOL}}$  pin and status bit to indicate when lock is achieved. See "5.7.4. LOL Detection" on page 34 for more details on the operation of the loss of lock circuit.

(<https://docs.rs-online.com/9a98/0900766b81411ff1.pdf>)

(h) generating, within the field programmable gate array, a transmitter side clock signal derived from the second clock signal;

The Accused Instrumentalities perform “generating, within the field programmable gate array, a transmitter side clock signal derived from the second clock signal,” e.g.:



“Two jitter cleaners remove unwanted noise from the QSFDD clock inputs.”

(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

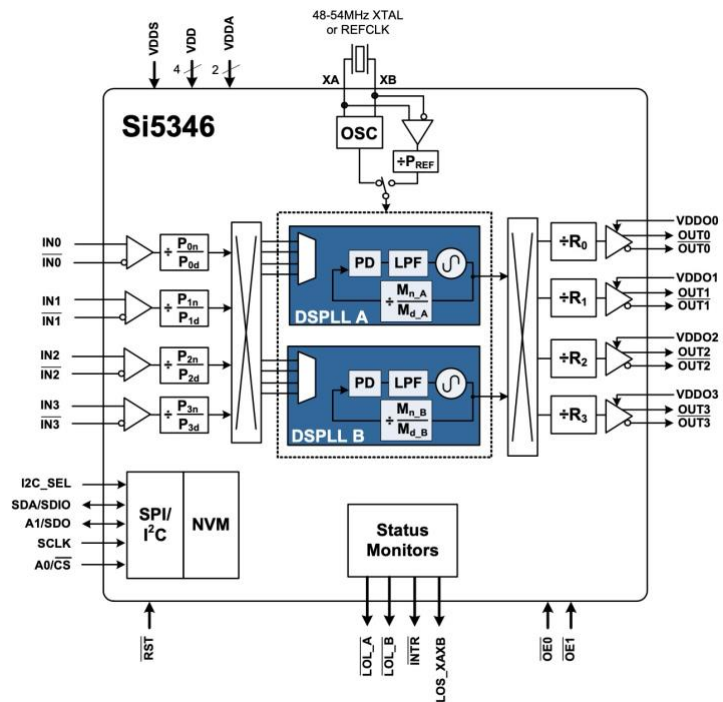


Figure 10. Si5346 Detailed Block Diagram

### 5.8.1. Output Crosspoint

A crosspoint allows any of the output drivers to connect with any of the DSPLLs as shown in Figure 24. The crosspoint configuration is programmable and can be stored in NVM so that the desired output configuration is ready at power-up.

(<https://docs.rs-online.com/9a98/0900766b81411ff1.pdf>)

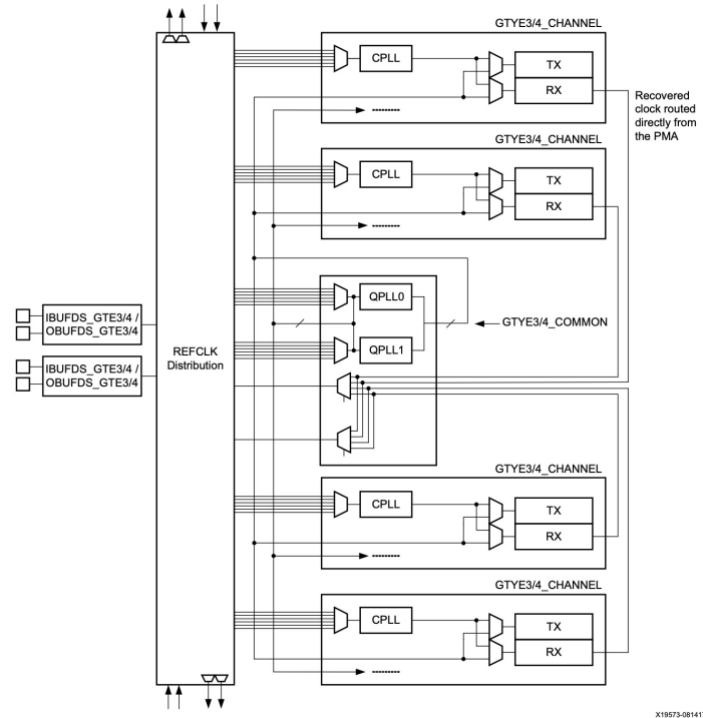


Figure 1-1: GTY Transceiver Quad Configuration

## Channel PLL

### Functional Description

Each GTY transceiver channel contains one ring-based channel PLL (CPLL). The internal channel clocking architecture is shown in [Figure 2-11](#). The TX and RX clock dividers can individually select the clock from the QPLL0/1 or CPLL to allow the TX and RX datapaths to operate at asynchronous frequencies using different reference clock inputs.

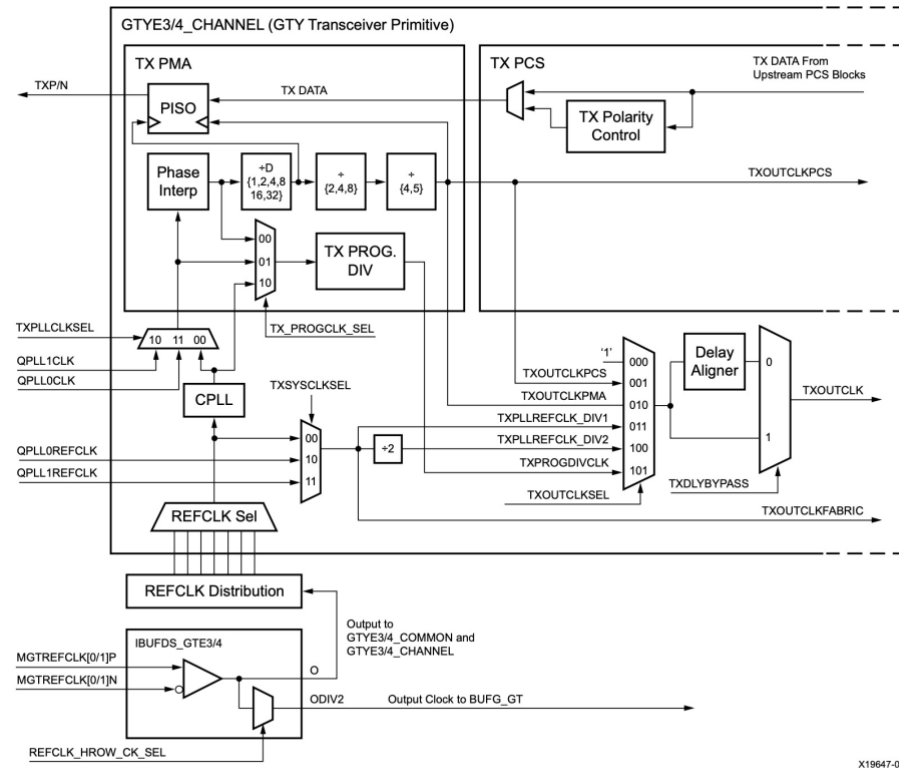


Figure 3-30: TX Serial and Parallel Clock Divider

**Parallel Clock Divider and Selector**

The parallel clock outputs from the TX clock divider control block can be used as an interconnect logic clock, depending on the line rate requirement.

The recommended clock for the interconnect logic is the TXOUTCLK from one of the GTY transceivers. It is also possible to bring the MGTREFCLK directly to the interconnect logic

**TX Programmable Divider**

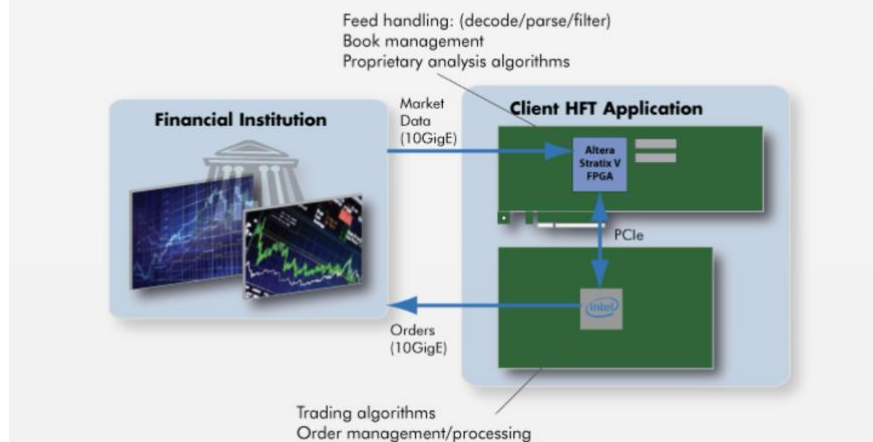
The TX programmable divider shown in Figure 3-30 uses one of the PLL output clocks to generate a parallel output clock. By using the transceiver PLL, TX programmable divider, and BUF6GT, TXOUTCLK (TXOUTCLKSEL = 101) can be used as a clock source for the interconnect logic. The supported divider values are 0.0, 4.0, 5.0, 8.0, 10.0, 16.0, 16.5, 20.0, 32.0, 33.0, 40.0, 64.0, 66.0, 80.0, and 100.0.

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

(i) performing, by the computational circuitry, a set of operations on at least a portion of the first plurality of parallel data streams to generate a second plurality of parallel data streams; and

The Accused Instrumentalities perform “performing, by the computational circuitry, a set of operations on at least a portion of the first plurality of parallel data streams to generate a second plurality of parallel data streams,” e.g.:

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.



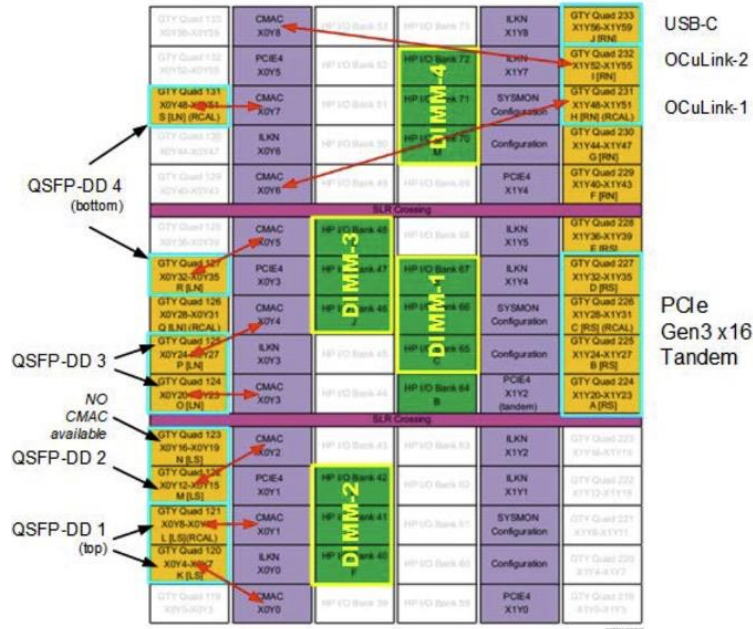
<https://www.bittware.com/about/financial/>

At Citadel Securities, a leading global market maker, our team of FPGA Engineers create next generation FPGA solutions to support ultra-low latency trading systems across the firm. FPGA Engineers work closely with business leaders to design and deploy FPGA solutions that help optimize trading system performance.

<https://www.citadelsecurities.com/careers/details/fpga-engineer/>

(j) transmitting, from the field programmable gate array system, the second serial data stream, derived from the second plurality of parallel data streams, wherein said method does not use clock domain crossing operations that delay processing of the first set of parallel data streams.

The Accused Instrumentalities perform “transmitting, from the field programmable gate array system, the second serial data stream, derived from the second plurality of parallel data streams, wherein said method does not use clock domain crossing operations that delay processing of the first set of parallel data streams,” e.g.:



(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

## TX Overview

### Functional Description

This chapter shows how to configure and use each of the functional blocks inside the transmitter (TX). Each transceiver includes an independent transmitter, which consists of a PCS and a PMA. Figure 3-1 shows the functional blocks of the transmitter. Parallel data flows from the device logic into the TX interface, through the PCS and PMA, and then out the TX driver as high-speed serial data.

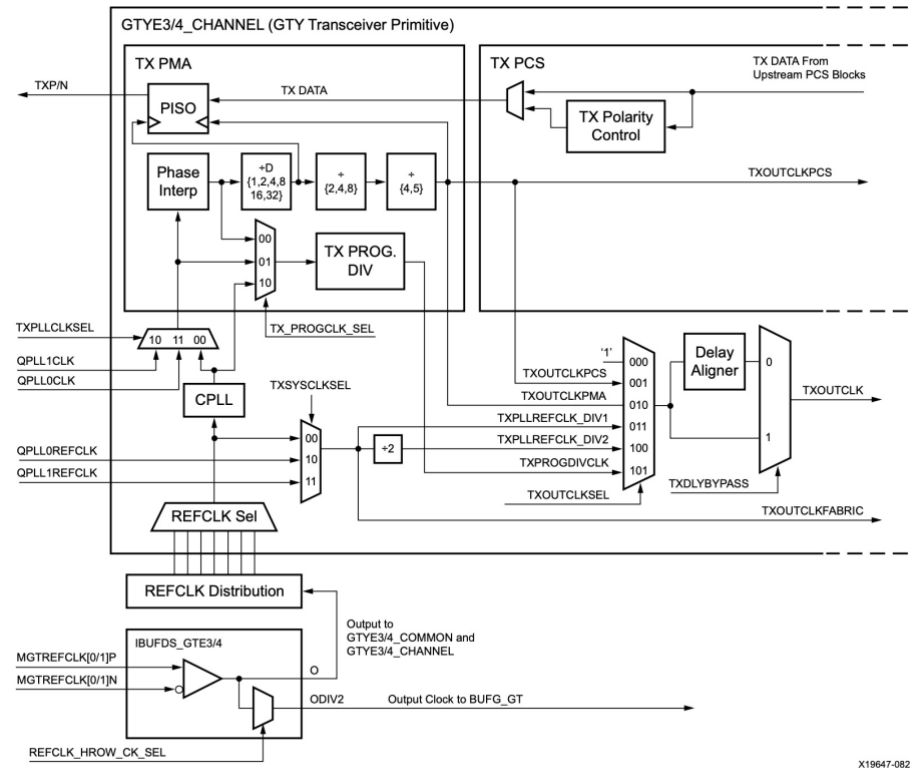
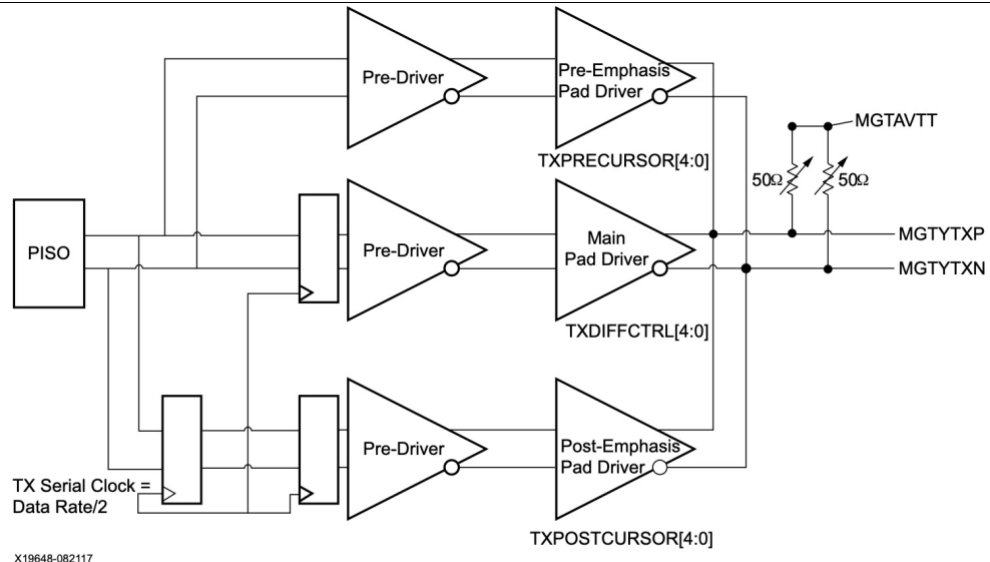


Figure 3-30: TX Serial and Parallel Clock Divider

X19647-082117

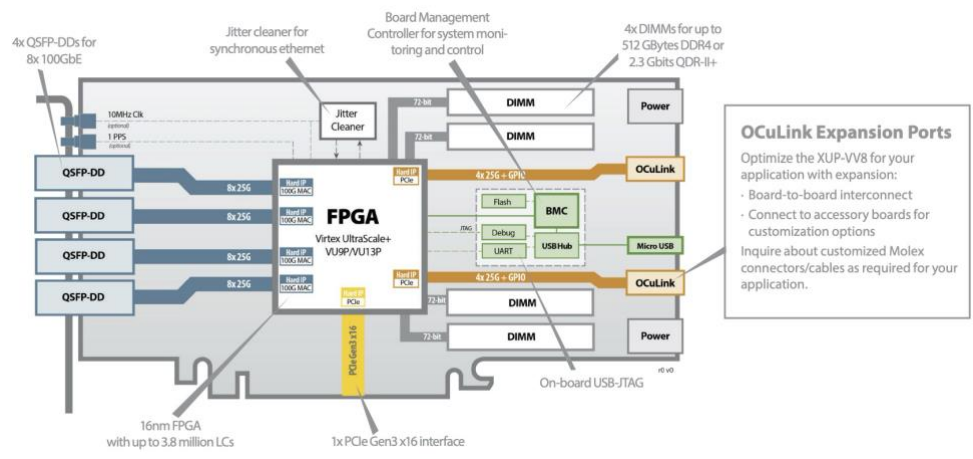


X19648-082117

Figure 3-31: TX Configurable Driver Block Diagram

MGTYTXP MGTYTXN	Out (Pad)	TX Serial Clock	Differential complements of one another forming a differential transmit output pair. These ports represent the pads. The locations of these ports must be constrained (see <a href="#">Implementation, page 20</a> ) and brought to the top level of the design.
--------------------	--------------	--------------------	--

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)



**OCuLink Expansion Ports**  
 Optimize the XUP-VV8 for your application with expansion:  
 - Board-to-board interconnect  
 - Connect to accessory boards for customization options  
 Inquire about customized Molex connectors/cables as required for your application.

([https://www.bittware.com/files/XUP-VV8\\_datasheet\\_r0v0.pdf](https://www.bittware.com/files/XUP-VV8_datasheet_r0v0.pdf))



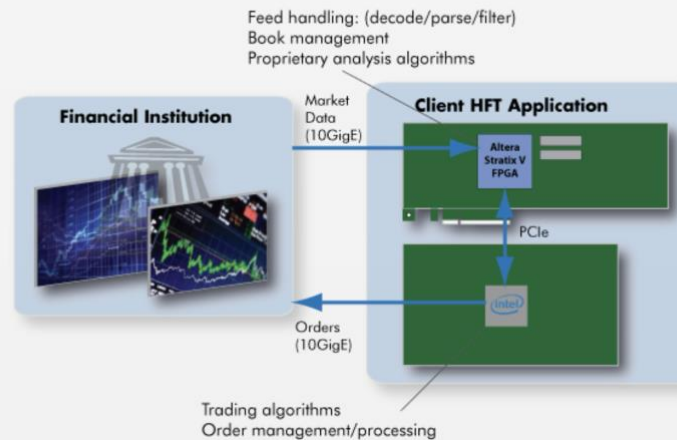
“Two jitter cleaners remove unwanted noise from the QSFP-DD clock inputs.”

(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

2. The method of claim 1, wherein the first serial data stream comprises market data, the second serial data stream comprises order entry data, and the set of operations are associated with a trading algorithm.

The Accused Instrumentalities perform “method of claim 1, wherein the first serial data stream comprises market data, the second serial data stream comprises order entry data, and the set of operations are associated with a trading algorithm,” e.g.:

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.

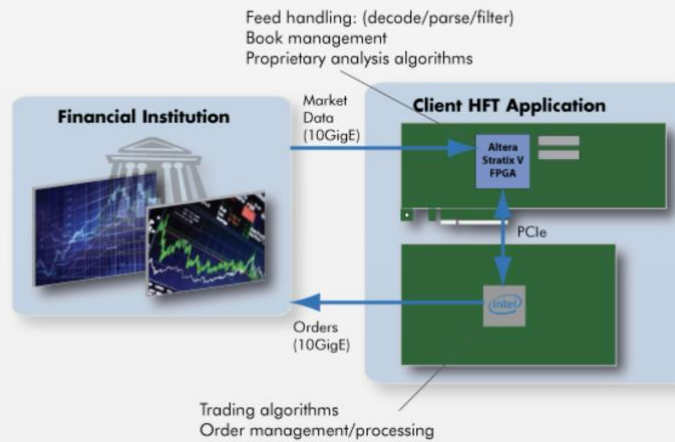


[\(https://www.bittware.com/about/financial/\)](https://www.bittware.com/about/financial/)

3. The method of claim 1, wherein the first serial data stream includes market data and the second serial data stream includes trading data.

The Accused Instrumentalities perform “method of claim 1, wherein the first serial data stream includes market data and the second serial data stream includes trading data,” e.g.:

Two different scenerios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.

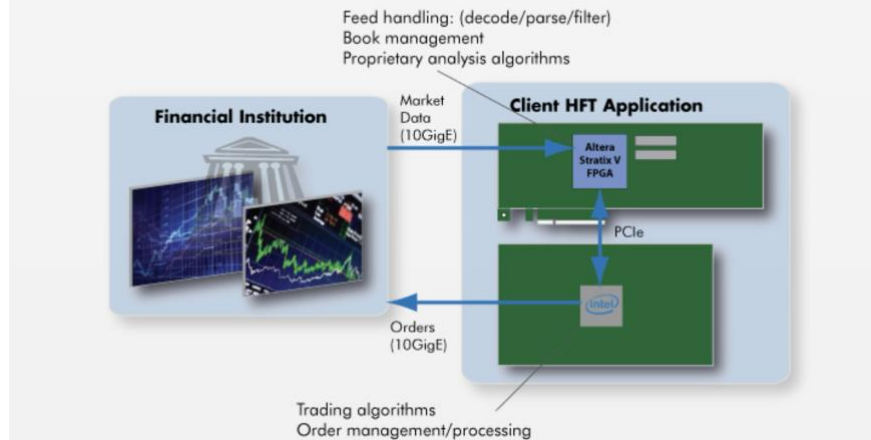


[\(https://www.bittware.com/about/financial/\)](https://www.bittware.com/about/financial/)

4. The method of claim 1, wherein the set of operations includes at least one of the following:  
(i) an arithmetic operation;  
(ii) a logical operation;  
(iii) a pipeline operation; and  
(iv) a memory access operation.

The Accused Instrumentalities perform “method of claim 1, wherein the set of operations includes at least one of the following: (i) an arithmetic operation; (ii) a logical operation; (iii) a pipeline operation; and (iv) a memory access operation,” e.g.:

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.



(<https://www.bittware.com/about/financial/>)

5. The method of claim 1, wherein at least a portion of the set of operations performed in step (h) are performed prior to step (g).

The Accused Instrumentalities perform “method of claim 1, wherein at least a portion of the set of operations performed in step (h) are performed prior to step (g),” e.g.:

Without the benefit of discovery in this case and on information and belief, the Accused Products perform at least one of claim 5, 6, or 7.

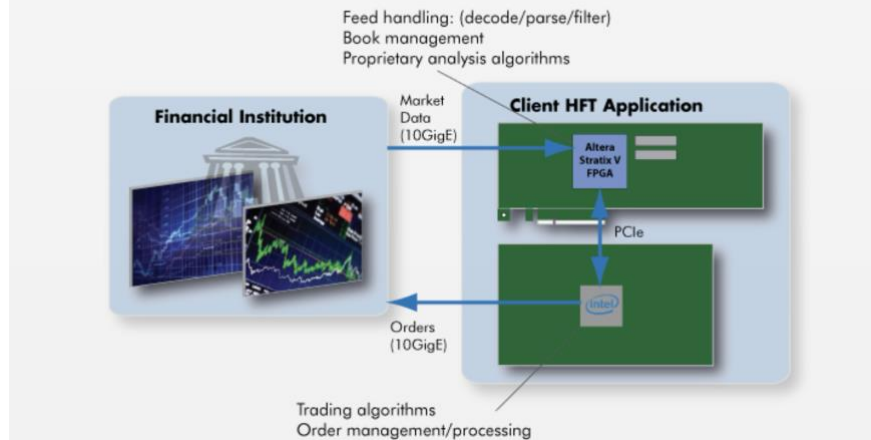
**Parallel Clock Divider and Selector**

The parallel clock outputs from the TX clock divider control block can be used as an interconnect logic clock, depending on the line rate requirement.

The recommended clock for the interconnect logic is the TXOUTCLK from one of the GTY transceivers. It is also possible to bring the MGTREFCLK directly to the interconnect logic

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.



(<https://www.bittware.com/about/financial/>)

6. The method of claim 1, wherein at least a portion of the set of operations performed in step (h) are performed after step (g).

The Accused Instrumentalities perform “method of claim 1, wherein at least a portion of the set of operations performed in step (h) are performed after step (g),” e.g.:

Without the benefit of discovery in this case and on information and belief, the Accused Products perform at least one of claim 5, 6, or 7.

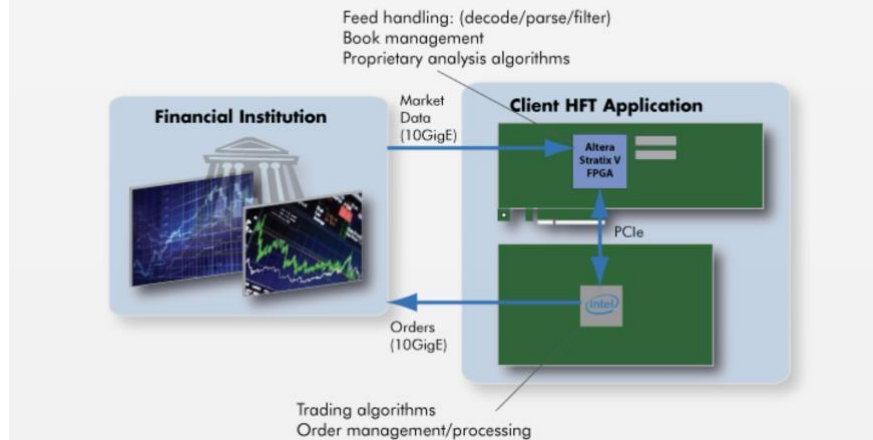
**Parallel Clock Divider and Selector**

The parallel clock outputs from the TX clock divider control block can be used as an interconnect logic clock, depending on the line rate requirement.

The recommended clock for the interconnect logic is the TXOUTCLK from one of the GTY transceivers. It is also possible to bring the MGTREFCLK directly to the interconnect logic

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.



(<https://www.bittware.com/about/financial/>)

7. The method of claim 1, wherein all of the set of operations performed in step (h) are performed after step (g).

The Accused Instrumentalities perform “method of claim 1, wherein all of the set of operations performed in step (h) are performed after step (g),” e.g.:

Without the benefit of discovery in this case and on information and belief, the Accused Products perform at least one of claim 5, 6, or 7.

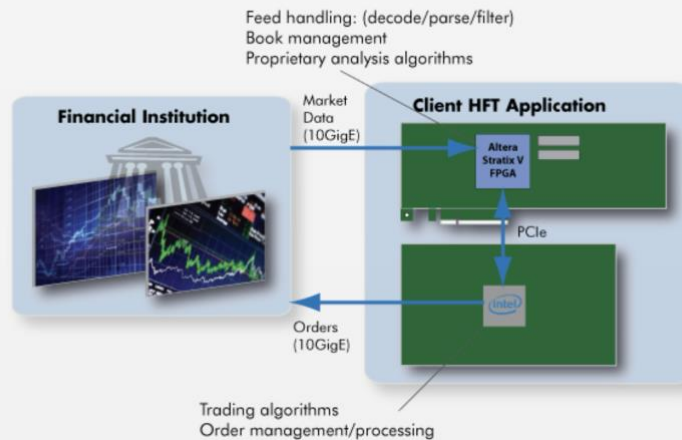
**Parallel Clock Divider and Selector**

The parallel clock outputs from the TX clock divider control block can be used as an interconnect logic clock, depending on the line rate requirement.

The recommended clock for the interconnect logic is the TXOUTCLK from one of the GTY transceivers. It is also possible to bring the MGTREFCLK directly to the interconnect logic

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

Two different scenarios exist for this type of application. In both examples, the FPGA in the customer's high frequency trading application receives market data via 10GigE from the financial institution. The FPGA decodes, parses, and filters the feed; and optionally keeps the books. In the first example, the FPGA then runs trade/risk algorithms; and manages orders. The CPU is used for system management. Orders are sent back to the financial institution via the FPGA over Ethernet.



(<https://www.bittware.com/about/financial/>)

8. The method of claim 1, wherein step (g) further comprises:

(i) generating, by the phase lock loop, a feedback clock signal associated with the transmitter side clock signal by performing the following steps until a first output of a phase detector of the field programmable gate array system is below a first threshold level:

- (1) generating, by an adjustable oscillator in the field programmable gate array system, the second clock signal;
- (2) generating, by the phase detector, the first output based on a comparison of the receiver side clock signal and the feedback clock signal obtained from the second clock signal;

The Accused Instrumentalities perform “method of claim 1, wherein step (g) further comprises: (i) generating, by the phase lock loop, a feedback clock signal associated with the transmitter side clock signal by performing the following steps until a first output of a phase detector of the field programmable gate array system is below a first threshold level: (1) generating, by an adjustable oscillator in the field programmable gate array system, the second clock signal; (2) generating, by the phase detector, the first output based on a comparison of the receiver side clock signal and the feedback clock signal obtained from the second clock signal; (3) transmitting, from the phase detector to a phase controller of the field programmable gate array system, the first output; (4) determining, by the phase controller, interim adjustment information based on the first output; and (5) transmitting, from the phase controller to the adjustable oscillator, the interim adjustment information; wherein, the adjustable oscillator adjusts the second clock signal based on the interim adjustment information, wherein steps (1) through (5) are repeated until the first output of the phase detector is below the first threshold level,” e.g.:

(3) transmitting, from the phase detector to a phase controller of the field programmable gate array system, the first output;

(4) determining, by the phase controller, interim adjustment information based on the first output; and

(5) transmitting, from the phase controller to the adjustable oscillator, the interim adjustment information;

wherein, the adjustable oscillator adjusts the second clock signal based on the interim adjustment information, wherein steps (1) through (5) are repeated until the first output of the phase detector is below the first threshold level.

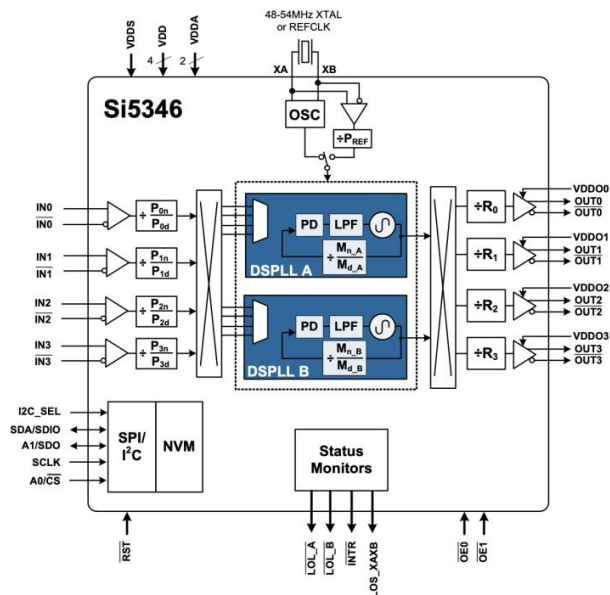


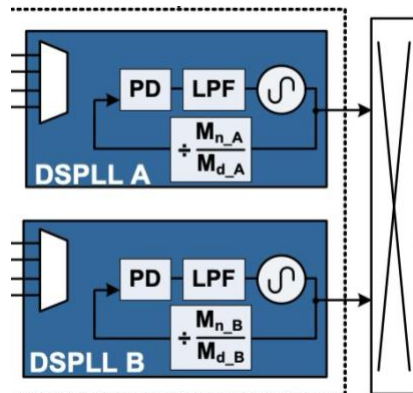
Figure 10. Si5346 Detailed Block Diagram

### Description

The Si5347 is a high performance jitter attenuating clock multiplier which integrates four any-frequency DSPLLs for applications that require maximum integration and independent timing paths. The Si5346 is a dual DSPLL version in a smaller package. Each DSPLL has

### 5.6. Inputs (IN0, IN1, IN2, IN3)

There are four inputs that can be used to synchronize any of the DSPLLs. The inputs accept both differential and single-ended clocks. A crosspoint between the inputs and the DSPLLs allows any of the inputs to connect to any of the DSPLLs as shown in Figure 14.



### 5.3.3. Lock Acquisition Mode

Each of the DSPLLs independently monitors its configured inputs for a valid clock. If at least one valid clock is available for synchronization, a DSPLL will automatically start the lock acquisition process.

If the fast lock feature is enabled, a DSPLL will acquire lock using the Fastlock Loop Bandwidth setting and then transition to the DSPLL Loop Bandwidth setting when lock acquisition is complete. During lock acquisition the outputs will generate a clock that follows the VCO frequency change as it pulls-in to the input clock frequency.

### 5.3.4. Locked Mode

Once locked, a DSPLL will generate output clocks that are both frequency and phase locked to their selected input clocks. At this point, any XTAL frequency drift will not affect the output frequency. Each DSPLL has its own  $\overline{\text{LOL}}$  pin and status bit to indicate when lock is achieved. See "5.7.4. LOL Detection" on page 34 for more details on the operation of the loss of lock circuit.

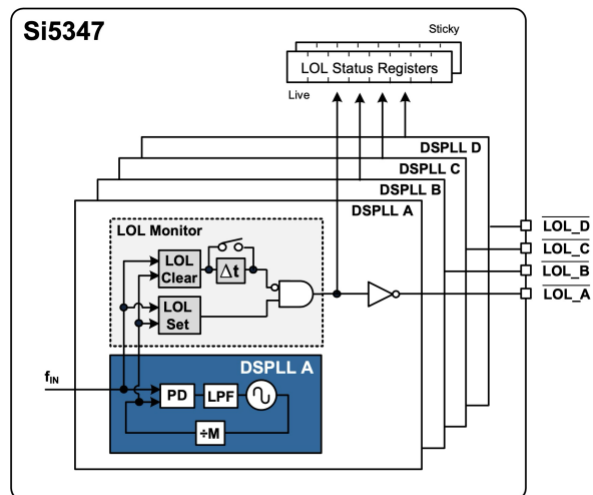


Figure 21. LOL Status Indicators

### 5.7.4. LOL Detection

There is an LOL monitor for each of the DSPLLs. The LOL monitor asserts an LOL register bit when a DSPLL has lost synchronization with its selected input clock. There is also a dedicated loss of lock pin that reflects the loss of lock condition for each of the DSPLLs (LOL\_A, LOL\_B, LOL\_C, LOL\_D). The LOL monitor functions by measuring the frequency difference between the input and feedback clocks at the phase detector. There are two LOL

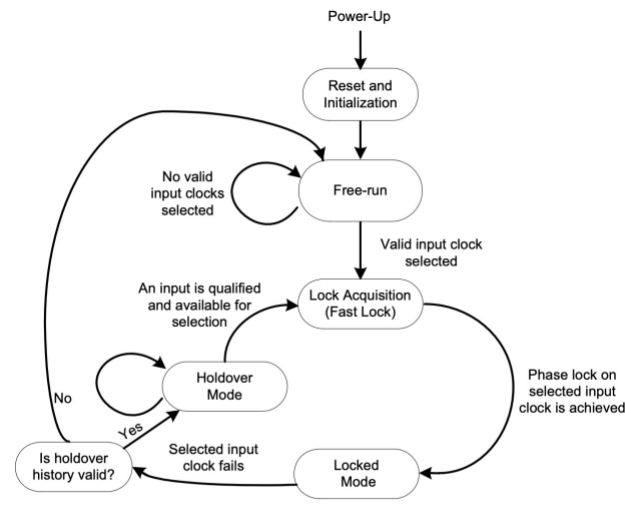


Figure 11. Modes of Operation

<https://docs.rs-online.com/9a98/0900766b81411ff1.pdf>

9. The method of claim 1, wherein the first plurality of data streams has the same number of data streams as the second plurality of data streams.

The Accused Instrumentalities perform “method of claim 1, wherein the first plurality of data streams has the same number of data streams as the second plurality of data streams,” e.g.:

**Interface Width Configuration**

The GTY transceiver contains 2-byte, 4-byte, and 8-byte internal datapaths and is configurable by setting the TX\_INT\_DATAWIDTH attribute. The interface width is configurable by setting the TX\_DATA\_WIDTH attribute. When the 8B/10B encoder is enabled, the TX\_DATA\_WIDTH attribute must be configured to 20 bits, 40 bits, or 80 bits, and in this case, the TX interface only uses the TXDATA ports. For example, TXDATA[15:0] is used when the interface width is 16. When the 8B/10B encoder is bypassed, the TX\_DATA\_WIDTH attribute can be configured to any of the available widths: 16, 20, 32, 40, 64, 80, 128, or 160 bits.

Table 3-1 shows how the interface width for the TX datapath is selected. 8B/10B encoding is described in more detail in [TX 8B/10B Encoder, page 112](#).

Table 3-1: TX Interface Datapath Configuration

TX8B10BEN	TX_DATA_WIDTH	TX_INT_DATAWIDTH	Interface Width	Internal Data Width
1	20	0	16	20
	40	0	32	20
	40	1	32	40
	80	1	64	40
0	16	0	16	16
	20	0	20	20
	32	0	32	16
	32	1	32	32
	40	0	40	20
	40	1	40	40
	64	1	64	32
	64	2	64	64
	80	1	80	40
	80	2	80	80
	128	2	128	64
	160	2	160	80

When the 8B/10B encoder is bypassed and the TX\_DATA\_WIDTH is 20, 40, 80, or 160, the TXCTRL1 and TXCTRL0 ports are used to extend the TXDATA port from 16 to 20 bits, 32 to 40 bits, 64 to 80 bits, or 128 to 160 bits. Table 3-2 shows the data transmitted when the 8B/10B encoder is disabled. When the TX\_DATA\_WIDTH is 16, 32, or 64, the TXCTRL1/0 ports are ignored and the data transmission follows the same order as Table 3-2, but without the TXCTRL1/0 bits. When the TX gearbox is used, refer to [TX Synchronous Gearbox, page 116](#) for data transmission order.

<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

10. The method of claim 1, wherein the first plurality of data streams and the second plurality of data streams comprise one of the following:  
 (i) eight (8) data streams;  
 (ii) ten (10) data streams;  
 (iii) sixteen (16) data streams;  
 (iv) twenty (20) data streams;  
 (v) thirty-two (32) data streams;  
 (vi) forty (40) data streams;  
 (vii) sixty-four (64) data streams;  
 (viii) eighty (80) data streams;

The Accused Instrumentalities perform “method of claim 1, wherein the first plurality of data streams and the second plurality of data streams comprise one of the following: (i) eight (8) data streams; (ii) ten (10) data streams; (iii) sixteen (16) data streams; (iv) twenty (20) data streams; (v) thirty-two (32) data streams; (vi) forty (40) data streams; (vii) sixty-four (64) data streams; (viii) eighty (80) data streams; (ix) one hundred twenty-eight (128) data streams; and (x) one hundred sixty (160) data streams,” e.g.:

(ix) one hundred twenty-eight (128) data streams; and  
 (x) one hundred sixty (160) data streams.

**Interface Width Configuration**

The GTY transceiver contains 2-byte, 4-byte, and 8-byte internal datapaths and is configurable by setting the TX\_INT\_DATAWIDTH attribute. The interface width is configurable by setting the TX\_DATA\_WIDTH attribute. When the 8B/10B encoder is enabled, the TX\_DATA\_WIDTH attribute must be configured to 20 bits, 40 bits, or 80 bits, and in this case, the TX interface only uses the TXDATA ports. For example, TXDATA[15:0] is used when the interface width is 16. When the 8B/10B encoder is bypassed, the TX\_DATA\_WIDTH attribute can be configured to any of the available widths: 16, 20, 32, 40, 64, 80, 128, or 160 bits.

Table 3-1 shows how the interface width for the TX datapath is selected. 8B/10B encoding is described in more detail in [TX 8B/10B Encoder, page 112](#).

Table 3-1: TX Interface Datapath Configuration

TX8B10BEN	TX_DATA_WIDTH	TX_INT_DATAWIDTH	Interface Width	Internal Data Width
1	20	0	16	20
	40	0	32	20
	40	1	32	40
	80	1	64	40
0	16	0	16	16
	20	0	20	20
	32	0	32	16
	32	1	32	32
	40	0	40	20
	40	1	40	40
	64	1	64	32
	64	2	64	64
	80	1	80	40
	80	2	80	80
	128	2	128	64
	160	2	160	80

When the 8B/10B encoder is bypassed and the TX\_DATA\_WIDTH is 20, 40, 80, or 160, the TXCTRL1 and TXCTRL0 ports are used to extend the TXDATA port from 16 to 20 bits, 32 to 40 bits, 64 to 80 bits, or 128 to 160 bits. Table 3-2 shows the data transmitted when the 8B/10B encoder is disabled. When the TX\_DATA\_WIDTH is 16, 32, or 64, the TXCTRL1/0 ports are ignored and the data transmission follows the same order as Table 3-2, but without the TXCTRL1/0 bits. When the TX gearbox is used, refer to [TX Synchronous Gearbox, page 116](#) for data transmission order.

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

11. The method of claim 1, wherein the transmitter side clock signal and the receiver side clock signal have the same frequency and phase.

The Accused Instrumentalities perform “method of claim 1, wherein the transmitter side clock signal and the receiver side clock signal have the same frequency and phase,” e.g.:

Without the benefit of discovery in this case and on information and belief, the Accused Products perform at least one of claims 11 or 12.



“Two jitter cleaners remove unwanted noise from the QSFP-DD clock inputs.”

(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

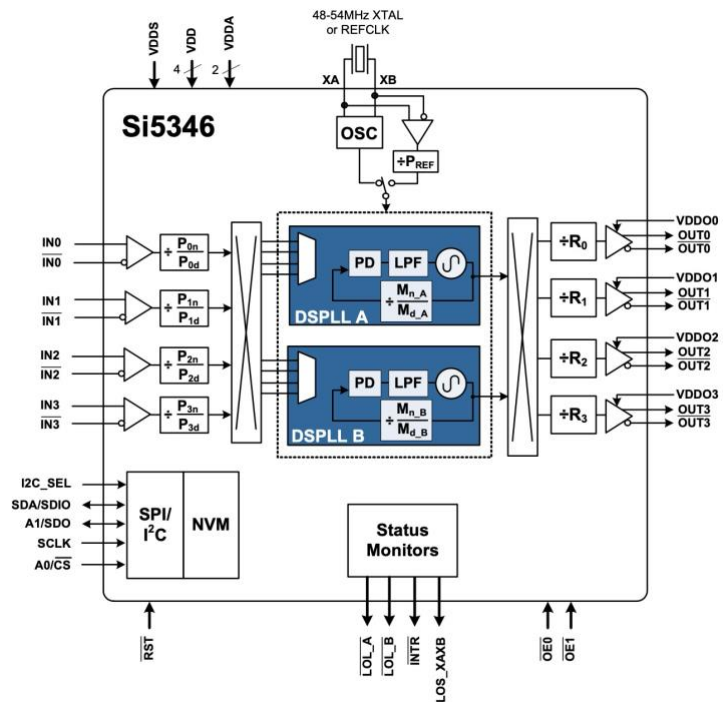


Figure 10. Si5346 Detailed Block Diagram

### 5.8.1. Output Crosspoint

A crosspoint allows any of the output drivers to connect with any of the DSPLLs as shown in Figure 24. The crosspoint configuration is programmable and can be stored in NVM so that the desired output configuration is ready at power-up.

(<https://docs.rs-online.com/9a98/0900766b81411ff1.pdf>)

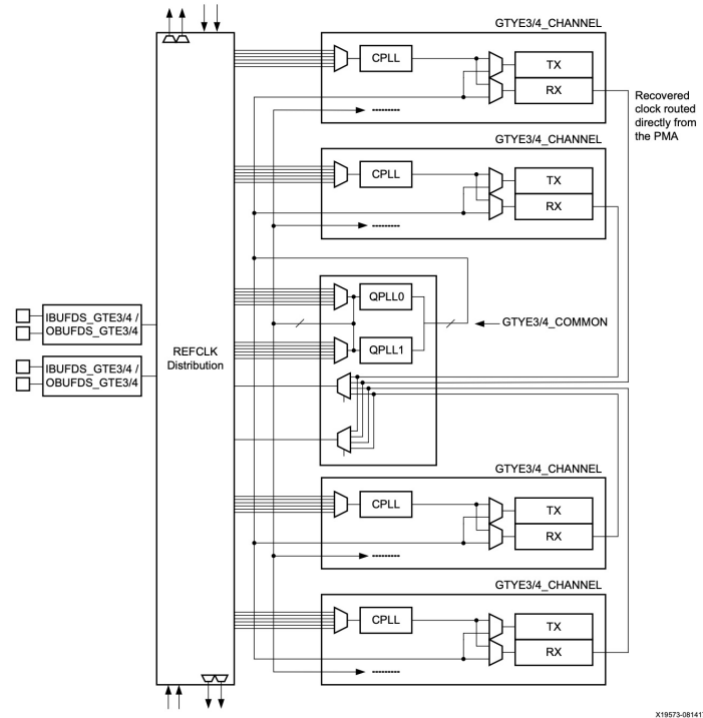


Figure 1-1: GTY Transceiver Quad Configuration

X19573-081417

## Channel PLL

### Functional Description

Each GTY transceiver channel contains one ring-based channel PLL (CPLL). The internal channel clocking architecture is shown in [Figure 2-11](#). The TX and RX clock dividers can individually select the clock from the QPLL0/1 or CPLL to allow the TX and RX datapaths to operate at asynchronous frequencies using different reference clock inputs.

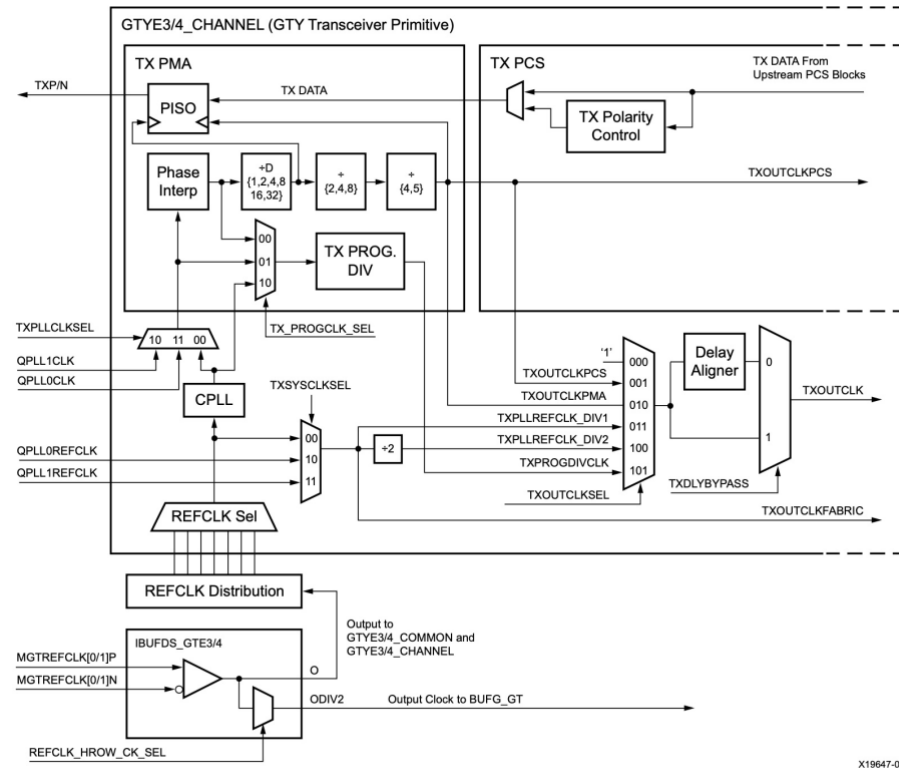


Figure 3-30: TX Serial and Parallel Clock Divider

### Parallel Clock Divider and Selector

The parallel clock outputs from the TX clock divider control block can be used as an interconnect logic clock, depending on the line rate requirement.

The recommended clock for the interconnect logic is the TXOUTCLK from one of the GTY transceivers. It is also possible to bring the MGTREFCLK directly to the interconnect logic

### TX Programmable Divider

The TX programmable divider shown in Figure 3-30 uses one of the PLL output clocks to generate a parallel output clock. By using the transceiver PLL, TX programmable divider, and BUFG\_GT, TXOUTCLK (TXOUTCLKSEL = 101) can be used as a clock source for the interconnect logic. The supported divider values are 0.0, 4.0, 5.0, 8.0, 10.0, 16.0, 16.5, 20.0, 32.0, 33.0, 40.0, 64.0, 66.0, 80.0, and 100.0.

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)





“Two jitter cleaners remove unwanted noise from the QSFP-DD clock inputs.”

(Hardware Reference Guide, available through <https://www.bittware.com/products/xup-vv8/>)

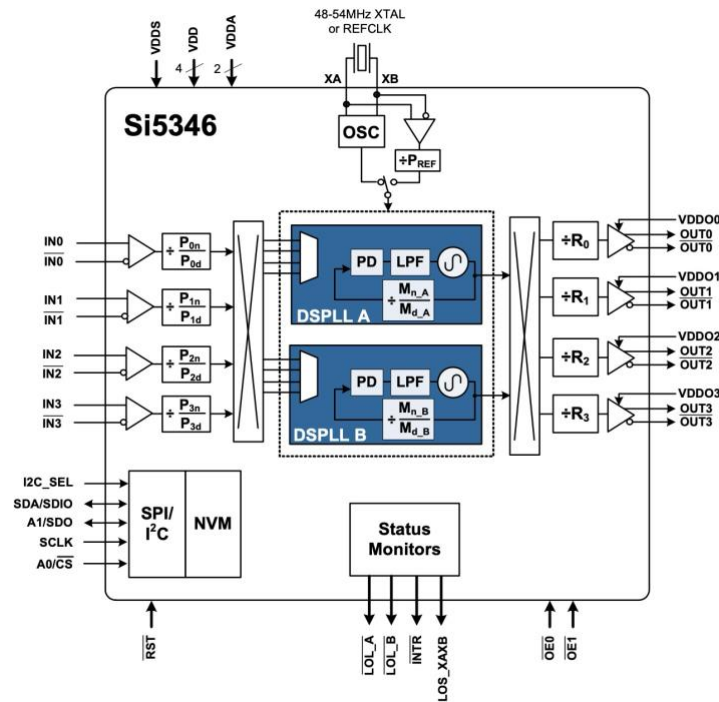


Figure 10. Si5346 Detailed Block Diagram

### 5.8.1. Output Crosspoint

A crosspoint allows any of the output drivers to connect with any of the DSPLLs as shown in Figure 24. The crosspoint configuration is programmable and can be stored in NVM so that the desired output configuration is ready at power-up.

(<https://docs.rs-online.com/9a98/0900766b81411ff1.pdf>)

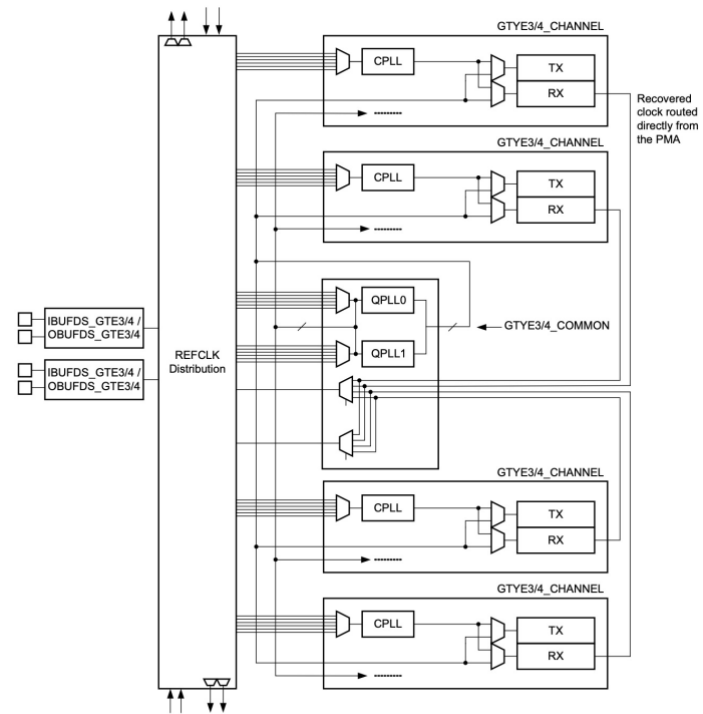


Figure 1-1: GTY Transceiver Quad Configuration

## Channel PLL

### Functional Description

Each GTY transceiver channel contains one ring-based channel PLL (CPLL). The internal channel clocking architecture is shown in Figure 2-11. The TX and RX clock dividers can individually select the clock from the QPLL0/1 or CPLL to allow the TX and RX datapaths to operate at asynchronous frequencies using different reference clock inputs.

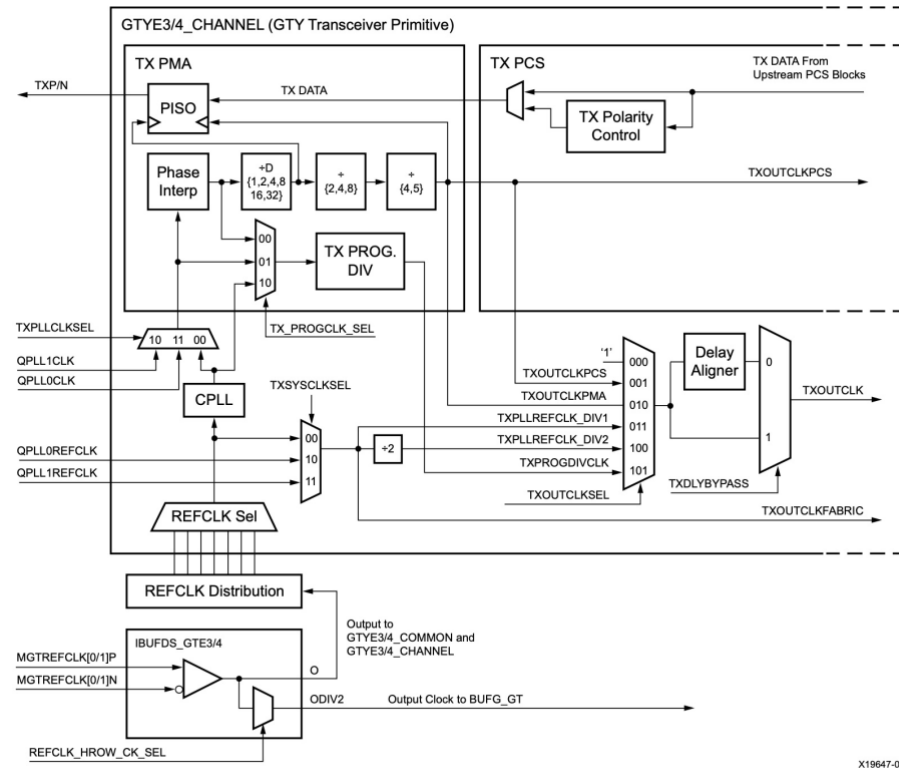


Figure 3-30: TX Serial and Parallel Clock Divider

### Parallel Clock Divider and Selector

The parallel clock outputs from the TX clock divider control block can be used as an interconnect logic clock, depending on the line rate requirement.

The recommended clock for the interconnect logic is the TXOUTCLK from one of the GTY transceivers. It is also possible to bring the MGTREFCLK directly to the interconnect logic

### TX Programmable Divider

The TX programmable divider shown in Figure 3-30 uses one of the PLL output clocks to generate a parallel output clock. By using the transceiver PLL, TX programmable divider, and BUFG\_GT, TXOUTCLK (TXOUTCLKSEL = 101) can be used as a clock source for the interconnect logic. The supported divider values are 0.0, 4.0, 5.0, 8.0, 10.0, 16.0, 16.5, 20.0, 32.0, 33.0, 40.0, 64.0, 66.0, 80.0, and 100.0.

(<https://docs.amd.com/v/u/en-US/ug578-ultrascale-gty-transceivers>)

