*User's Guide*

# *AFE77xxD JESD204C Usage and Applications*

*Kang Hsia*

**ABSTRACT**

The AFE77xxD (AFE7728D/AFE7768D/AFE7769D) family of RF sampling transceiver devices supports JESD204C link Subclass 1. The device has both JESD204C TX block on the ADC cores and also JESD204C RX block on the DAC cores. To facilitate users with the device operation and the JESD204C link bring-up process, the device includes various test modes and error reporting structures to ensure each layer within the JESD204C stage are functional.

## Table of Contents

**1 Introduction**....2
1.1 Terminology....2
**2 AFE77xxD Alarm Structure and Logics**....3
2.1 General Overview of the DAC JESD204 Cores....3
2.2 General Overview of the Alarm Structures....3
2.3 Alarm Propagation Path....3
2.4 Alarm Logic....4
2.5 Typical Alarm Applications....4
**3 Physical Layer**....6
3.1 Physical Layer Error....6
3.2 Physical Layer Testing....8
3.3 Common Questions on SerDes Optimization....11
3.4 SerDes Block Register Access....12
**4 Link Layer**....14
4.1 Link Status....14
4.2 Link and Lane Errors....18
4.3 Lane and Link Aggregation....27
4.4 Link Layer Testing....27
4.5 RBD Optimization....29
**5 Transport Layer**....34
5.1 Transport Layer Data Packing....34
5.2 Transport Layer Testing....41
**6 Summary**....45
**7 References**....46
**8 Summary of C Functions**....47

## Trademarks

All trademarks are the property of their respective owners.

# 1 Introduction

The overall signal chain of the JESD204C link for both the ADC and DAC are highlighted in Figure 1-1. The physical layer, link layer, and transport layer functionality and capabilities are highlighted in this JESD204C overview. Completing these tests during customer end-product characterization can ensure robustness of the AFE77xxD JESD204C link.

Throughout this document, the JESD204C reference is based on released JESD204C standard on December 2017 by JEDEC Standards Committee. The JESD204C standard includes the backwards compatible 8B/10B encoding standard to the JESD204B, and also the newly defined standard of 64B/66B and 64B/80B encoding.

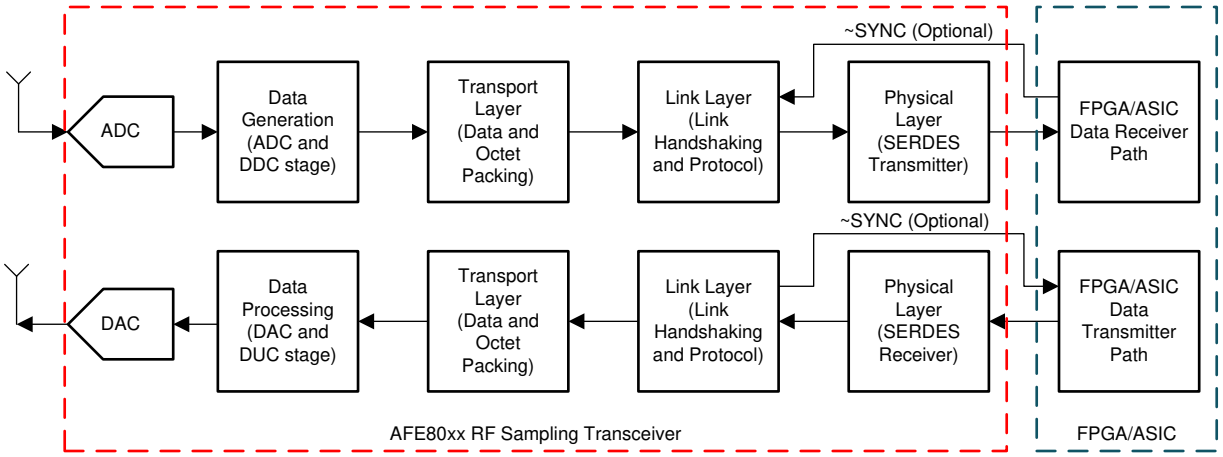


**Figure 1-1. Overall JESD204C Link Diagram for AFE77xxD and FPGA/ASIC**

## 1.1 Terminology

| | |
|---|---|
| **DAC** | Digital-to-Analog Converter (Transmitter Downlink) |
| **TX** | Transmitter Downlink |
| **ADC** | Analog-to-Digital Converter (Receiver Uplink) |
| **RX** | Receiver Uplink |
| **JESD204** | JEDEC Standard for Serial Link Transfer for Data Converters |
| **JESD204 RX block** | JESD204 receiver block to receive JESD204 data |
| **JESD204 TX block** | JESD204 transmitter block to transmit JESD204 data |
| **SerDes** | Serializer and De-serializer Circuits |
| **CDR** | Clock Data Recovery within the SerDes Receivers |
| **STX** | SerDes Transmitter ranges from STX1 to STX8 for AFE77xxD |
| **SRX** | SerDex Receiver ranges from SRX1 to SRX8 for AFE77xxD |
| **BER** | Bit Error Rate |
| **PRBS** | Pseudo Random Bit Sequence |
| **/K/, K28.5, Comma Characters** | For 8B/10B encoding, this is the alignment character in 0xBCBC format for code group synchronization. |
| **/R/, K28.0** | For 8B/10B coding. This is the character indicating the end of code group synchronization. |
| **CGS** | Code group synchronization |
| **Frame (F)** | a set of consecutive octets in which the position of each octet can be identified by reference to a frame alignment signal. |
| **Multi-frame (K)** | a set of consecutive frames in which the position of each frame can be identified by reference to a multiframe alignment signal. |

| | |
|---|---|
| **GPIO** | General Purpose Input and or Output |
| **DC PLL** | on-chip phase locked loop for the data converters |
| **SerDes PLL** | on-chip phase locked loop for the SerDes |
| **Logic Device** | FPGA (field programmable gate array) or ASIC (applications specific integrated circuits) |
| **LMFC** | Local Multiframe Clock (8B/10B Encoding) |
| **LEMC** | Local Extended Multiblock Clock (64B/66B and 64B/80B Encoding) |

# 2 AFE77xxD Alarm Structure and Logics

## 2.1 General Overview of the DAC JESD204 Cores

- With respect to the JESD204C link, the error reporting resides on the JESD204 RX block. With respect to the AFE device, the error reporting resides within the DAC JESD204 core.
- There are four independent JESD204 lanes in the JESD204 core.

## 2.2 General Overview of the Alarm Structures

- Alarms: error reporting structure in the AFE JESD DAC section to report any warnings or potential problems in the AFE JESD204 RX Link (for example, AFE Downlink).
- The alarms are located within the DAC JESD pages for the two DAC JESD core.
- The alarms can be categorized into two different set of main properties: propagation path and logic.

**Table 2-1. AFE77xxD JESD204 Alarm Structures and Logics**

| Alarm Names | Alarm Propagation Path | Alarm Logics | Register Address |
|---|---|---|---|
| Common Alarm | Pin | Clear | 0x1EC |
| | | Mask | 0x1ED |
| | | Alarm | 0x210 |
| | PAP | Clear | 0x1EE |
| | | Mask | 0x1EF |
| | | Alarm | 0x212 |
| laneX Alarm<br>X = [0:3] for each DAC JESD Core, with total of eight lanes | Pin | Clear | 0x1F0 to 0x1F7 |
| | | Mask | 0x1F8 to 0x1FF |
| | | Alarm | 0x214 to 0x21B |
| laneX Alarm<br>X = [0:3] for each DAC JESD Core, with total of eight lanes | PAP | Clear | 0x200 to 0x207 |
| | | Mask | 0x208 to 0x20F |
| | | Alarm | 0x21C to 0x223 |
| Master Mask to Pin | Pin | Mask | 0x224, bit[1] |
| Master Clear to Pin | | Clear | 0x224, bit[0] |
| Master Mask to PAP | PAP | Mask | 0x224, bit[3] |
| Master Clear to PAP | | Clear | 0x224, bit[2] |

## 2.3 Alarm Propagation Path

### 2.3.1 Alarm-to-Pin

- These are the alarms to trigger the general ALARM GPIO to logic HI or logic LOW.
- The ALARM GPIO can interrupt the host processor to perform additional steps within the host processor to mitigate the issue and remedy the situation.

### 2.3.2 Alarm-to-PAP

- These are the alarms to trigger the AFE's internal autonomous power amplifier protection (PAP) logic.
- Upon the preferred alarms to be routed to the PAP logic (under the TXDIG section), the PAP logic can ramp up or ramp down the downlink signal according to the error condition to protect the power amplifier

Logically speaking both Alarm-to-Pin and Alarm-to-PAP have the functionality, with the exception of final propagation of the alarm signal. The user can choose which alarms gets propagated through the masking feature, which is described in the next main property.

## 2.4 Alarm Logic

The registers for the alarms can be categorized into the next property of status: mask, clear, master mask, and master clear.

### 2.4.1 Alarm Status

- These reflect the actual alarm status.
- Sticky: alarms in the AFE are default programmed as *sticky*. This means the alarms do *not* self-clear upon circuit/logic recovery. This allows the user to poll the non-critical errors regularly and detect any errors in between the polls.
- Sticky alarms need to be cleared before updating the latest result.
- Sticky nature of the alarm design is to allow the user to be able to read any errors occurred in the past, even if the errors have been self-fixed in the circuit. The user simply read the alarms first without clearing the alarms.

### 2.4.2 Alarm Clear

- These bits can clear the Alarms Status Registers. A bit logic level from 1b'0 to 1b'1 clears the sticky alarm to reflect on the latest status.
- The Alarm Clear Registers must be set to 1b'0 for the Alarm Status Registers to reflect properly.

### 2.4.3 Alarm Master Clear

- A Master Alarm Clear bit can clear all the sticky alarms. A bit logic level from 1b'0 to 1b'1 clears the sticky alarm to reflect on the latest status.
- The Master Alarm Clear bit must be set to 1b'0 for the Alarm Status Registers to reflect properly.

### 2.4.4 Alarm Masking

- These register bits can mask the alarm bits to prevent the error from presenting on the ALARM pins or to the PAP logic. The ALARM pins are mainly used for interrupt service routines.
- If the critical alarms need to be present to the ALARM pin logic or PAP logic, then the associated alarm bits must be unmasked by writing 1b'0 to the bits.

### 2.4.5 Alarm Master Mask

- A Master Mask Register can mask all the alarm bits. Setting this bit to 1b'1 can mask all alarms.

## 2.5 Typical Alarm Applications

After understanding of the alarm logics of mask, clear, and actual alarm reading, the following are useful for applications usage of the alarm propagation.

### 2.5.1 Alarm-to-Pin

- Alarm-to-Pin can be triggered through any of the triggered

  common alarms and laneX alarms.
- Alarm-to-Pin can also be triggered whenever any of the count of the individual lane errors (4-bit error threshold counters) crosses a corresponding threshold set for that particular error.
- Alarm generation to PIN can be made sticky or non-sticky through C function.
- Sticky version of all the errors (for generating Alarm-to-Pin) can be read/cleared through SPI.
- All the errors can be independently masked (through SPI) in order not to generate Alarm-to-Pin.

### 2.5.2 Alarm-to-PAP

- Alarm-to-PAP can be triggered through any of the triggered common alarms and laneX alarms.
- Alarm-to-PAP can also be triggered whenever any of the count of the individual lane errors (4-bit error threshold counters) crosses a corresponding threshold set for that particular error.
- Alarm generation to PAP module can be made sticky or non-sticky through C function.

- Sticky version of all the errors (for generating Alarm-to-PAP) can be read/cleared through SPI.
- All the errors can be independently masked (through SPI) in order not to generate Alarm-to-PAP.
- In some use-cases, PAP Alarm generated by PAP module can be routed to the pins. In this case, the JESD204 block generated Alarm-to-PIN is not used.

### 2.5.3 General Information

- There is one global counter per JESD204C lane X error. This counter is shared amongst JESD204C link state machine, Alarm-to-Pin, and Alarm-to-PAP.
- The JESD204C link state machine, Alarm-to-Pin, and Alarm-to-PAP logics all have separate, programmable threshold comparators. Basically, the counters are the same for all three logics, but the thresholds of the logic can be different.
- For each threshold comparator, there is an independent mux option to tap off before or after the threshold counter. The mux is routed to the logic output.

# 3 Physical Layer

## 3.1 Physical Layer Error

### 3.1.1 General Overview of the SerDes Phyiscal Layer

The AFE77xxD devices have two SerDes macros. Each SerDes macro is composed by a STX block with four transmitter lanes, a SRX block with four receiver lanes, and a common PLL block for STX and a common PLL block for SRX. The two SerDes macro provides total of eight SerDes transceiver lanes. The SerDes transmitters, or STX, transmits the serialized JESD204 data streams from the ADC and DDC logics to the ASIC/FPGA. The SerDes receivers, or SRX, receives the serialized JESD204 data streams from the ASIC/FPGA devices to the DAC and DUC logics. The following shows the general block diagram for the SerDes transceivers. Within each SerDes macro, two consecutive STX lanes or two consecutive SRX lanes can be enabled or disabled independently to optimize the power dissipation. For instance, STX1 and STX2 can be powered down together.

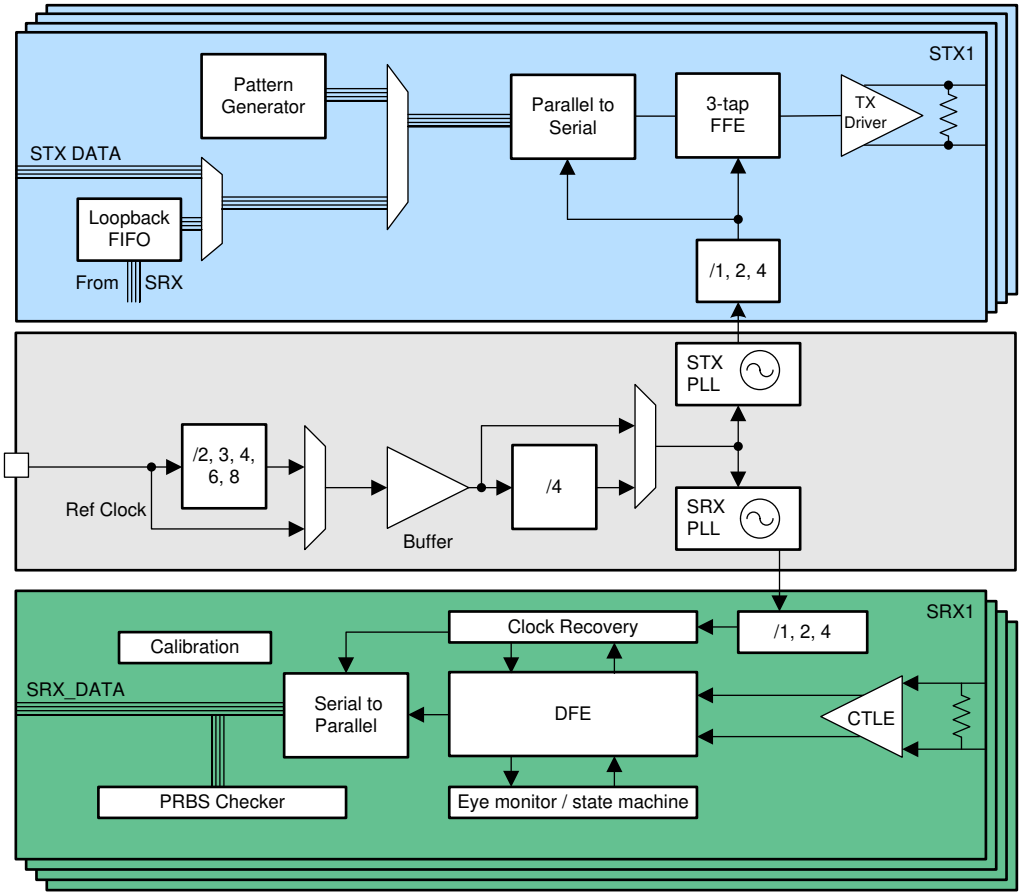


**Figure 3-1. AFE77xxD SerDes Block Diagram**

During the JESD204 operation, the SerDes block can detect errors and notify the user physical layer issues. The following error reportings are available:

### 3.1.2 SerDes Receiver Loss of Signal (LOS)

- This detector of this error is located for each SerDes receiver SRX ports (SRX1 to SRX8). Each SRX has a dedicated alarm.
- This error is triggered when the SRX port is in electrical idle state or has very weak signal input.
- Related C Functions:

```
getJesdRxMiscSerdesErrors()
```

### 3.1.3 SerDes PLL Unlock

- Besides the on-chip PLL, there is a dedicated on-chip SerDes PLL to provide all the necessary clocking for the SerDes block. This error is triggered when SERDES Rx/Tx PLL losses lock.
- Related C Functions:

```
getJesdRxMiscSerdesErrors()
```

### 3.1.4 SerDes Receiver FIFO Error

- Depending on the encoding type of the JESD204C usage, the SerDes block either operates at a rate close to 8B/10B or 64B/66B. To handle various encoding rates and to be able to hand off the data optimally to the JESD204C block, the design includes a FIFO in between the SerDes block and the JESD204C block.
- The SerDes block and the JESD204C block operates in different clock domain to handle various encoding rates. The FIFO in between the blocks absorbs the delay in data hand-off between the two blocks.
- The write-to-FIFO process is where the SerDes hands off the data output to the JESD204 block. This rate can be generically classified as FIFO_WRITE_CLK, and is derived from the SerDes SRX lanes and associated CDR clock. The read-from-FIFO process is where the JESD204 block reads the data from the FIFO. The JESD204 block reads from the FIFO and operates at a rate that is generically called FIFO_READ_CLK.
  - FIFO error occurs whenever the FIFO_WRITE_CLK and/or FIFO_READ_CLK are either missing or have mismatch in clock rate.
  - FIFO error can be cleared through the forced JESD204 RX block reset, assuming both FIFO_WRITE_CLK and FIFO_READ_CLK are stable.
  - If additional FIFO error occurs, the user can adjust the FIFO offset between the write and read point, and then re-initiate the JESD204C link. This is a very rare occurence for debug purpose only.
  - The following C function access the FIFO errors

```
getJesdRxLaneFifoErrors()
```

- Sources of Error:
  - When error occurs, check the STX lanes and JESD204 transmitter block on the FPGA/ASIC logic device are phase locked to the SRX lanes and JESD204 receiver block. If they are not phase locked, then the FIFO may underrun or overrun, hence causing errors.
  - Typically, the FIFO error indicates the FIFO input clock (FIFO_WRITE_CLK) and FIFO output clock (FIFO_READ_CLK) are disturbed. Other clocking path in the system may need to be thoroughly checked.
  - Check the SerDes line rate configured on both the logic device and also the AFE device are within expectations.
  - Signal integrity of the SerDes line such as small eye opening or wrong adaptation can cause CDR to have frequency and phase error. The FIFO_WRITE_CLK are partially based on the CDR, and if CDR is incorrect, the FIFO error can occur.
    - Check the SerDes link status using

```
getJesdRxLosErrors()
```

    - Check eye margin using

```
getSerdesRxLaneEyeMarginValue()
getSerdesEye()
```

  - Re-adapt the SRX of the AFE77xxD to ensure the clock data recovery clock (CDR) is adapting properly.

```
resetSerDesDfeLane()
reAdaptSerDesLane()
resetSerDesDfeAllLane()
reAdaptSerDesAllLane()
```

– Check the SERDES PLL lock status of the AFE device. If the SerDes PLL is unlocked, the clocks can be misaligned resulting FIFO errors.

```
getJesdRxMiscSerdesError()
```

– The JESD204C state machine should automatically re-link upon recovery of the errors. If the error cannot be recovered, JESD204C RX block on the DAC side need to have a re-sync (JESD re-init) procedure performed after all the clocks are recovered. The JESD re-init procedure also re-init/reset the FIFO.

```
adcDacSync()
```

### 3.1.5 SerDes Receiver Adaptation

For the AFE77xxD SRX, the initial bring-up configuration file of the device can have an initial one time CTLE adaptation to determine the CTLE gain value. During this time, the ASIC or FPGA STX need to send out proper training signals to make sure good adaptation of the AFE77xxD SRX core.

Re-adaptation is possible by manually force another adaptation after the initial bring-up. This can force the CTLE to go through the adaptation process given that there are sufficient random and proper training signal from the ASIC or FPGA STX.

```
reAdaptSerDesLane()
```

Regardless of the adaptation process was done during bring-up or manually triggered, the search algorithm is automatic or “AUTO mode”. After adaptation process, the JESD204 link need to be re-established.

The user can also set manually to the fixed gain profile setting and to not go through the gain search cycle. This is typically used for short reach (for example, less than 2 inches and low loss PCB traces) environment. TI recommends to force the CTLE at a low gain setting for low loss environment.

```
setSerdesLaneCtle()
```

The DFE of the AFE77xxD SRX is a one tap DFE, and it can always be adapting. It is possible to freeze the DFE adaptation, but it is not possible to disable the DFE at this point. The DFE and CTLE work together as a loop in the SerDes receiver block.

## 3.2 Physical Layer Testing

For the SerDes based testing, the AFE77xxD supports the generation and the verification of the PRBS patterns. A few example PRBS pattern supported are listed:

- PRBS9 ($x^9 + x^5 + 1$)
- PRBS15 ($x^{15} + x^{14} + 1$)
- PRBS23 ($x^{23} + x^{18} + 1$)
- PRBS31 ($x^{31} + x^{28} + 1$)

The higher the order, the longer the pattern duration is before it repeats itself. For example, for PRBS31, it takes $2^{31}$ cycles before the pattern repeats itself.

For 8B/10B encoding, assuming the worst case repetitve DC code, the coding technically can repeat every 10bit, or approximately $2^3$ cycle. This is assuming neutral parity. Therefore, the best PRBS pattern to mini mimic such condition is the PRBS9. If the signal becomes more random or pseudo-random, then higher order PRBS can be utilized.

The 64B/66B or 64B/80B encoding by design of the coding have up to $58^{th}$ order of scrambling. Therefore, PRBS31 can be used for the error testing.

### 3.2.1 STX Chain Test Pattern Generator

The STX chain supports the generation of the PRBS pattern. This pattern generation can test the SRX chain at the FPGA/ASIC side for initial physical layer quality test. It can also intentionally generate errors to check the validity of the test.

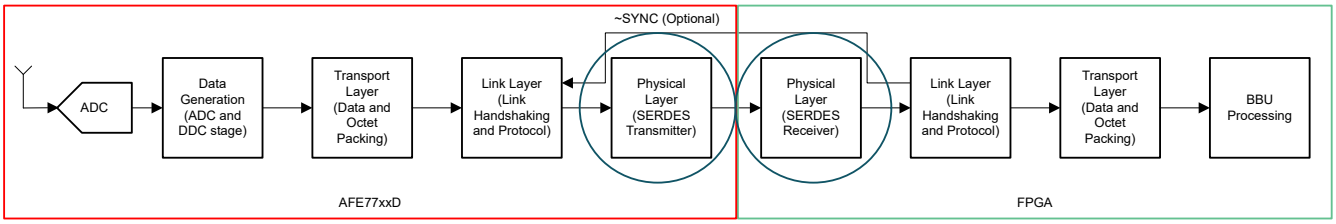


**Figure 3-2. Physical Layer of ADC to Send Check Pattern to Physical Layer of FPGA**

Related C Functions for STX Test Pattern Generator Configurations:

```
sendSerdesTxPrbs()
serdesTx1010Pattern()
serdesTxCustomPattern()
serdesTxSendData()
serdesTxPrbsInsertSingleError()
```

### 3.2.2 SRX Chain Test Pattern Verifier and Eye Margin Check

The SerDes receiver chain integrates a checker for detecting the PRBS pattern error. The PRBS error can be accumulated through error counter to achieve BER testing (for example, the number of error occurred over certain time interval). The receiver includes also an SRX received eye monitor (after SRX equalization) to detect the eye height as well as generating more detailed eye diagrams, that can be read through the SPI.

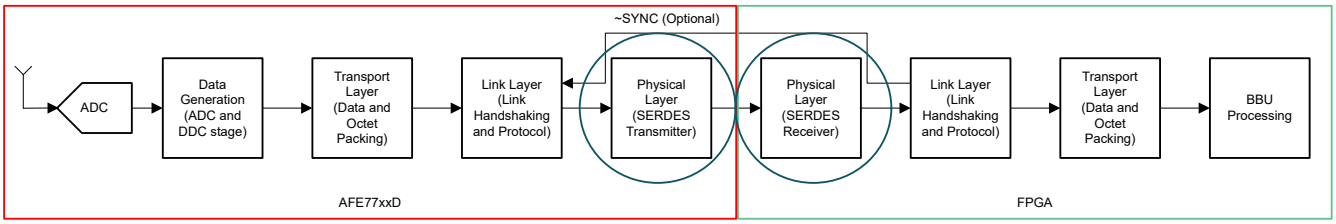


**Figure 3-3. Physical Layer of FPGA to Send Check Pattern to Physical Layer of DAC**

Related C Functions for the SRX Error Checker Configurations:

```
enableSerdesRxPrbsCheck()
getSerdesRxPrbsError()
clearSerdesRxPrbsErrorCounter()
```

Related C Functions to Acquire Eye Margin for the SRX:

```
getSerdesRxLaneEyeMarginValue()
```

This function returns the measured peak height of the serdes eye in mV after the SRX CTLE and DFE equalizer. The larger the number, the better the eye opening. The output of the function returns two times the peak height.

### 3.2.3 SRX to STX Internal Loopback

#### SRX to STX Internal Loopback

The SerDes block within the AFE77xxD also supports loopback of the RX block recovered data to the TX block through a FIFO. This loop can be used to test the overall SerDes transmit and receive quality at the FPGA/ASIC side, as the FPGA/ASIC is responsible for generating the pattern and receiving the pattern. Any error detection is done on the FPGA/ASIC side. The routing of the bit error rate testing using loopback is shown in Figure 3-4.

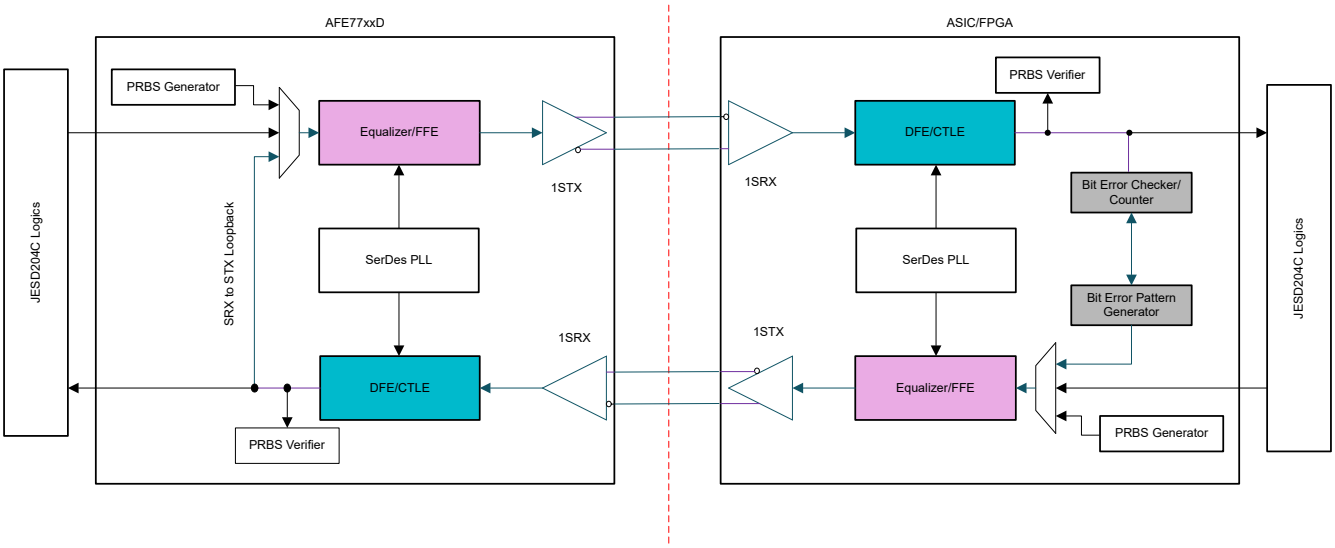


**Figure 3-4. General Block Diagram for the SerDes Loopback Testing From the ASIC/FPGA to the AFE77xxD Devices**

Related C Functions for SerDes Loopback Testing:

```
serdesRxToTxLoopBack()
```

### 3.2.4 SRX Eye Diagram Construction

The SRX block also supports eye diagram and eye height margin. For training related to eye diagram, please visit the training portal, *TI Precision Labs – Signal Conditioning: What is an Eye Diagram?*

Each SRX has the ability to create a matrix of data designed for plotting an eye diagram of BER against phase error along the x-axis and amplitude along the y-axis.

**Note**

The measurement is non-destructive and can be made on live traffic, although the BER of the channel can temporarily deteriorate. Please note the test can take several minutes to run.

The complete matrix of data is 33 x 95 pixels in size in a matrix of the form {phase[-16:16], amplitude[-47:47]. Each pixel returns a value from 0 to 255 with which the BER can be calculated by the function.

Related C Functions for SerDes Eye Diagram Constructions:

```
getSerdesEye()
getSerdesLaneCtle()
getSerdesLaneDfe()
```

After running the function getSerdesEye(), the user can use the data from the function to generate the eye diagram. The following example Python code can take the data and plot the eye diagram.

```
import numpy
import matplotlib.pyplot
# eyeReadVals and extent are obtained from the C read out.
ber=[]
for i in range(95):
    ber+=eyeReadVals[i::95][:]
ber=numpy.array(ber).reshape((95,33))
f=matplotlib.pyplot.figure()
x=numpy.arange(-16.5, 17.5)*0.0232
y=numpy.arange(-47.5, 48.5)*extent/47
c=matplotlib.pyplot.pcolor(x, y, ber/-16, cmap='jet', vmin=-8, vmax=0, figure=f)
cax=f.add_axes([0.9, 0.1, 0.02, 0.8])
```

```
f.colorbar(mappable=c, cax=cax)
matplotlib.pyplot.show()
```

**Note**

The *extent* parameter is the actual scale retuned by the eye diagram code. This parameter can be applied to the returned eye plot values when trying to plot the eye diagram.

### 3.2.5 Bit Error Rate

Each SRX lane has its own PRBS checker. All eight SRX lane PRBS checkers can work simuletaneously. Two 16-bit registers are used to count PRBS errors with the max number of 4294967295. The PRBS checker is updated real-time, and the update frequency depends on the SPI bus speed.

To calculate BER, the user must define a time duration between the starting and stopping of the PRBS counter. The start and stop are controlled via SPI registers writes. In terms of C function, the user can simply execute the following C functions, once at starting time, and another at the stopping time. The difference of the error count over the time duration will be the BER. Note that if the time duration is sufficiently long to allow the error accumulation to be greater than the max counter value, the PRBS count register will cycle back to 1 when the max count is reached.

**Note**

TI recommends to *not* execute both SRX eye scan and SRX PRBS error counter at the same time. Doing so will impact the accuracy of the PRBS error counter.

#### *Pseudo Code for BER Calculation*

```
Error_start = getSerdesRxPrbsError()
wait(T_duration)
Error_stop = getSerdesRxPrbsError()
BER = (Error_stop - Error_start)/(T_duration)
```

#### *BER Measurement Example*

```
#Step 1: enable the PRBS Checker with PRBS31
enableSerdesRxPrbsCheck(afeId, laneNo, 3, 1);
getSerdesRxPrbsError(afeId, laneNo, &Error_start)

#Step 2: have sufficient wait time for the error to accumulate. This depends on the BER definition
wait(T_duration);

#Step 3: gather statistics such as PRBS error, Eye margin, CTLE value, and DFE coefficients.
getSerdesRxPrbsError(afeId, laneNo, &Error_stop);
getSerdesRxLaneEyeMarginValue(afeId, laneNo, &margin);
getSerdesLaneCTLE(afeId, laneNo, &ctle);
getSerdesLaneDFE(afeId, laneNo, &dfe);

#Step 4: calculate BER
#General BER calculation
#BER = (Error_stop - Error_start)/(T_duration)

#Step 5: disable PRBS checker
enableSerdesRxPrbsCheck(afeId, laneNo, 3, 0);
```

## 3.3 Common Questions on SerDes Optimization

#### *CDR Lock Detect*

***Q: Are there any registers indicating if the CDR is locked or not ?***

A: Yes there is a CDR locked status register. It can be accessed through the cFunction *getSerdesLinkStatus()*

#### *Minimum Eye Height Requirement*

***Q: What is the minimum eye height and width (for instance, at 16.22Gbps and 32.44Gbps) after the equalization that can be used for customizing an eye mask for the eye monitor?***

A: There is no specification for any parameter after equalization because there is no direct correlation between BER and eye margin. As a rule of thumb, the eye margin should not be smaller than 300mVpp for any application. TI recommends to use BER as criteria for the eye optimization.

***FOM (figure of merit)***

***Q: Does AFE77xxD has a register with FOM (figure of merit) value or something similar. FOM is a very common register on SerDes to indicate the health of the SRX signal.***

A: The AFE77xxD SRX ports have eye height read-out, which is an equivalent to FOM. This can be accessed using function *getSerdesRxLaneEyeMarginValue()*.

***Short Reach Applications***

***Q: What are the recommendation for short reach traces and medium from the ASIC/FPGA to the AFE77xxD (i.e. 2 inches of traces from the FPGA to the AFE77xxD)?***

A: TI recommends to contact the applications team for additional detail. Typically, the FPGA STX driver will need adjustment to the pre-cursor of -5% of main STX cursor. TI will also need to set SRX to manual CTLE mode as oppose to adapative mode to fix the CTLE gain and bypass the initial CTLE adaptation.

## 3.4 SerDes Block Register Access

For accessing SerDes registers, first the appropriate page needs to be selected.

**Table 3-1. SerDes Block Page Select Writes**

| Functional Blocks | Page Select Write Address | Page Select Write Data |
|---|---|---|
| SerDes 0 | 0x0016 | 0x20 |
| SerDes 1 | 0x0016 | 0x40 |
| SerDes 0 and SerDes 1 | 0x0016 | 0x60 |
| SUBCHIP | 0x0016 | 0x10 |

There are two types of SerDes registers within the SerDes. Some registers are common to the four lanes within the block, and there are lane specific registers.

Accessing the SerDes registers is not direct as the native registers within the SerDes block are 16-bit wide. However, the AFE77xxD SPI data is only 8-bit wide. Therefore, additional translation steps are in place to use AFE77xxD SPI bus to access the native SerDes block register.

For writing to register address, A, first write the MSB 8-bits of the data to address 2*((A+0x2000)[13:0])+1 and then the LSB 8-bits to 2*((A+0x2000)[ [13:0]). Accessing only 8-bits is not possible. The complete 16-bits needs to be accessed in the same sequence.

Please note that register addres A is a 16-bit format SerDes register address referenced in the AFE77xxD TRM. The corresponding 8-bit format register address is referenced Table 3-2.

**Table 3-2. SerDes Block Lane Prefix Addresses per SRX Lane**

| Lane | Page Select | Serdes Lane Prefix (16-bit Format) | Serdes Lane Prefix (8-Bit Format) |
|---|---|---|---|
| SRX1 | 0x0016 0x20 | 0x81xx | 0x42xx, 0x43xx |
| SRX2 | 0x0016 0x20 | 0x80xx | 0x40xx, 0x41xx |
| SRX3 | 0x0016 0x20 | 0x82xx | 0x44xx, 0x45xx |
| SRX4 | 0x0016 0x20 | 0x83xx | 0x46xx, 0x47xx |

***Writing 0x1234 to SerDes Block Register 0x8413***

For example to write 0x1234 to register 0x8413,

Step 1: Data to write is 0x12. Address to write 0x413*2+1=0x827.

0x4827 0x12

Step 2: Data to write is 0x34. Address to write 0x413*2=0x826.

0x4826 0x34

***Reading from SerDes Block Register 0x8413***

Read is also similar, however, each read needs to happen twice and only the second read is valid. To read register, 0x8413,

Step1: Read 0x4827 discard the read.

Step2: Read 0x4827 the read back value is the MSB.

Step3: Read 0x4826 discard the read.

Step4: Read 0x4826 the read back value is the LSB.

**CAUTION**

In register map, the MSB and LSB addresses are directly given. While accessing, the user need to access both together. Not doing so can cause invalid access to SerDes block registers, and can be recovered only through complete reset of the AFE77xxD.

# 4 Link Layer

## 4.1 Link Status

### 4.1.1 8B10B Encoding Status

Per JESD204C standard, section 8, the 8B/10B encoding includes two main state machines: code group synchronization and initial frame alignment.

#### *4.1.1.1 Code Group Synchronization:*

Per JESD204C Standard, section 8.6.1, Figure 101 and Table 58, the code group synchronization has the following process with three states.

*Copyright JEDEC Solid State Technology Association*

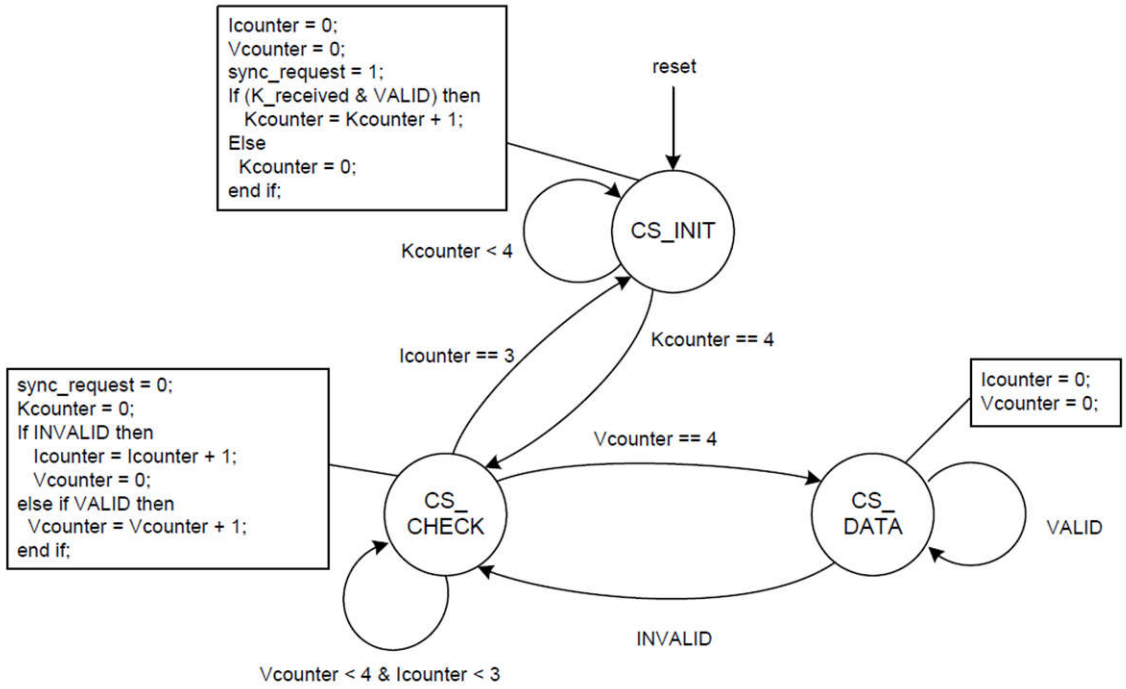


**Figure 4-1. 8B/10B Encoding Code Group Synchronization States (JESD204C Figure 101 and Table 58)**

Each lane will have a two bit indicator to reflect the current status of the CS state. The definition of the CS state is as follow, corresponding to the three states defined in Section 8.6.1 of the JESD204C documentation:

| **CS State** | |
|---|---|
| **Bits** | **Definition** |
| 2B'00 | CS_INIT |
| 2B'01 | CS_CHECK |
| 2B'10 | CS_DATA |
| 2b'11 | Undefined |

#### 4.1.1.2 Initial Frame Alignment or Frame Synchronization

Per JESD204C documentation, section 8.6.2, Figure 102, the initialization frame alignment has the following process with three states:

*Copyright JEDEC Solid State Technology Association*

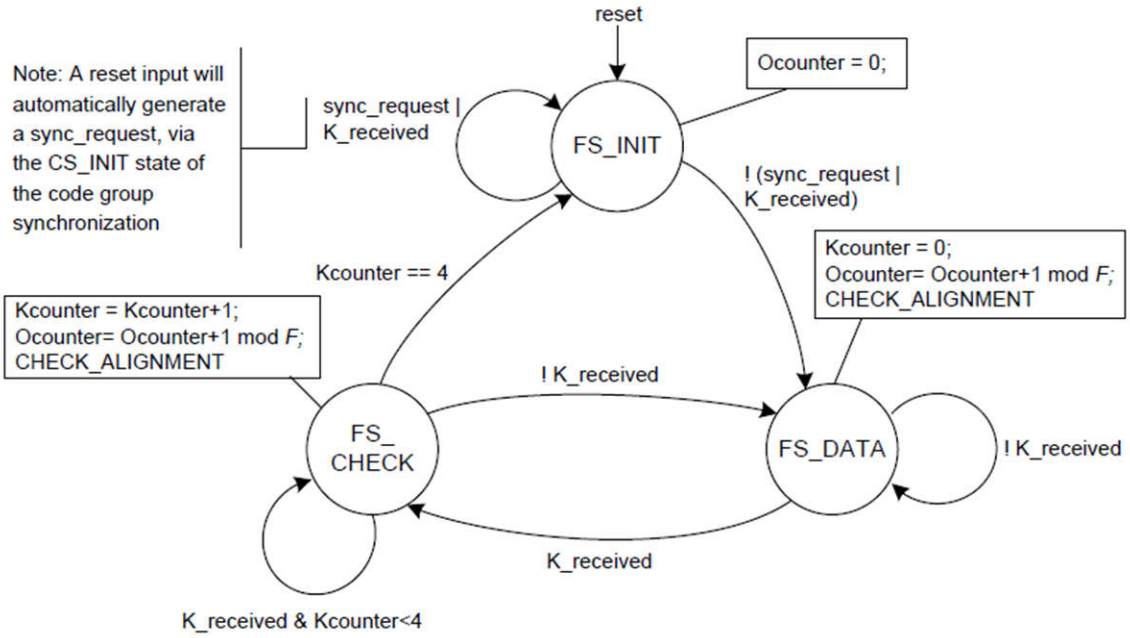


**Figure 4-2. 8B/10B Encoding Frame Alignment States (JESD204C Figure 101)**

Each lane will have a two bit indicator to reflect the current status of the FS state. The definition of the FS state is as follow, corresponding to the three states defined in section 8.6.2 of the JESD204C documentation:

| **FS State** | |
|---|---|
| **Bits** | **Definition** |
| 2b'00 | FS_INIT |
| 2b'10 | FS_CHECK |
| 2b'01 | FS_DATA |
| 2b'11 | FS_UNDEFINED |

In order for 8b/10b encoding to pass link establishment, all the lanes within the link (aggregation) must have valid CS_DATA (i.e. 2b’10) and FS_DATA (i.e. 2b’01) state.

### 4.1.2 64B/66B and 64B/80B Encoding Status

#### 4.1.2.1 Sync Header Aligment

Per JESD204C documentation, section 7.6.1, Figure 83, the 64/66B and 64/80B encoding will first go through sync header alignment. The goal of the sync header alignment is to recognize the boundary of the eight octets

along with dedicated two bit sync alignment header, and detect any potential loss of alignments where there are absent or incorrect sync header streams.



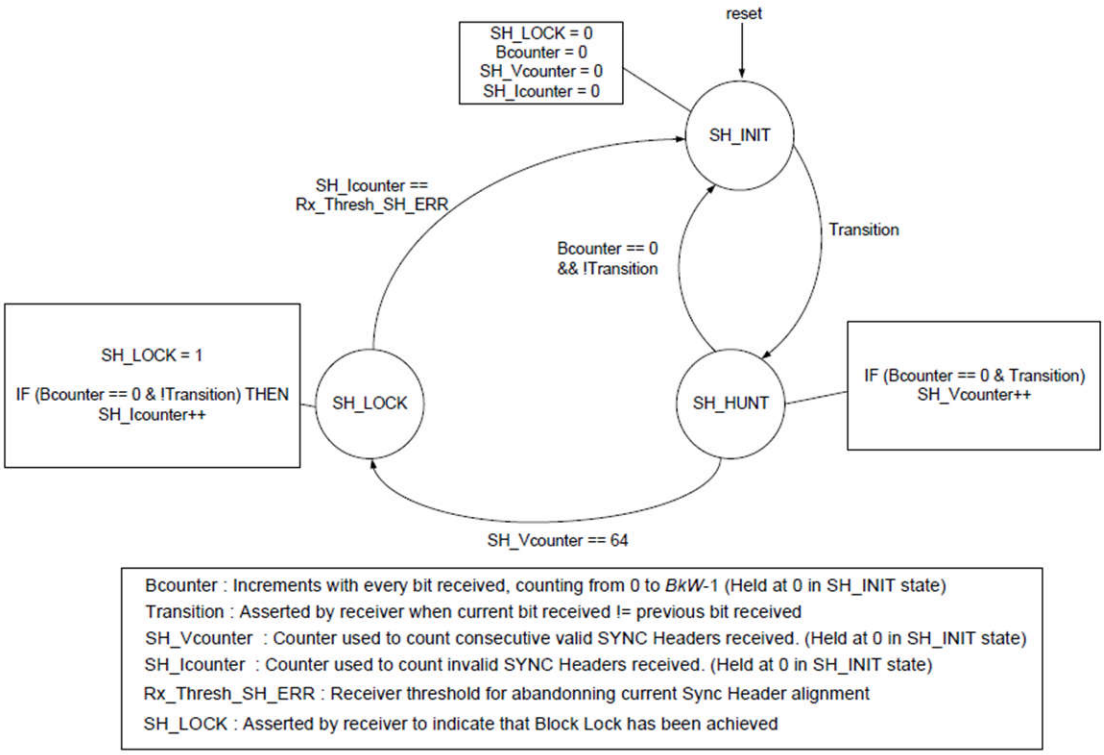


**Figure 4-3. 64B/66B and 64B/80B Encoding Sync Header Alignment States (JESD204C Figure 83)**

Each lane will have a two bit indicator to reflect the current status of the SH state. The definition of the SH state is as follow, corresponding to the three states defined in section 7.6.1 of the JESD204C documentation:

| **SH State** | |
|---|---|
| **Bits** | **Definition** |
| 2B'00 | SH_INIT |
| 2B'01 | SH_HUNT |
| 2B'10 | SH_LOCK |
| 2b'11 | Undefined |

### 4.1.2.2 Extended Multiblock Aligment

Per JESD204C Standard, section 7.6.2, Figure 84, the 64/66B and 64/80B encoding will also go through the extended multiblock (EMB) alignment. After the EMB alignment, the **E** number of multiblocks will be aligned on the JESD204 RX IP. The state machine also monitor any loss of alignments to the EMB.



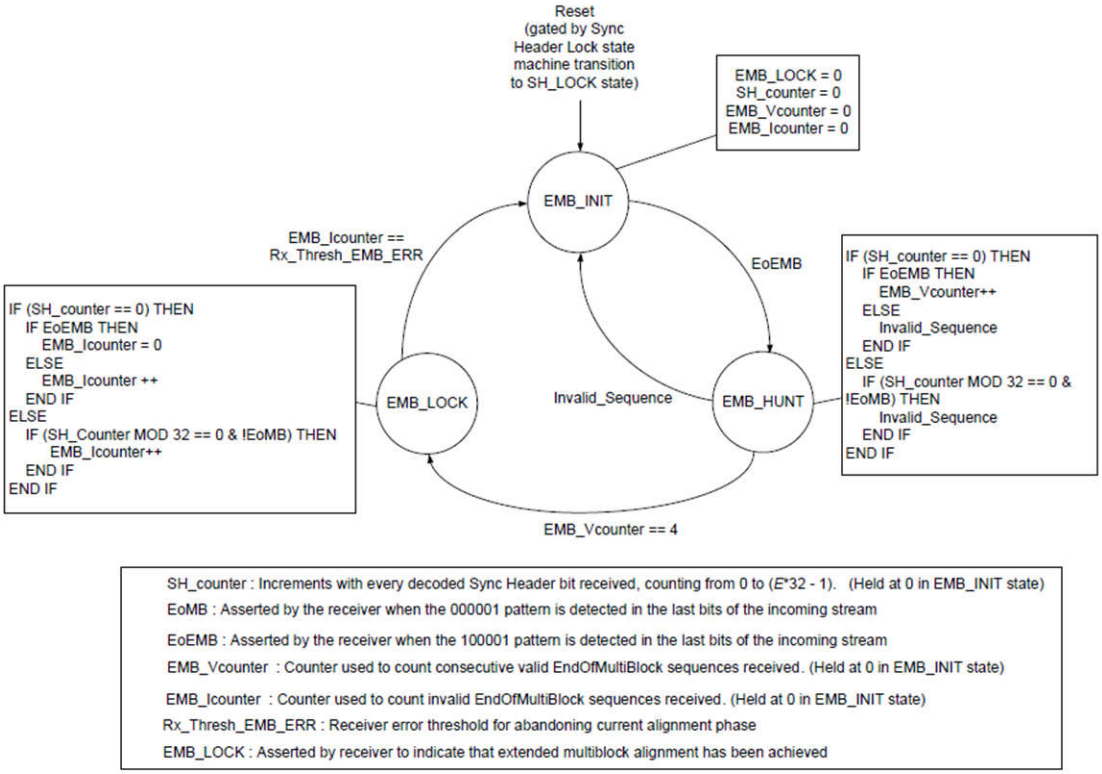


**Figure 4-4. 64B/66B and 64B/80B Encoding Extended Multiblock Alignment States (JESD204C Figure 84)**

Each lane will have a two bit indicator to reflect the current status of the EMB state. The definition of the EMB state is as follow, corresponding to the three states defined in section 7.6.2 of the JESD204C documentation:

| **EMB State** | |
|---|---|
| **Bits** | **Definition** |
| 2b'00 | EMB_INIT |
| 2b'01 | EMB_HUNT |
| 2b'10 | EMB_LOCK |
| 2b'11 | EMB_UNDEFINED |

In order for 64B/66B and 64B/80B encoding to pass link establishment, all the lanes within the link (aggregation) must have valid SH_LOCK (i.e. 2b’10) and EMB_LOCK (i.e. 2b’01) state.

### 4.1.3 Current State Status

Besides reading the current state machine status, the AFE77xxD also allows the reading of the previous state machine status to ensure the correct state transition. The corresponding regsiters are:

DAC JESD page:

| Register Address | Register Name | | Function |
|---|---|---|---|

DAC JESD page:

| | | | |
|---|---|---|---|
| 0x19C | jesd_sh_state | 8B/10B | Current Code Synchronization (K28.5 Search State). For stable link, this value should be read as 2b'10. |
| | | 64B/66B and 64B/80B | Current Sync Header State. For stable link, this value should be read as 2b'10. |
| 0x19E | jesd_cs_state | 8B/10B | Current Code Synchronization State. For stable link, this value should be read as 2b'10. |
| | | 64B/66B and 64B/80B | Current Extended Multiblock Alignment Statement. For stable link, this value should be read as 2b'10. |
| 0x1A0 | jesd_fs_state | 8B/10B | Current Initial Frame Alignment State. For stable link, this value should be read as 2b'01. |
| | | 64B/66B and 64B/80B | Not Used |
| 0x1A2 | jesd_buf_state | 8B/10B | Current Elastic Buffer (RBD) State. For stable link, this value should be read as 2b'11 for each lane |
| | | 64B/66B and 64B/80B | |

### 4.1.4 JESD204C RX (DAC Link) C Functions

***Corresponding C function for the JESD204C RX Section (DAC Link):***

1. `getJesdRxLinkStatus()`

   for all the lanes aggregated into the link.
2. `getJesdRxLaneLinkStatus()`

   for individual lane status

### 4.1.5 JESD204C TX (ADC Link) Status

The JESD204C TX block also has the state machine indicator. It is located in the ADC JESD page, JESD_SYNC_ST ATE_LANE3/2/1/0 for each of the four lanes in the ADC JESD block. There are two ADC JESD blocks available for total of eight JESD204C TX lanes.

**ADC JESD page:**

| Register Address | Register Name |
|---|---|
| 0x180, bit[7:6] | JESD_SYNC_STATE_Lane3 |
| 0x180, bit[5:4] | JESD_SYNC_STATE_Lane2 |
| 0x180, bit[3:2] | JESD_SYNC_STATE_Lane1 |
| 0x180, bit[1:0] | JESD_SYNC_STATE_Lane0 |

## 4.2 Link and Lane Errors

The AFE77xxD can detect link and lane errors on the DAC JESD204C RX logic. Please see Table 4-1 for the summary of error detections.

**Table 4-1. AFE77xxD Link and Lane Error Detections**

| **8B/10B Encoding** | |
|---|---|
| **Error Number** | **Error Description** |
| **10** | /R/ and /Q/ control characters are found in data mode |
| **9** | ILA ramp counter check invalid (along with control characters /R/, /Q/, and /A/) |
| **8** | Frame sync error (received /K/ characters during data mode) |
| **7** | Multi-frame alignment error |
| **6** | Frame alignment error |
| **5** | Link configuration error |
| **4** | Elastic buffer overflow (bad RBD value) |
| **3** | Elastic buffer match error. The first non-control /K/ character does not match the "match_ctrol" and "match_data" programmed values. |
| **2** | Code synchronization error |
| **1** | 8b/10b not-in-table code error |
| **0** | 8b/10b disparity error |
| **64B/66B and 64B/80B Encoding** | |
| **10** | Unused |
| **9** | Unused |
| **8** | Fixed ones in pilot signal error |
| **7** | EoEMB alignment error |
| **6** | EoMB alignment error |
| **5** | CMD data parity error in CRC mode. The CMD data in CRC mode is not matching the pre-registered key in SPI register. |
| **4** | Elastic buffer overflow (bad RBD value) |
| **3** | FEC syndrome error. |
| **2** | Extended multiblock alignment error |
| **1** | Sync-header invalid error |
| **0** | CRC error (OR) FEC uncorrectable error |

### 4.2.1 *8B/10B Errors (8B/10B Encoding Specification Only)*

- **Disparity error**: *the received code group exists in the 8b/10b decoding table, but is not found in the proper column to the current + or – running disparity.* The 8b/10b encoding scheme utilizes disparity to keep the number of zero bits and number of one bits to 1:1 ratio and maintains DC balance on the physical SerDes line. Typically, for every negative disparity (i.e. four zeros and six ones) code, there follows a positive disparity (i.e. six zeros and four ones) code. There are other exceptions for completely balanced code (i.e. Five zeros an five ones). The error occurs when the disparity sequence received is not within the rules of the 8b/10b encoding.
- **Not-in-table error**: *the received code group is not found in the 8b/10b decoding table for either disparity*. With 8-bit word, there are $2^8$ or 256 bit combinations. This is translated into 10-bit word, which has $2^{10}$ or 1024 bit combinations. When a not-in-table error occurs, this is usually a true bit error since the detected word lies within the 768 unused bit sequences.

Error Handling:

- If error occurs, please check the JESD204C TX logic device (i.e. FPGA or ASIC) to see if it is initialized correctly with proper 8b/10b coding logics running. Uninitialized 8B/10B coding logic may result such error.

- Typically, both 8b/10b not-in-table and 8b/10b disparity occur simultaneously, and indicate general bit error due to the SerDes signal conditioning. This raises the need to improve signal integrity through use of equalizer, for example.
- If only 8b/10b disparity occurs, the error may lead to the 8b/10b logic in the JESD204C IP not being reset properly.
- Probe the SerDes lines with high speed scope to check the eye diagram for SerDes signal quality. PRBS pattern such as PRBS9 may be run over period of time for bit error check.
- Check if the SRX of the AFE77xxD are detecting any bit errors. Re-adapt the SRX of the AFE77xxD and adjust the signal integrity settings (i.e. FIR settings from STX of the FPGA or ASIC) accordingly.

### 4.2.2 *Code Group Synchronization Error (8B/10B Encoding Specification Only)*

Per JESD204C Section 8, consecutive 8B/10B disparity error or Not-In-Table errors will cause loss of link synchronization. Upon link synchronization, the JESD204C link will go through code group synchronization (CGS) which involves the JESD204C RX IP to detect the handshaking CGS code, or /K/ or K28.5 code. If the JESD204C RX IP cannot recognize the code or unable to establish link synchroniztion, the error will occur. This error is cleared when the link has been re-established.

- Code group synchronization (CGS) is achieved when the lane has received four consecutive /K/ (K28.5) characters successfully after asserting a synchronization request (~SYNC) and then another four valid character (i.e. non-/K/ characters that are valid in the 8B/10B coding) s after de-asserting the synchronization request. Basically, the lane alignment is complete at this point and the data is being buffered until frame alignment completes.
- If three invalid characters are received within a certain amount of time, code group synchronization would be lost.
- This would trigger the code group synchronization error.

Error Handling:

- If error occurs, please check the JESD204C logic device at either FPGA or ASIC to see if the logic device is initialized correctly with proper CGS state machine running
- Probe the SerDes line with high speed scope to see if the actual /K/ are detected with sufficiently good eye opening.
- Run link layer testing to check for validity of the /K/ CGS patterns
- Check if the SRX of the AFE77xxD are detecting any bit errors. Re-adapt the SRX of the AFE77xxD and adjust the signal integrity settings (i.e. FIR settings from STX of the FPGA or ASIC) accordingly.

### 4.2.3 *Elastic Buffer Mismatch (8B/10B Encoding Specification Only)*

Unexpected control character: after the code group synchronization with /K/, the first character received is not the expected /R/ character.

**ĐAC JESD Page**

| Register Address | Register Information | Register Description |
|---|---|---|
| 0xE9, bit[6] | Match_Specific | – If set to 1b’0, the JESD204C RX buffer will start buffering with the first non-/K/ value and ignore match control character. Programming match_specific to 1b’0 should not be used unless for debugging purpose<br>– If set to 1b’1, the JESD204C RX buffer will start buffering only with /R/ character, immediately followed by /K/ after the completion of CGS |
| 0xE9, bit[7] | Match_CTRL | – When set to 1b'0 the match character is a DATA character.<br>– When set to 1b'1 the match character is a CONTROL character (/R/) instead of a DATA character. |

Error Handling:

- If error occurs, please check the JESD204C logic device to see if it is set correctly to subclass 1 mode as oppose to unsupported subclass 0 mode.
- Check with high speed scope to see if the actual /R/ is detected after the /K/ characters in the CGS.
- Set match_specific to 1b'0 to see if the link establishment can proceed.
- Check if the SRX are detecting any bit errors. Re-adapt and adjust the signal integrity settings accordingly.

### 4.2.4 *Elastic Buffer Overrun (Common Specification)*

This is the elastic buffer overflow within the JESD204C RX IP. This is either due to incorrect configuration of RBD or very large lane-skew beyond the buffer size.

Refer to RBD Optimization

### 4.2.5 *Link Configuration Error (8B/10B Encoding Specification Only)*

- The frame alignment block of the JESD204C RX IP performs the initial frame synchronization, verification of the link configuration parameters, and frame alignment monitoring.
- When the initial lane alignment sequence arrives in this block, the first non-/K/ character is marked as the first octet of the first frame. The link configuration data is expected to arrive in the second multiframe. This block verified that the JESD204C TX lanes are programmed with the same configuration as the JESD204C RX lane.
  - Any mismatches in the configuration parameter would trigger the link configuration error.
  - Link configuration data is expected to arrive on the second multiframe of the initial lane alignment sequence. Basically the second /R/ indicates the ILAS
  - Each lane should have unique lane ID that is tied to the physical lane.
  - Correct programming of the ILAS configuration are needed for both side of the JESD204C link (i.e. ASIC/ FPGA and DAC logic) in order for the ILAS check to be successful.



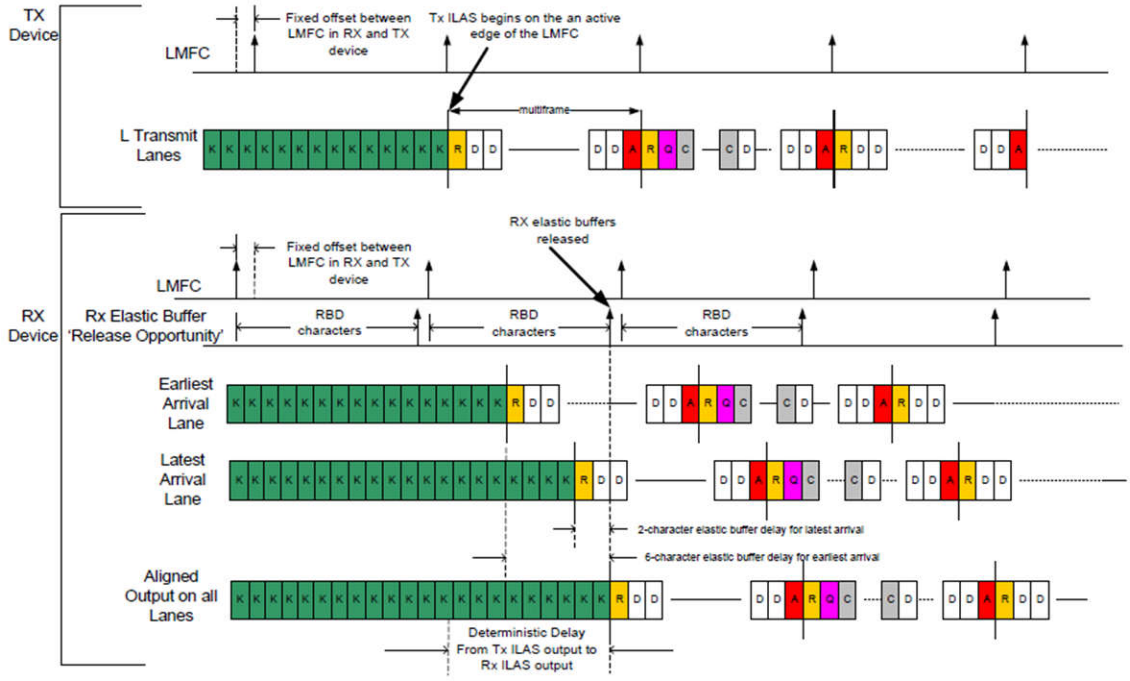


**Figure 4-5. Illustrating ILAS After /K/ and /R/ Characters (JESD204C Figure 5)**

Error Handling

- If link cannot be established, try to ignore ILAS check in SYNC_REQUEST to check if the JESD204C link can be established properly.

- Check the ASIC/FPGA ILAS setting and sequence are set correctly. Both the DAC JESD204C RX IP parameters and the ASIC/FPGA parameters much be set the same.
- Probe the SerDes line with high speed scope to trigger on first non-K28.5 character to decode the ILAS sequence. Go through the sequence to see if the octets are properly replaced for ILAS.
- Run link layer test to test for ILAS sequence.

### 4.2.6 *Frame and Multiframe Alignment Error (8B/10B Encoding Specification Only)*

Section 8.4.4 of the JESD204C specification defines the frame and multiframe alignment errors.

> *Frame alignment is monitored via alignment characters, which are inserted by the transmitter under certain conditions at the end of a frame. The receiver resynchronizes its frame to the alignment characters after checking that their reception is not likely to have been caused by a bit error on the lane.*
>
> *Resynchronization will require repeated reception of a valid alignment character at the same unexpected position in the frame.*
>
> *The alignment character shall be a frame alignment character /F/= /K28.7/. However, if both sides of the lane support lane alignment, the lane alignment character /A/= /K28.3/ shall be used in the last frame of a multiframe.*

The following errors are indications of frame and multiframe alignment mismatch.

1. Frame Alignment Error: /F/ received at the unexpected location
2. Multiframe/Lane Alignment Error: /A/ received at the unexpected location.
3. Frame Synchronization Error: Consecutive /K/ received during data mode.

These errors are cleared when the JESD204C link is reset.

The following process describes the relationship of frame, multiframe, local multiframe boundary, and the potential sources of errors:

- Frame (F): a set of consecutive octets in which the position of each octet can be identified by reference to a frame alignment signal.
- Multi-frame (K) : a set of consecutive frames in which the position of each frame can be identified by reference to a multiframe alignment signal.
- Frame boundary is needed for both the JESD204C transmitter and JESD204C receiver to have pre-determined way to “process” the bit packing and octet packing of the data octets. This is performed in the transport layer of the JESD204B specification.
- Multiframe boundary is needed such that the JESD204C transmitter and JESD204C receiver can perform logics in the data path using their own perspective “local clock”. This is the local multiframe clock (or LMFC), which defines the logical processing clock to process the data octets.
- The ideas is that the LMFC of each perspective JESD204C block can have known, well-defined, and deterministic delay between the data source and data receiver. The clocks do not have to have absolute delay defined in order to simplify clocking solution.
- The deterministic delays can be absorbed by the release buffer (elastic buffer) in the JESD204C receiver.

Multiframe and Frame Alignment Alarms

- Frame alignment monitoring is constantly performed by checking that replacement characters /F/ and /A/ arrive at expected positions in the frame and multiframe.
- When these characters are received, they are replaced with actual data.
- If they are consistently received at unexpected positions, then frame alignment error or the multiframe alignment error would be triggered to indicate the alignment has been lost.
- Frame alignment correction is *not* supported. Therefore, the host need to perform re-alignment and resynchronization of the JESD204C link based on alarm feedback
- The following sections of the JESD204C Standard highlights the Frame and Multiframe Alignment:
  - 8.4.4 Frame alignment monitoring and correction
  - 8.4.4.1 Alignment characters

Sources of Error:

- Drift” of SYSREF reset of LMFC over time.
  - SYSREF reset of LMFC change in the “periodic” time instance over time.
  - The overall LMFC boundary changes and potentially cause alignment errors.
  - “Drift” of the clock source to various logics.
  - “Drift” of incoming data on the JESD204C transmitter over time.
- The incoming data are sufficiently periodic and not sufficiently random. The character replacement rules for the /A/ and /F/ characters are not exercised sufficiently, and hence the periodic frame and multi-frame are not checked frequently.
- Incorrect initialization of the logical block in the JESD204C TX. The JESD204C IP was not reset properly upon start-up. Glitch occurs after various temperature cycle.

### 4.2.7 Sync Header Errors (64B/66B and 64B/80B Encoding Specification Only)

For every multi-block (consisted of 32 blocks of octets), there are two bit sync headers of either 2b’01 or 2b’10 format to ensure continuous synchronization of the JESD204 ink. These sync headers are also used to construct the CRC3, CRC12, or the FEC coding for error detection. Depending on the number of Extended Multi-block setup (EMB), the error on EMB could also be triggered.

The following are the related errors reported related to Sync Header.

***CRC12/CRC3 Error or FEC uncorrectable Error.***

This occurs when the full frame reconstruction of CRC3, CRC12, or FEC coding received from the JESD204 TX logic is not matching the expected behavior within the JESD204 RX.

***Sync-header Invalid Error:***

The Sync Header need to be binary bit stream of either 2b’01 or 2b’10. This error is triggered when the JESD204 RX received the non-valid code of 2b’00 or 2b’11.

Encoding and decoding invalid error could occur for the general pilot signal, CRC header, FEC header, and CMD header.

**Table 4-2. SYNC Header Encoding**

*Copyright JEDEC Solid State Technology Association*

| Sync Transition Bit | Sync Header[0:1] |
|---|---|
| 0 | 10 |
| 1 | 01 |

**Table 4-3. Sync Header Decoding**

*Copyright JEDEC Solid State Technology Association*

| Sync header [0:1] | Sync transition bit |
|---|---|
| 00 | Invalid |
| 01 | 1 |
| 10 | 0 |
| 11 | Invalid |

***Sync-header Alignment loss (OR) EMB Alignment***

When this error occurs, the JESD204C link is lost and need to resynchronize. The sync-head aligment loss is defined as per JESD204C section 7.6.1 where event of *SH_Icounter* equals *Rx_Thresh_SH_ERR* in the sync header aligment state machine.

The EMB alignment loss is defined per JESD204C section 7.6.2 where event of EMB_Icounter equals Rx_Thresh_EMB_ERR in the EMB alignment state machine.

With these errors result link loss, the alarm is cleared when sync-header state machine goes to lock state.

### 4.2.8 CMD parity Error in CRC12/CRC3 mode

There are total of 6 CMD bits (6 SPI bits + 1 parity derived from the XOR of the 6 SPI bits). This parity check utilizes the CMD bits from the JESD204C transmitter (i.e. FPGA or ASIC) and check against the SPI bits. If the two does not match, the error will be flagged more immediately than the CRC errors. The default is set to zeros for the CMD data.

**Table 4-4. CMD Parity Error Programming**

**DAC JESD Page**

| Register Address | Register Description |
|---|---|
| 0x1AC | JESDC_CMD_DATA[7:0] |
| 0x1AD | JESDC_CMD_DATA[15:8] |
| 0x1AE | JESDC_CMD_DATA[17:16] |
| 0x1AB, bit[2] | JESDC_CMD_PARITY_FLIP |
| 0x1AB, bit[1] | JESDC_CMD_DATA_EN |

### 4.2.9 FEC Syndrome Error

This error is set when non-zero syndrome is detected, and is based on section 7.5 of the JESD204C specification.

> *The transmitter computes the FEC parity bits of the scrambled data bits in a multiblock and encodes these parity bits on the sync header stream of the next multiblock. The receiver calculates the syndrome of the received bits, which is the difference between the locally generated and the received parity. If the syndrome is zero, the received data bits are assumed to be correct. If the syndrome is non-zero, it can be used to determine the most likely error.*

**Table 4-5. CMD Parity Error Programming**

DAC JESD Page

| Register Address | Register Description |
|---|---|
| 0x1AB, bit[0] | JESDC_FEC_MODE |

### 4.2.10 End of Multi-block and End of Extended Multiblock Error (64B/66B and 64B/80B Encoding Specification Only)

1. EoMB Alignment Error: 5'b00001 not received at correct location in sync-header. Refer to JESD204C document, Table 47, bit[31:27] of the Sync word mapping for detail.
2. EoEMB Alignment Error: EoEMB not received at correct location in sync-header. Refer to JESD204C document,Table 47, bit[22] of the Sync word mapping.
3. EoEMB in the pilot signal is defined as:
   a. 1 if the current multiblock is the last multiblock of extended multiblock
   b. 0 if the current multiblock is not the last multiblock of the extended multiblock

**Table 4-6. Sync Word Mapping With Stand-Alone Command Channel**



| Bit | Function | Bit | Function | Bit | Function | Bit | Function |
|---|---|---|---|---|---|---|---|
| 0 | CMD[18] | 8 | CMD[12] | 16 | CMD[6] | 24 | CMD[2] |
| 1 | CMD[17] | 9 | CMD[11] | 17 | CMD[5] | 25 | CMD[1] |
| 2 | CMD[16] | 10 | CMD[10] | 18 | CMD[4] | 26 | CMD[0] |
| 3 | 1 | 11 | 1 | 19 | 1 | 27 | 0 |

**Table 4-6. Sync Word Mapping With Stand-Alone Command Channel (continued)**



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | CMD[15] | 12 | CMD[9] | 20 | CMD[3] | 28 | 0 |
| 5 | CMD[14] | 13 | CMD[8] | 21 | 1 | 29 | 0 |
| 6 | CMD[13] | 14 | CMD[7] | 22 | EoEMB | 30 | 0 |
| 7 | 1 | 15 | 1 | 23 | 1 | 31 | 1 |

**Table 4-7. Meaning of the CMD Sync Word Fields**



| Content | Description |
|---|---|
| CMD | Command Channel (see JESD204C, section 7.3.7.3) |
| 1,0,EoEMB | Pilot Signal (see JESD204C, section 7.3.3) |

### 4.2.11 Pilot Signal Error

Fixed Ones in pilot Signal Error: Fixed seven ones in CRC12/CRC3/FEC not received at correct location in sync-header

Refer to JESD204C document, Table 47, bit[3], bit[7] , bit[11] , bit[15] , bit[19] , bit[21] , bit[23] , bit[31] of the Sync word mapping for detail.

### 4.2.12 Error Rate Detection

For a lane rate of 32.5Gbps, in a normal scenario the BER would be $10^{-15}$. This would be one error in 8.5 hours. This error rate detection can trigger if there are two or more errors in one minute (BER of $10^{-11}$). The error can trigger link reset and the associated alarm can be raised.

### 4.2.13 Error Event Counter

The error detections can be counted in an 8-bit error counter:

1. Within a link, each lane has its own 8-bits error counter to read errors per lane.
2. The error counter saturates to decimal 255 and does not roll over.
3. Any of these (or) all of these errors can be enabled by a 16-bit ***error_ena[15:0]*** configuration register (per link). The error_counter is a cumulative of all the errors enabled.

**Table 4-8. Error Enable Notification**

DAC JESD Page

| Register Address | Register Name |
|---|---|
| 0xF0 | link0_error_ena[7:0] |
| 0xF1 | link0_error_ena[15:8] |
| 0xF2 | link1_error_ena[7:0] |
| 0xF3 | link1_error_ena[15:8] |

**Note**

The error enable registers select the errors generated are counted in the error count for the lanes. The bits also control what signals are sent out the SYNC pin (if configured) for error notification. Please note that sync requests take priority over the error notification.

Each of the bits for the above mentioned lane errors are mapped to Table 4-1.

1. To read the live error counter, a configuration register(per lane) ***LANEx_ERROR_EVENT_CNT_AUTOLOAD[15:0]*** need to be triggered from 1b'0 to 1b'1 to 1b'0 upon which the counter will be loaded into a read-only register ***LANEx_ERROR_EVENT_CNTy[7:0]***, where x is indicative of the lane number, and y is indicative of the error number.
2. The error counter will be cleared whenever the link is re-established or whenever the configuration register (per lane) ***LANEx_ERROR_EVENT_CNT_CLR[15:0]*** is triggered from programming bit transition of 1b'0 to 1b'1 to 1b'0 again.

**Table 4-9. Event Counter and Event Threshold Counter**

DAC JESD Page

| Register Address | Register Name | Register description | Event Counter or Event Threshold Counter |
|---|---|---|---|
| 0x110 to 0x117 | ***LANEx_ERROR_ THRESH_CNT_CLR[15:0]*** | These bits clear the error_thresh_counter for each of the errors mentioned in error_ena register for lanes | Event Threshold Counter |
| 0x118 to 0x11F | ***LANEx_ERROR_ EVENT_CNT_CLR[15:0]*** | These bits clear the error_event_counter for each of the errors mentioned in error_ena register for LANEx. | Event Counter |
| 0x120 to 0x127 | ***LANEx_ERROR_ EVENT_CNT_OUT_SEL[15:0]*** | These bit select either error_thresh_counter or error_event_counter to be read out for LANEx. | Event Counter and Event Threshold Counter |
| 0x128 to 0x12F | ***LANEx_ERROR_ EVENT_CNT_AUTOLOAD[15:0]*** | These bit loads the error_event_counters or error_thresh_counter into registers to be read through SPI for LANEx | Event Counter and Event Threshold Counter |
| 0x132 to 0x13F | ***LANEx_ERROR_ THRESH_CNTy*** | These bits sets threshold for each of the counters. | Event Threshold Counter |
| 0x155 to 0x183 | ***LANEx_ERROR_EVENT_CNTy[ 7:0]*** | 8-bit counter for the error event or error threshold count, depending on the LANEx_ERROR_EVENT_CNT_ OUT_SEL[15:0] setting. | Event Counter and Event Threshold Counter |

### 4.2.14 Error Threshold Counters

1. 4-bit Error Threshold Counters (***LANEx_ERROR_ THRESH_CNTy***) exists for each of the lane errors[10:0] mentioned for JESD204C standard.
2. These are for debug functionality only and does not impact the Lane Reset or PAP alarm
3. Sets a threshold for the count of each JESD error type. If the threshold is reached and if it is configured (programmable), the logic can pull the SYNC signal and reset the links.
4. Each of these 4-bit counters can be independently cleared/read through SPI.
5. These counters stop incrementing during link-reset or lane-reset.
6. These counters are saturated to decimal 15.
7. The following C functions will get the error count and clear the error count.

```
getJesdRxSyncErrorCnt
clearJesdRxSyncErrorCnt
```

## 4.3 Lane and Link Aggregation

- For JESD204C 64B/66B and 64B/80B encoding, the link establishment is done autonomously in the JESD204C Receiver (DAC Downlink Direction) without SYNC line interaction.
- Therefore the link aggregation is based on the programming of the lanes and link within the JESD204 receiver. This is the sync muxing setting in the AFE products.
- Within the JESD204 receiver, the data buffer release of all the lanes in a single link happen at the same time.
- The release occurs when all lanes in a link passes frame alignment in 8b/10b protocol
- The release occurs when all lanes in a link passes Multi Block alignment in 64B/66B or 64B/80B protocol.
- Any lane error in JESD204C lane will force the entire link to re-initialize

### Link Reset Functionality

1. Link-reset can be triggered through any of the above mentioned JESD204C lane errors[10:0] getting triggered.
2. Link-reset can also be triggered whenever any of the count of the individual lane errors[10:0] in the 4-bit Error Threshold Counters crosses a corresponding threshold set for that particular error.
3. Error Threshold Counters are reset on link-reset. Additionally, Error Threshold Counters can be reset periodically.
4. Each of the Error Threshold Counters can be independently read/cleared through SPI

### Link Reset Counter

1. An 8-bit Link reset counter, LANEx_RESET_COUNTER[7:0], exists which counts the number of times the link got reset.
2. Along with this counter, LANEx_RESET_ERRORS[15:0], which caused the last link-reset can also be read through SPI.
3. These are for debug functionality only and does not impact the Lane Reset or PAP alarm.
4. This counter can be independently cleared/read through SPI.
5. This counter is saturated to decimal 255.

DAC JESD Page

**Table 4-10. Lane Reset Counter and Lane Reset Error**

| Register Address | Register Name | Functionality |
|---|---|---|
| 0x144, 0x148, 0x14C, 0x150 | LANEx_RESET_ COUNTER[7:0] | read back of the lane reset counter which indicates how many times the lane got reset because of lane errors. |
| 0x146, 0x147, 0x14A, 0x14B, 0x14E, 0x14F, 0x152, 0x153 | LANEx_RESET_ERRORS[15:0] | read back of the lane errors which caused lane reset. |

## 4.4 Link Layer Testing

Per JESD204C Standard Section 8.4.8 (JESD204B Standard Section 5.3.3.8), the AFE77xxD family of devices supports link layer test modes for the 8B/10B encoding. The corresponding JESD204C Standard is in Section 8.4.8. Link Layer testing is not defined for the 64B/66B and 64B/80B encoding within the JESD204C Standard. Per the standards, the link layer test mode is a state where predetermined 8B/10B characters are transmitted in all frames and on all lanes on the multipoint link.

### 4.4.1 8B/10B JESD204C TX (ADC) Link Layer Testing

The JESD204C TX side of the ADC supports the test sequence transmission. Each of the JESD204B TX lanes can support the following sequences:

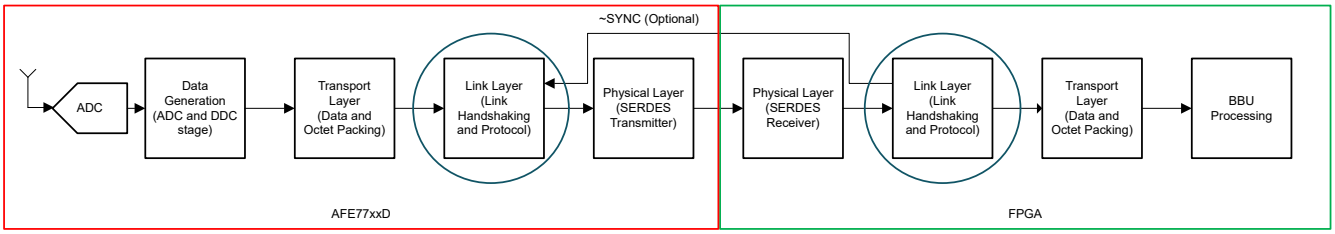


**Figure 4-6. Link Layer of ADC to Send Pattern to Link Layer of FPGA to Check**

**Table 4-11. JESD204C TX IP Test Sequence**

| Register Address | Register Description |
|---|---|
| 0xF7 | LINK0_JESD_TEST_SEQ_SEL[2:0] |
| 0x10F | LINK1_JESD_TEST_SEQ_SEL[2:0] |
| 0x127 | LINK2_JESD_TEST_SEQ_SEL[2:0] |

**Table 4-12. JESD204B TX Link Layer Test Sequence**

| TEST SEQUENCE NUMBER | TEST SEQUENCE NAME |
|---|---|
| 0 | Test Sequence Disabled |
| 1 | Repeat /D21.5/ high frequency pattern for random jitter (RJ) |
| 2 | Repeat /K28.5/ mixed frequency pattern for deterministic jitter (DJ) |
| 3 | Repeat initial lane alignment sequence (ILAS) |
| 4 | Modified random pattern (RPAT/CRPAT) |
| 5 | Scrambled jitter pattern (JSPAT) |
| 6 | Repeat /K28.7/ low frequency pattern |

The most commonly used sequence for the JESD204C TX test is the repeating /K28.5/ pattern and the repeating ILAS sequence. The repeating /K28.5/ pattern is useful in analyzing the correct pattern is send during the code group synchronization (CGS) state of the JESD204C 8B/10B handshaking protocol, and the repeating ILAS sequence is useful in debugging the process of initial lane alignment state of the protocol.

### 4.4.2 8B/10B JESD204C RX (DAC) Link Layer Testing

The JESD204C RX on the DAC side supports the test sequence pattern checking capability as described in the JESD204C standard. The standard specified the two main test sequences to be repeating /K28.5/ patterns and also repeating ILAS sequence. Other optional sequences will be covered in the 8B/10B decoder of the JESD204C RX IP. Each JESD204C RX link has an independent test sequence verifier, and each verifier has the following capabilities.

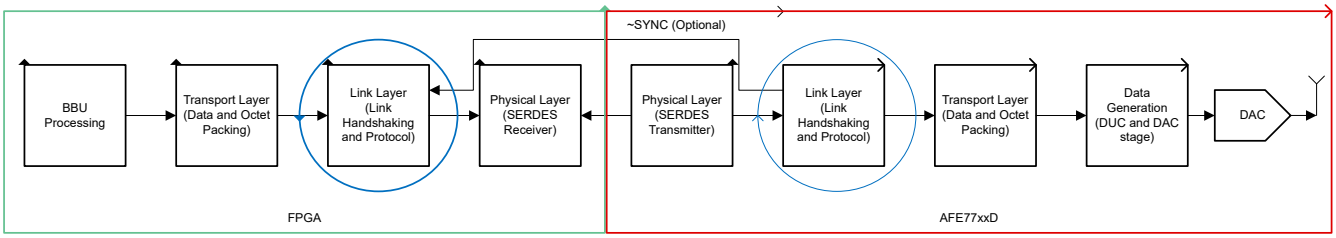


**Figure 4-7. Link Layer of FPGA to Send Pattern to Link Layer of DAC to Check**

Corresponding Registers (DAC JESD Page)

| JESD_TEST_SEQ |
|---|

**Table 4-13. JESD204B RX Link Layer Test Verifier**

| TEST SEQUENCE NUMBER | TEST SEQUENCE NAME |
|---|---|
| 0 | Test Sequence Disabled |

**Table 4-13. JESD204B RX Link Layer Test Verifier (continued)**

| TEST SEQUENCE NUMBER | TEST SEQUENCE NAME |
|---|---|
| 1 | Repeat /D21.5/ high frequency pattern for random jitter (RJ) |
| 2 | Repeat /K28.5/ mixed frequency pattern for deterministic jitter (DJ) |
| 3 | Repeat initial lane alignment sequence (ILAS) |

Besides configuring the test pattern verifier, the user will need to clear and read back certain alarm registers to confirm test result of the test pattern verifier. TI recommends the user first configure the test pattern generator on the JESD204B TX side (usually on the ASIC or FPGA), and then configure the test pattern verifier on the JESD204B RX side of the DAC. The clearing and reading of the test result should be done when both the JESD204B TX and JESD204B RX test pattern logics are stable.

## 4.5 RBD Optimization

The RBD optimization is a simple procedure for finding the optimal RBD in AFE77xxD JESD204C RX link for the DAC. The AFE77xxD JESD204C receiver block has unique features to read the skew and arrival of lanes with respect to LMFC/LEMC clock to help user find the right RBD for the system.

### 4.5.1

The JEDEC committee started the JESD204C standard to ensure compatibility over a common serial data interface for data converters and FPGA/ASICs. Even though different manufacturers may have different design implementations and clocking structures for various devices, the JESD204C standardized the data interface clock such that the serial data transferred will be processed in a common rate clock.

For the 8B/10B encoding system, the LMFC was standardized. For the 64B/66B or 64B/80B encoding system, the LEMC was standardized. To ensure that the processing clock LMFC or LEMC between the JESD204 TX and JESD204 RX are aligned at start-up of the system without drift or wonder, a global system reference clock (SYSREF) provides the clock synchronization and alignment. The SYSREF is distributed throughout the JESD204C system in a time aligned, fixed delay manner throughout various temperature cycle and system restart cycle. Since SYSREF is essentially deterministic, the data transfer through the JESD204C link will also be deterministic.

The JESD204C link is usually transferred over multiple lanes (aggregated as one link). With PCB trace delay and logic delays from the JESD204C TX to JESD204C RX, the lane to lane skew may contribute inconsistencies in the latency of the data transfer. To compensate for the lane-to-lane skew, the JESD204 Rx have internal buffer to first absorb the skews amongst all the lanes, and then re-align the lanes at the output of the buffer upon the release of the buffer. This essentially created a zero skew environment for data processing after the JESD204 Rx block. The buffer and the release of the buffer is controlled by RBD, or release buffer delay. Finding the optimal RBD value in a system that will work across various temperature and restart cycle is essential in the overall system stability.

The following is a common process of JESD204 link establishment and a typical process of setting the RBD

In the JESD204C Tx block, the start of any LMFC/LEMC shall be initiated simultaneously across all lanes at a well-defined moment in time. The ‘well-defined moment in time’ for TX transmission is a deterministic period of time from all active LMFC/LEMC edges.

In the JESD204C Rx block, to align data across lanes, a buffer exists to hold all lane data for release simultaneously at a well-defined moment in time. The ‘well-defined moment in time’ for RX buffer release is a programmable number of steps after an active LMFC/LEMC edge. This programmable number of steps is referred to as the RX Buffer Delay (RBD).

**Note**

The following are different time stamp indicator between different encoding systems to indicate measured skews amongst *Earliest Arrival Lane* and *Last Arrival Lane*.

- 8B/10B link: the difference between the times of earliest lane and latest lane for the bit transition between the last /K/ character to the first bit data or the start of the initial lane alignment sequence.
- 64B/66B or 64B/80B link: the difference between the times of earliest lane and latest lane for the bit transition in the sync header of the first block in a multiblock

In most applications, the RBD is set equal to ‘K’ for 8B/10B encoding or 'E' for other encoding to achieve deterministic latency. However for RBD=’K’ or 'E', the user has to exercise caution when the last arrival lane is close to the LMFC or LEMC edge. With physical lane skew varying across temperature or bringups, the RBD elastic buffers can be released in either current LMFC edge or next LMFC edge resulting in latency change of one LMFC period across multiple link-ups/device bringups. The same applies for LEMC.

RBD<K can be used to achieve the *lowest* deterministic latency. The user does not have to sweep the RBD from 0 to K-1 to determine the optimal RBD in AFE77xxD. The AFE77xxD device provides information on lane arrival, skew and RBD release opportunity via SPI readout to determine the right RBD value for given platform in few steps.

**LMFC Offset**

In AFE77xxD, LMFC/LEMC can be reset to a programmable register instead of a fixed *zero* value with respect to the sysref rising edge. The reset allows user to have a variable offset between TX LMFC and RX LMFC.

**Elastic Buffer:**

In AFE77xxD, the elastic buffer size in 204B and 204C mode is 14×64 octets.

### 4.5.2 Register Set

AFE77xxD internal counters operate at LaneRate/80 for 8b/10b encoding or LaneRate/66 for 64b/66b encoding.In the case of 8b/10b, the internal LMFC counter runs from 0 to (F×K/8)-1, and in 64b/66b, the internal LEMC counter runs from 0 to 32×E-1.

1. **link0/1_init_f_counter (R/W)**: On the rising edge of sysref, the internal frame_counter (8B/10B ENCODING) emb_counter (64B/66B ENCODING) can be loaded with init_f_counter value
2. **lane0/1/2/3_f_counter_any_lane_ready (R):** This is the arrival time of the multiframe or extended multiblock boundary relative to the active LMFC/LEMC **edge for the particular lane.** The unit for this would be LaneRate/80 (8B/10B encoding) and LaneRate/66 (64B/66B encoding).
3. **lane0/1/2/3_f_counter_all_lane_ready (R):** This is the arrival time of the multiframe or extended multiblock boundary relative to the active LMFC/LEMC edge for the slowest received lane. This value would be equal to the lane_f_counter_any_lane_ready for the slowest lane. The unit for this is LaneRate/80 (8B/10B encoding) and LaneRate/66 (64B/66B encoding).
4. **lane0/1/2/3_skew (R):** This is the difference between the arrival times of the multiframe or extended multiblock boundary between the fastest and slowest received lanes. The unit for this would be LaneRate/80 (8B/10B encoding) and LaneRate/66 (64B/66B encoding).
5. **link0_rbd_m1 (R/W):** The unit of RBD is also LaneRate/80 (4*octets) for 204B and LaneRate/66 for 204C.
6. **Link0_sysref_cnt_on_release_opportunity:** The JESD204 Rx IP and the Digital-Up-Converter (DUC) digital processing block have separate operating clock domains. To ensure optimal hand-off of the JESD204 Rx IP data (processed in the JESD_CLK domain of JESD204 Rx IP) to the DUC digital block (processed in the DUC_CLK domain of the DUC), a FIFO in between the clock domains is added. This FIFO write and read pointers are reset on the SYSREF rising edge, and have to be aligned with the JESD204 LMFC/LEFC clock to achieve deterministic latency. The “link0_sysref_cnt_on_release_opportunity” register is the difference between the write/read pointers and the clock edge of the LMFC/LEMC. To ensure optimal FIFO data hand-off between the two operating clock domains and also proper LMFC/LEMC edge alignment, the “link0_sysref_cnt_on_release_opportunity” should be read back as 0. The iterative approach on ensuring “link0_sysref_cnt_on_release_opportunity” is described in the RBD setting procedure.

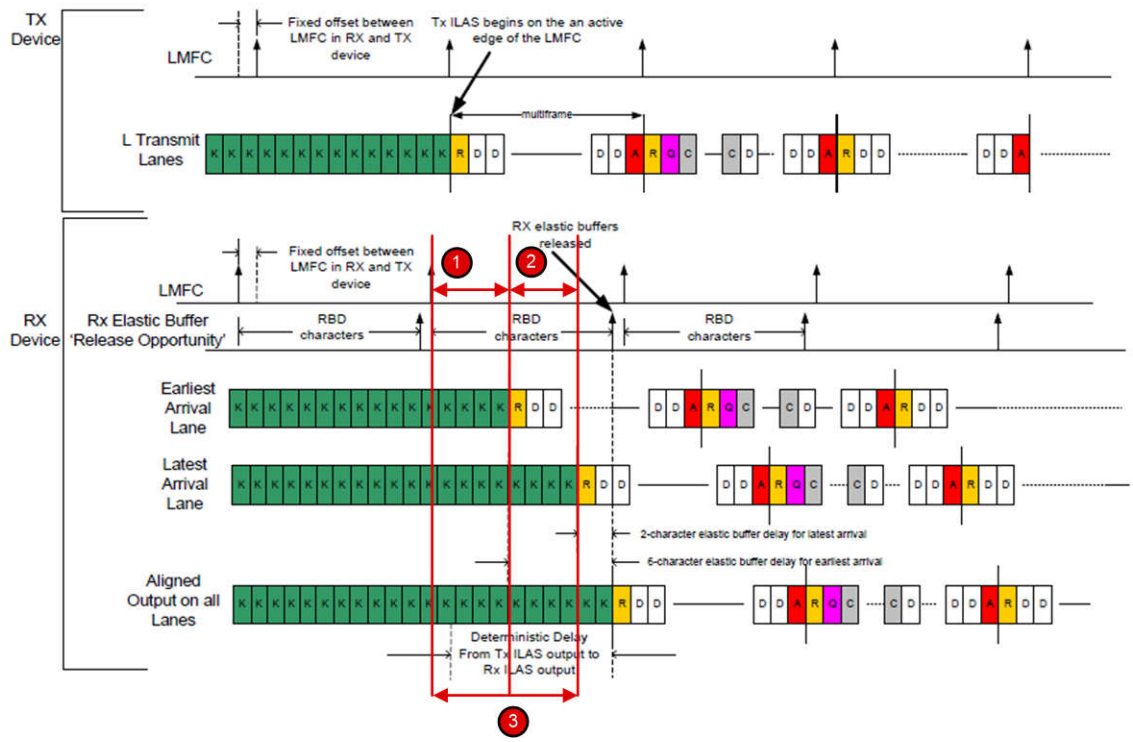


**Figure 4-8. F-Counter and Skew Reading in the Timing Diagram**

1. lane0/1/2/3_f_counter_any_lane_ready
2. lane0/1/2/3_skew
3. lane0/1/2/3_f_counter_all_lane_ready

### 4.5.3 Procedure to Set the Correct RBD

**Table 4-14. Setting RBD**

| **For 8b/10b Encoding Protocol** | |
|---|---|
| Step-1 | First run with some **link0_rbd_m1** register programmed |
| Step-2 | Read the **lane0_f_counter_all_lanes_ready** status register |
| Step-3 | Program the link0_rbd_m1 registers with the modulo((**lane0_f_counter_all_lanes_ready** + 4), F*K/8) and run. (offset of 4 is taken as additional lane skew margin) |
| Step-4 | Now read the **link0_sysref_cnt_on_release_opportunity** status register, if zero programmed **link0_rbd_m1 is the RBD, else move to step 5** |
| Step-5 | Again program the **link0_rbd_m1** registers with the **link0_rbd_m1** from Step-3 as modulo(('link0_rbd_m1 from Step-3' + link0_sysref_cnt_on_release_opportunity), F*K/8) and run |
| Step-6 | Now if you read back the register ' **link0_sysref_cnt_on_release_opportunity'**, it should read back as 0, the Programmed **link0_rbd_m1 is the RBD** |
| **For 64B/66B and 64B/80B Encoding Protocol** | |
| Step-1 | First run with some **link0_rbd_m1** register programmed |
| Step-2 | Read the **lane0_f_counter_all_lanes_ready** status register |
| Step-3 | Program the **link0_rbd_m1 registers** with the 'modulo((**lane0_f_counter_all_lanes_ready** ' + 4), 32*K) and run (offset of 4 is taken as additional skew margin) |
| Step-4 | Now read the ' **link0_sysref_cnt_on_release_opportunity**' status register, if zero programmed **link0_rbd_m1 is the RBD, else move to step 5** |
| Step-5 | Again program the **link0_rbd_m1** registers with the **link0_rbd_m1** from Step-3 as modulo(('**link0_rbd_m1** from Step-3' **+ link0_sysref_cnt_on_release_opportunity**), 32*K) and run |
| Step-6 | Now if you read back the register ' **link0_sysref_cnt_on_release_opportunity'**, it should read back as 0, the Programmed **link0_rbd_m1 is the RBD** |

**Example: 8B/10B ENCODING:**

For F=2 and K=32, internal lmfc_counter would run from 0 to F*K/8-1 = 0 to 15.

If the skew between the lanes is around 3 clocks of LaneRate/80, then

f_counter_any_lane_ready and f_counter_all_lane_ready will be read back as any of the given values

[f_counter_any_lane ready, f_counter_all_lane_ready] is the format, where f_counter_any_lane ready can be one of the multiple lanes in the link.

[0,3], [1,4], [2,5]……[12,15], [13,0], [14,1] and [15,2].

If they are [0,3], then we can set rbd_m1 as modulo(3+4,F*K/8) = modulo(7,16) = 7. (assuming sysref_cnt_on_release_opportunity as read back as 0).

If they are [12,15], then we can set rbd_m1 as modulo(15+4,F*K/8) = modulo(19,16) = 3. (assuming sysref_cnt_on_release_opportunity as read back as 0).

**Example: 64B/66B and 64B/80B ENCODING:**

For F=3 and E=3, internal lmfc_counter would run from 0 to 32*E-1 = 0 to 191.

If the skew between the lanes is around 3 clocks of LaneRate/66, then

f_counter_any_lane_ready and f_counter_all_lane_ready will be read back as any of the given values

[0,3], [1,4], [2,5]……[188,191], [189,0], [190,1], [191,0].

If they are [189,0], then we can set rbd_m1 as modulo(0+4,32*E) = modulo(4,192) = 4. (assuming sysref_cnt_on_release_opportunity as read back as 0).

If they are [188,191], then we can set rbd_m1 as modulo(191+4,32*E) = modulo(195,192) = 3. (assuming sysref_cnt_on_release_opportunity as read back as 0).

#### *4.5.3.1 RBD Adjustment C Functions*

```
getAllLaneReady()
```

This function reads the all lane ready counter which is the offset between the internal LMFC boundary and the multiframe boundary (in 8B/10B encoding) or extended multi block boundary (in 64B/66B or 64B/80B Encoding) of the last lane of arrival. This value after an offset (for example, offset of two) with modulus of F*K/8 (in 8B/10B encoding) or with modulus of 32*E (in 64B/66B encoding) should be writen to the RBD register.

```
getGoodRbdRange()
```

This function returns the minimum and maximum good RBD range.

```
checkIfRbdIsGood()
```

Checks if the currently set RBD value is optimized or not.

```
setGoodRbd()
```

This function does the following:

1. Reads the internal counter between the internal LMFC counter to the received LMFC boundary (multiframe boundary in 8B/10B and Extended Multiblock Boundary in other encoding).
2. Sets the RBD by giving an offset of four to the LMFC Counter.

After executing this function a relink process is necessary, and therefore all the related conditions of adcDacSync() function are to be satisfied here, too. Note that this sequence may not ensure deterministic latency across bring-ups amongst multiple AFE77xxD devices.

To achieve deterministic latency, the necessary functions getAllLaneReady(), setManualRbd(), adcDacSync() with external pin SYSREF. The getAllLaneReady() should be done only once and the same value should be loaded each time during the intialization in order to ensure the same RBD is used throughout multiple start-ups. This particular RBD value may also be an input as a parameter to Latte.

```
setManualRbd()
```

Manually sets the RBD value of the link.

# 5 Transport Layer

## 5.1 Transport Layer Data Packing

### 5.1.1 JESD204C Frame Assembly for RX ADCs

The receiver channel outputs are output in complex I/Q pair mode. The following tables list the JESD204C transport layer frame mapping for each mode with complex I/Q pair outputs.

In the frame format tables, xRXIN_iy_sz [msb:lsb] and xRXIN_qy_sz [msb:lsb] are bits msb through lsb of the I stream sample, Q stream samples at time z of DDC channel n of the ADC *x*, respectively, where *x* = {1,2,3,4,5,6,7,8},
y = {0, 1} and z = {0, 1}.

For example, 2RXIN_i0_s0 [15:0] are 16-bit data of the I stream sample at time 0 of the DDC channel 0 of 2RXIN.

**Table 5-1. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 2-4-4-1-0 (Base Link for 2RXIN, Complex Mode)**

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1RXIN_i0[15:0] | | 1RXIN_q0[15:0] | |
| | | Lane STX2 | 2RXIN_i0[15:0] | | 2RXIN_q0[15:0] | |
| | | Lane STX3 | 3RXIN_i0[15:0] | | 3RXIN_q0[15:0] | |
| | | Lane STX4 | 4RXIN_i0[15:0] | | 4RXIN_q0[15:0] | |
| | Link2 | Lane STX5 | 5RXIN_i0[15:0] | | 5RXIN_q0[15:0] | |
| | | Lane STX6 | 6RXIN_i0[15:0] | | 6RXIN_q0[15:0] | |
| | | Lane STX7 | 7RXIN_i0[15:0] | | 7RXIN_q0[15:0] | |
| | | Lane STX8 | 8RXIN_i0[15:0] | | 8RXIN_q0[15:0] | |

Base Link for 2xRXIN with equivalent L-M-F-S-Hd of 2-4-4-1-0

Single Link 1x8xRXIN with equivalent L-M-F-S-Hd of 8-16-4-1-0

Dual Link 2x4xRXIN with equivalent L-M-F-S-Hd of 4-8-4-1-0

**Table 5-2. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 1-4-8-1-0 (Base Link for 2xRXIN, Complex Mode)**

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1RXIN_i0[15:0] | | 1RXIN_q0[15:0] | | 2RXIN_i0[15:0] | | 2RXIN_q0[15:0] | |
| | | Lane STX2 | 3RXIN_i0[15:0] | | 3RXIN_q0[15:0] | | 4RXIN_i0[15:0] | | 4RXIN_q0[15:0] | |
| | Link2 | Lane STX3 | 5RXIN_i0[15:0] | | 5RXIN_q0[15:0] | | 6RXIN_i0[15:0] | | 6RXIN_q0[15:0] | |
| | | Lane STX4 | 7RXIN_i0[15:0] | | 7RXIN_q0[15:0] | | 8RXIN_i0[15:0] | | 8RXIN_q0[15:0] | |

Base Link for 2xRXIN with equivalent L-M-F-S-Hd of 1-4-8-1-0

Single Link for 1x8xRXIN with equivalent L-M-F-S-Hd of 4-16-8-1-0

Dual Link for 2x4xRXIN with equivalent L-M-F-S-Hd of 2-8-8-1-0

**Table 5-3. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 2-4-6-1-0 (Base Link for 2xRXIN, Complex Mode)**

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| Link 1 | Link1 | Lane STX1 | 1RXIN_i0 [15:0] | | Zeros[7:0] | 1RXIN_q0 [15:0] | | Zeros[7:0] |
| | | Lane STX2 | 2RXIN_i0 [15:0] | | Zeros[7:0] | 2RXIN_q0 [15:0] | | Zeros[7:0] |
| | | Lane STX3 | 3RXIN_i0 [15:0] | | Zeros[7:0] | 3RXIN_q0 [15:0] | | Zeros[7:0] |
| | | Lane STX4 | 4RXIN_i0 [15:0] | | Zeros[7:0] | 4RXIN_q0 [15:0] | | Zeros[7:0] |
| | Link2 | Lane STX5 | 5RXIN_i0 [15:0] | | Zeros[7:0] | 5RXIN_q0 [15:0] | | Zeros[7:0] |
| | | Lane STX6 | 6RXIN_i0 [15:0] | | Zeros[7:0] | 6RXIN_q0 [15:0] | | Zeros[7:0] |
| | | Lane STX7 | 7RXIN_i0 [15:0] | | Zeros[7:0] | 7RXIN_q0 [15:0] | | Zeros[7:0] |
| | | Lane STX8 | 8RXIN_i0 [15:0] | | Zeros[7:0] | 8RXIN_q0 [15:0] | | Zeros[7:0] |

Base Link for 2xRXIN with equivalent L-M-F-S-Hd = 2-4-6-1-0

Single Link for 1x8xRXIN with equvialent L-M-F-S-Hd = 8-16-6-1-0

Dual Link for 2x4xRXIN with equivalent L-M-F-S-Hd = 4-8-6-1-0

**Table 5-4. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 1-8-16-1-0 (Base Link for 4xRXIN, Complex Mode)**

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1RXIN_i0[15:0] | | 1RXIN_q0[15:0] | | 2RXIN_i0[15:0] | | 2RXIN_q0[15:0] | | 3RXIN_i0[15:0] | | 3RXIN_q0[15:0] | | 4RXIN_i0[15:0] | | 4RXIN_q0[15:0] | |
| | Link2 | Lane STX2 | 5RXIN_i0[15:0] | | 5RXIN_q0[15:0] | | 6RXIN_i0[15:0] | | 6RXIN_q0[15:0] | | 7RXIN_i0[15:0] | | 7RXIN_q0[15:0] | | 8RXIN_i0[15:0] | | 8RXIN_q0[15:0] | |

Base Link for 4xRXIN with equivalent L-M-F-S-Hd of 1-8-16-1-0

Single Link for 1x8xRXIN with equivalent L-M-F-S-Hd of 2-16-16-1-0

Dual Link for 2x4xRXIN with equivalent L-M-F-S-Hd of 1-8-16-1-0

**Table 5-5. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 1-4-12-1-0 (Base Link for 2xRXIN, Complex Mode)**

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1RXIN_i0[15:0] | | Zeros[7:0] | 1RXIN_q0[15:0] | | Zeros[7:0] | 2RXIN_i0[15:0] | | Zeros[7:0] | 2RXIN_q0[15:0] | | Zeros[7:0] |
| | | Lane STX2 | 3RXIN_i0[15:0] | | Zeros[7:0] | 3RXIN_q0[15:0] | | Zeros[7:0] | 4RXIN_i0[15:0] | | Zeros[7:0] | 4RXIN_q0[15:0] | | Zeros[7:0] |
| | Link2 | Lane STX3 | 5RXIN_i0[15:0] | | Zeros[7:0] | 5RXIN_q0[15:0] | | Zeros[7:0] | 6RXIN_i0[15:0] | | Zeros[7:0] | 6RXIN_q0[15:0] | | Zeros[7:0] |
| | | Lane STX4 | 7RXIN_i0[15:0] | | Zeros[7:0] | 7RXIN_q0[15:0] | | Zeros[7:0] | 8RXIN_i0[15:0] | | Zeros[7:0] | 8RXIN_q0[15:0] | | Zeros[7:0] |

Base Link for 2xRXIN with equivalent L-M-F-S-Hd of 1-4-12-1-0

Single Link for 1x8xRXIN with equivalent L-M-F-S-Hd of 4-16-12-1-0

Dual Link for 2x4xRXIN with equivalent L-M-F-S-Hd of 2-8-12-1-0

**Table 5-6. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 4-4-3-1-0 (Base Link for 2xRXIN, Complex Mode)**

| Single Link | Dual Link | OCTET | 1 | 2 | 3 |
|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1RXIN_i0[15:0] | | Zeros[7:0] |
| | | Lane STX2 | 1RXIN_q0[15:0] | | Zeros[7:0] |
| | | Lane STX3 | 2RXIN_i0[15:0] | | Zeros[7:0] |
| | | Lane STX4 | 2RXIN_q0[15:0] | | Zeros[7:0] |
| | Link2 | Lane STX5 | 3RXIN_i0[15:0] | | Zeros[7:0] |
| | | Lane STX6 | 3RXIN_q0[15:0] | | Zeros[7:0] |
| | | Lane STX7 | 4RXIN_i0[15:0] | | Zeros[7:0] |
| | | Lane STX8 | 4RXIN_q0[15:0] | | Zeros[7:0] |

Base Link for 2xRXIN with equivalent L-M-F-S-Hd of 4-4-3-1-0

Single Link for 1x4xRXIN with equivalent L-M-F-S-Hd of 8-8-3-1-0

Dual Link for 2x2xRXIN with equivalent L-M-F-S-Hd of 4-4-3-1-0

### 5.1.2 JESD204C Frame Assembly for FB ADCs

The feedback receiver channel outputs are output in complex I/Q pair mode. The following tables list the JESD204C transport layer frame mapping for each mode with complex I/Q pair outputs.

In the frame format tables, xFBIN_iy_sz [msb:lsb] and xFBIN_qy_sz [msb:lsb] are bits msb through lsb of the I stream sample, Q stream samples at time z of DDC channel n of the ADC *x*, respectively, where *x* = {1,2,3,4,5,6,7,8},
y = {0, 1} and z = {0, 1}.

For example, 2FBIN_i0_s0 [15:0] are 16-bit data of the I stream sample at time 0 of the DDC channel 0 of 2FBIN.

**Table 5-7. JESD204C Frame Format for FB ADCs: L-M-F-S-Hd = 1-2-8-2-0 (Base Link for 1xFBIN, Complex Mode)**

Base Link for 1xFBIN with equivalent L-M-F-S-Hd of 1-2-8-2-0

Single Link for 1x2xFBIN with equivalent L-M-F-S-Hd of 2-4-8-2-0

Dual Link for 2x1xFBIN with equivalent L-M-F-S-Hd of 1-2-8-2-0

Paired JESD204C Mode for TDD Switching RX ADC Mode of 1-4-8-1-0 per

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1FBIN_i0_s0[15:0] | | 1FBIN_i0_s1[15:0] | | 1FBIN_q0_s0[15:0] | | 1FBIN_q1_s1[15:0] | |
| | Link2 | Lane STX3 | 2FBIN_i0_s0[15:0] | | 2FBIN_i0_s1[15:0] | | 2FBIN_q0_s0[15:0] | | 2FBIN_q1_s1[15:0] | |

FB ADC mode of 1-2-8-2-0 has a paired JESD204C Mode for TDD Switching RX ADC Mode of 1-4-8-1-0

**Table 5-8. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 2-2-6-2-0 (Base Link for 1xFBIN, Complex Mode)**

Base Link for 1xFBIN with equivalent L-M-F-S-Hd of 2-2-6-2-0

Single Link for 1x2xFBIN with equivalent L-M-F-S-Hd of 4-4-6-2-0

Dual Link for 2x1xFBIN with equivalent L-M-F-S-Hd of 2-2-6-2-0

Paired JESD204C Mode for TDD Switching RX ADC Mode of 2-4-6-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1FBIN_i0_s0[15:0] | | Zeros[7:0] | 1FBIN_i0_s1[15:0] | | Zeros[7:0] |
| | | Lane STX2 | 1FBIN_q0_s0[15:0] | | Zeros[7:0] | 1FBIN_q0_s1[15:0] | | Zeros[7:0] |
| | Link2 | Lane STX3 | 2FBIN_i0_s0[15:0] | | Zeros[7:0] | 2FBIN_i0_s1[15:0] | | Zeros[7:0] |
| | | Lane STX4 | 2FBIN_q0_s0[15:0] | | Zeros[7:0] | 2FBIN_q0_s1[15:0] | | Zeros[7:0] |

FB ADC mode of 2-2-6-2-0 has a paired JESD204C Mode for TDD Switching RX ADC Mode of 2-4-6-1-0

**Table 5-9. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 1-4-16-2-0 (Base Link for 2xFBIN, Complex Mode)**

Base Link for 2xFBIN with equivalent L-M-F-S-Hd of 1-4-16-2-0

Single Link for 1x2xFBIN with equivalent L-M-F-S-Hd of 1-4-16-2-0

Paired JESD204C Mode for TDD Switching RX ADC Mode of 1-16-16-1-0

| Single Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Lane STX1 | 1FBIN_i0_s0[15:0] | | 1FBIN_q0_s0[15:0] | | 1FBIN_i0_s1[15:0] | | 1FBIN_q0_s1[15:0] | | 2FBIN_i0_s0[15:0] | | 2FBIN_q0_s0[15:0] | | 2FBIN_i0_s1[15:0] | | 2FBIN_q0_s1[15:0] | |

FB ADC mode of 1-4-16-2-0 has a paired JESD204C Mode for TDD Switching RX ADC Mode of 1-8-16-1-0

**Table 5-10. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 1-2-12-2-0 (Base Link for 2xFBIN, Complex Mode)**

Base Link for 2xRXIN with equivalent L-M-F-S-Hd of 1-2-12-2-0

Single Link for 1x8xRXIN with equivalent L-M-F-S-Hd of 2-4-12-2-0

Dual Link for 2x4xRXIN with equivalent L-M-F-S-Hd of 1-2-12-2-0

Paired JESD204C Mode for TDD Switching RX ADC Mode of 1-4-12-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1FBIN_i0_s0[15:0] | | Zeros[7:0] | 1FBIN_q0_s0[15:0] | | Zeros[7:0] | 1FBIN_i0_s1[15:0] | | Zeros[7:0] | 1FBIN_q0_s1[15:0] | | Zeros[7:0] |
| | Link2 | Lane STX2 | 2FBIN_i0_s0[15:0] | | Zeros[7:0] | 2FBIN_q0_s0[15:0] | | Zeros[7:0] | 2FBIN_i0_s1[15:0] | | Zeros[7:0] | 2FBIN_q0_s1[15:0] | | Zeros[7:0] |

FB ADC mode of 1-2-12-2-0 has a paired JESD204C Mode for TDD Switching RX ADC Mode of 1-4-12-1-0

**Table 5-11. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 4-2-3-2-0 (Base Link for 1xFBIN, Complex Mode)**

Base Link for 1xFBIN with equivalent L-M-F-S-Hd of 4-2-3-2-0

Single Link for 1x2xFBIN with equivalent L-M-F-S-Hd of 8-4-3-2-0

Dual Link for 2x1xFBIN with equivalent L-M-F-S-Hd of 4-2-3-2-0

Paired JESD204C Mode for TDD Switching RX ADC Mode of 4-4-3-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 |
|---|---|---|---|---|---|
| Link1 | Link1 | Lane STX1 | 1FBIN_i0_s0[15:0] | | Zeros[7:0] |
| | | Lane STX2 | 1FBIN_q0_s0[15:0] | | Zeros[7:0] |
| | | Lane STX3 | 1FBIN_i0_s1[15:0] | | Zeros[7:0] |
| | | Lane STX4 | 1FBIN_q0_s1[15:0] | | Zeros[7:0] |
| | Link2 | Lane STX5 | 2FBIN_i0_s0[15:0] | | Zeros[7:0] |
| | | Lane STX6 | 2FBIN_q0_s0[15:0] | | Zeros[7:0] |
| | | Lane STX7 | 21FBIN_i0_s1[15:0] | | Zeros[7:0] |
| | | Lane STX8 | 2FBIN_q0_s1[15:0] | | Zeros[7:0] |

FB ADC mode of 4-2-3-2-0 has a paired JESD204C Mode for TDD Switching RX ADC Mode of 4-4-3-1-0

**Table 5-12. JESD204C Frame Format for RX ADCs: L-M-F-S-Hd = 1-2-16-2-0 (Base Link for 1xFBIN, Complex Mode)**

Base Link for 1xFBIN with equivalent L-M-F-S-Hd of 1-2-16-2-0

Single Link for 1x1xFBIN with equivalent L-M-F-S-Hd of 1-2-16-2-0

Paired JESD204C Mode for TDD Switching RX ADC Mode of 1-16-16-1-0

| Single Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Lane STX1 | 1FBIN_i0_s0[15:0] | | 1FBIN_i0_s1[15:0] | | 1FBIN_i0_s2[15:0] | | 1FBIN_i0_s3[15:0] | | 1FBIN_q0_s0[15:0] | | 1FBIN_q0_s1[15:0] | | 1FBIN_q0_s2[15:0] | | 1FBIN_q0_s3[15:0] | |

### 5.1.3 JESD204C Frame Assembly for DACs

The transmitter channel inputs are in complex I/Q pair mode. The following tables list the JESD204C transport layer frame mapping for each mode with complex I/Q pair outputs.

The following tables list the frame formats for each mode.

In the frame format tables, xTX_iy_sz [msb:lsb] and xTX_qy_sz [msb:lsb] are bits msb through lsb of the I and Q samples at time z of DUC channel n of the DAC x, where x = {1,2,3,4,5,6,7,8}, y = {0, 1} and z = {0, 1}.

For example: 1TXOUT_i0_s0 [15:4] are 12-bit MSB data of the I stream sample at time 0 of the DUC channel 0 of 1TXOUT.

**Table 5-13. JESD204C Frame Format for DACs: L-M-F-S-Hd = 2-4-4-1-0 (Base Link for 2xTXOUT, Complex Mode)**

Base Link for 2xTXOUT with equivalent L-M-F-S-Hd of 2-4-4-1-0

Single Link for 1x8xTXOUT with equivalent L-M-F-S-Hd of 8-16-4-1-0

Dual Link for 2x4xTXOUT with equivalent L-M-F-S-Hd of 4-8-4-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane SRX1 | 1TXOUT_i0[15:0] | | 1TXOUT_q0[15:0] | |
| | | Lane SRX2 | 2TXOUT_i0[15:0] | | 2TXOUT_q0[15:0] | |
| | | Lane SRX3 | 3TXOUT_i0[15:0] | | 3TXOUT_q0[15:0] | |
| | | Lane SRX4 | 4TXOUT_i0[15:0] | | 4TXOUT_q0[15:0] | |
| | Link2 | Lane SRX5 | 5TXOUT_i0[15:0] | | 5TXOUT_q0[15:0] | |
| | | Lane SRX6 | 6TXOUT_i0[15:0] | | 6TXOUT_q0[15:0] | |
| | | Lane SRX7 | 7TXOUT_i0[15:0] | | 7TXOUT_q0[15:0] | |
| | | Lane SRX8 | 8TXOUT_i0[15:0] | | 8TXOUT_q0[15:0] | |

**Table 5-14. JESD204C Frame Format for DACs: L-M-F-S-Hd = 2-4-3-1-0 (Base Link for 2xTXOUT, Complex Mode)**

Base Link for 2xTXOUT with equivalent L-M-F-S-Hd = 2-4-3-1-0

Single Link for 1x8xTXOUT with equvialent L-M-F-S-Hd = 8-16-3-1-0

Dual Link for 2x4xTXOUT with equivalent L-M-F-S-Hd = 4-8-3-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 |
|---|---|---|---|---|---|
| Link 1 | Link1 | Lane SRX1 | 1TXOUT_i0 [15:4] | | 1TXOUT_q0 [15:4] |
| | | Lane SRX2 | 2TXOUT_i0 [15:4] | | 2TXOUT_q0 [15:4] |
| | | Lane SRX3 | 3TXOUT_i0 [15:4] | | 3TXOUT_q0 [15:4] |
| | | Lane SRX4 | 4TXOUT_i0 [15:4] | | 4TXOUT_q0 [15:4] |
| | Link2 | Lane SRX5 | 5TXOUT_i0 [15:4] | | 5TXOUT_q0 [15:4] |
| | | Lane SRX6 | 6TXOUT_i0 [15:4] | | 6TXOUT_q0 [15:4] |
| | | Lane SRX7 | 7TXOUT_i0 [15:4] | | 7TXOUT_q0 [15:4] |
| | | Lane SRX8 | 8TXOUT_i0 [15:4] | | 8TXOUT_q0 [15:4] |

**Table 5-15. JESD204C Frame Format for DACs: L-M-F-S-Hd = 1-4-8-1-0 (Base Link for 2xTXOUT, Complex Mode)**

Base Link for 2xTXOUT with equivalent L-M-F-S-Hd of 1-4-8-1-0

Single Link for 1x8xTXOUT with equivalent L-M-F-S-Hd of 4-16-8-1-0

Dual Link for 2x4xTXOUT with equivalent L-M-F-S-Hd of 2-8-8-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane SRX1 | 1TXOUT_i0[15:0] | | 1TXOUT_q0[15:0] | | 2TXOUT_i0[15:0] | | 2TXOUT_q0[15:0] | |
| | | Lane SRX2 | 3TXOUT_i0[15:0] | | 3TXOUT_q0[15:0] | | 4TXOUT_i0[15:0] | | 4TXOUT_q0[15:0] | |
| | Link2 | Lane SRX3 | 5TXOUT_i0[15:0] | | 5TXOUT_q0[15:0] | | 6TXOUT_i0[15:0] | | 6TXOUT_q0[15:0] | |
| | | Lane SRX4 | 7TXOUT_i0[15:0] | | 7TXOUT_q0[15:0] | | 8TXOUT_i0[15:0] | | 8TXOUT_q0[15:0] | |

**Table 5-16. JESD204C Frame Format for DACs: L-M-F-S-Hd = 1-4-6-1-0 (Base Link for 2xTXOUT, Complex Mode)**

Base Link for 2xTXOUT with equivalent L-M-F-S-Hd of 1-4-6-1-0

Single Link for 1x8xTXOUT with equivalent L-M-F-S-Hd of 4-16-6-1-0

Dual Link for 2x4xTXOUT with equivalent L-M-F-S-Hd of 2-8-6-1-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| Link1 | Link1 | Lane SRX1 | 1TXOUT_i0[15:4] | 1TXOUT_q0[15:4] | | 2TXOUT_i0[15:4] | 2TXOUT_q0[15:4] | |
| | | Lane SRX2 | 3TXOUT_i0[15:4] | 3TXOUT_q0[15:4] | | 4TXOUT_i0[15:4] | 4TXOUT_q0[15:4] | |
| | Link2 | Lane SRX3 | 5TXOUT_i0[15:4] | 5TXOUT_q0[15:4] | | 6TXOUT_i0[15:4] | 6TXOUT_q0[15:4] | |
| | | Lane SRX4 | 7TXOUT_i0[15:4] | 7TXOUT_q0[15:4] | | 8TXOUT_i0[15:4] | 8TXOUT_q0[15:4] | |

**Table 5-17. JESD204C Frame Format for DACs: L-M-F-S-Hd = 4-4-2-1-0 (Base Link for 2xTXOUT, Complex Mode)**

Base Link for 2xTXOUT with equivalent L-M-F-S-Hd of 4-4-2-1-0

Single Link for 1x4xTXOUT with equivalent L-M-F-S-Hd of 8-8-2-1-0

Dual Link for 2x2xTXOUT with equivalent L-M-F-S-Hd of 4-4-2-1-0

| Single Link | Dual Link | OCTET | 1 | 2 |
|---|---|---|---|---|
| Link1 | Link1 | Lane SRX1 | 1TX_i0[15:0] | |
| | | Lane SRX2 | 1TX_q0[15:0] | |
| | | Lane SRX3 | 2TX_i0[15:0] | |
| | | Lane SRX4 | 2TX_q0[15:0] | |
| | Link2 | Lane SRX5 | 3TX_i0[15:0] | |
| | | Lane SRX6 | 3TX_q0[15:0] | |
| | | Lane SRX7 | 4TX_i0[15:0] | |
| | | Lane SRX8 | 4TX_q0[15:0] | |

**Table 5-18. JESD204C Frame Format for DACs: L-M-F-S-Hd = 4-4-3-2-0 (Base Link for 2xTXOUT, Complex Mode)**

Base Link for 2xTXOUT with equivalent L-M-F-S-Hd of 4-4-3-2-0

Single Link for 1x4xTXOUT with equivalent L-M-F-S-Hd of 8-8-3-2-0

Dual Link for 2x2xTXOUT with equivalent L-M-F-S-Hd of 4-4-3-2-0

| Single Link | Dual Link | OCTET | 1 | 2 | 3 |
|---|---|---|---|---|---|
| Link1 | Link1 | Lane SRX0 | 1TX_i0_s0 [15:4] | | 1TX_i0_s1 [15:4] |
| | | Lane SRX1 | 1TX_q0_s0 [15:4] | | 1TX_q0_s1 [15:4] |
| | Link2 | Lane SRX2 | 2TX_i0_s0 [15:4] | | 2TX_i0_s1 [15:4] |
| | | Lane SRX3 | 2TX_q0_s0 [15:4] | | 2TX_q0_s1 [15:4] |

**Table 5-19. JESD204C Frame Format for DACs: L-M-F-S-Hd = 4-2-1-1-1 (Base Link for 1xTXOUT, Complex Mode)**

Base Link for 1xTXOUT with equivalent L-M-F-S-Hd of 4-2-1-1-1

Single Link for 1x2xTXOUT with equivalent L-M-F-S-Hd of 8-4-1-1-1

Dual Link for 2x1xTXOUT with equivalent L-M-F-S-Hd of 4-2-1-1-1

| Single Link | Dual Link | OCTET | 1 |
|---|---|---|---|
| Link1 | Link1 | Lane SRX0 | 1TX_i0[15:8] |
| | | Lane SRX1 | 1TX_i0[7:0] |
| | | Lane SRX2 | 1TX_q0[15:8] |
| | | Lane SRX3 | 1TX_q0[7:0] |
| | Link2 | Lane SRX4 | 2TX_i0[15:8] |
| | | Lane SRX5 | 2TX_i0[7:0] |
| | | Lane SRX6 | 2TX_q0[15:8] |
| | | Lane SRX7 | 2TX_q0[7:0] |

## 5.2 Transport Layer Testing

Per JESD204B Standard Section 5.1.6.2, the AFE77xxD family of devices supports short transport layer test pattern generator and verifier. Per the standard, the short test pattern has duration of one frame period and is repeated continuously for the duration of the test. Each sample shall have a unique value that can be identified with the position of the sample in the user data format.

### 5.2.1 JESD204C TX on the ADC Side

The JESD204C TX side of the ADC supports the test sequence transmission. The test sequence generator is enabled through the configuration of the SPI registers.

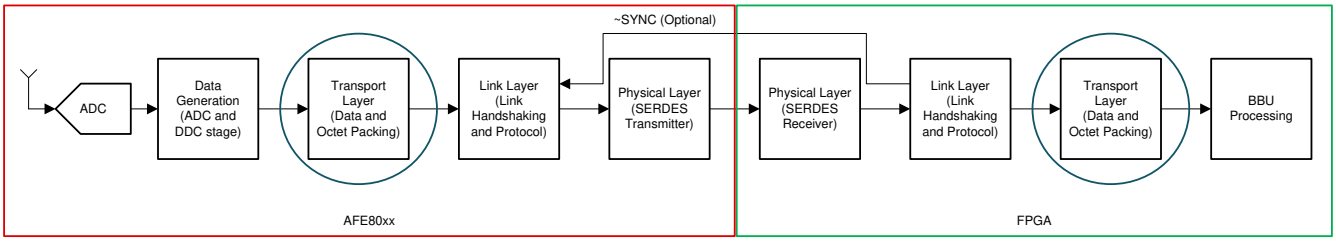


**Figure 5-1. Transport Layer of ADC to Send Check Pattern to Link Layer of FPGA**

The mapping of the programming patterns is done in octets, and the mapping of the programmable octets in the SPI register and to the actual transport layer pattern is done in one to one fashion. Also, since the JESD204 TX programming groups either ADC-A and ADC-B together or ADC-C and ADC-D together, the transport layer packing is addressed in two ADC channel at the same time. For example, the 2RX of 24410 transport layer is shown in Table 5-20, while the programmable octets in short pattern test is shown in Table 5-21.

| | | |
|---|---|---|
| 189h | ramptest_incr_a | jesd_test_sig_gen_mode_a |
| 18Ah | ramptest_incr_c | jesd_test_sig_gen_mode_c |
| 18Bh | ramptest_incr_fb | jesd_test_sig_gen_mode_fb |
| 18Ch | ramptest_incr_b | jesd_test_sig_gen_mode_b |
| 18Dh | ramptest_incr_d | jesd_test_sig_gen_mode_d |

| | |
|---|---|
| 190h | jesd_short_test_pattern_input0 |
| 191h | jesd_short_test_pattern_input0 |
| 192h | jesd_short_test_pattern_input1 |
| 193h | jesd_short_test_pattern_input1 |

| | |
|---|---|
| 194h | jesd_short_test_pattern_input2 |
| 195h | jesd_short_test_pattern_input2 |
| 196h | jesd_short_test_pattern_input3 |
| 197h | jesd_short_test_pattern_input3 |
| 198h | jesd_short_test_pattern_input4 |
| 199h | jesd_short_test_pattern_input4 |
| 19Ah | jesd_short_test_pattern_input5 |
| 19Bh | jesd_short_test_pattern_input5 |
| 19Ch | jesd_short_test_pattern_input6 |
| 19Dh | jesd_short_test_pattern_input6 |
| 19Eh | jesd_short_test_pattern_input7 |
| 19Fh | jesd_short_test_pattern_input7 |

**Table 5-20. Transport Layer Mapping for 2RX of 24410 Mode**

| EXAMPLE | JESD204B TX 1x2RX_24410 MODE | | | |
|---|---|---|---|---|
| Octet | 1 | 2 | 3 | 4 |
| lane STX0 | RXA_i0[15:8] | RXA_i0[7:0] | RXA_q0[15:8] | RXA_q0[7:0] |
| lane STX1 | RXB_i0[15:8] | RXB_i0[7:0] | RXB_q0[15:8] | RXB_q0[7:0] |

**Table 5-21. Mapping of Transport Layer Octet to 24410 Mode**

| EXAMPLE | JESD204B TX 1x2RX_24410 MODE | | | |
|---|---|---|---|---|
| Octet | 1 | 2 | 3 | 4 |
| lane STX0 | Octet0 | Octet1 | Octet2 | Octet3 |
| lane STX1 | Octet4 | Octet5 | Octet6 | Octet7 |

Other transport layer pattern arrangement may also have the same octet mapping. For example, simple arrangement of the transport layer data packing shown that the same pattern for 1x2RX of 24410 is also valid for 1x2RX of 44210 mode, as shown in Table 5-22 and Table 5-23.

**Table 5-22. Transport Layer Mapping for 2RX of 44210 Mode**

| EXAMPLE | JESD204B X 1x2RX_44210 MODE | |
|---|---|---|
| Octet | 1 | 2 |
| lane STX0 | RXA_i0[15:8] RXA_i0[7:0] | RXA_i0[7:0] |
| lane STX1 | RXA_q0[15:8] | RXA_q0[7:0] |
| Lane STX2 | RXB_i0[15:8] | RXB_i0[7:0] |
| Lane STX3 | RXB_q0[15:8] | RXB_q0[7:0] |

**Table 5-23. Mapping of Transport Layer Octet to 44210 Mode**

| EXAMPLE | JESD204B X 1x2RX_44210 MODE | |
|---|---|---|
| Octet | 1 | 2 |
| lane STX0 | Octet0 | Octet1 |
| lane STX1 | Octet2 | Octet3 |
| Lane STX2 | Octet4 | Octet5 |
| Lane STX3 | Octet6 | Octet7 |

After configuring the pattern, engineers can change the patterns in the SPI register (i.e. Octet0, Octet1) to change the transmitted pattern on the JESD204 TX logic. They can then read back the patterns from JESD204 RX logic on the ASIC/FPGA to see if the value matches the programmed value.

### 5.2.2 ADC Test Pattern Mode (Non-JESD204C Specification)

Besides the standard JESD204C transport layer based testing, the AFE77xxD has ramp based test pattern generator. This ramp generator replaces the standard JESD204C short pattern test and facilitates verification of all the bit transitions for the JESD204C TX IP.

```
AdcRampTestPattern()
```

Send Ramp Test Pattern from ADC JESD. This test pattern is near the ADC-JESD interface.

### 5.2.3 JESD204C RX on the DAC Side

Besides JESD204C TX short pattern generator, the JESD204C RX short pattern verifier is also available on the DAC side. One slight difference is that since each DAC has its own JESD204C RX IP logic, the transport layer packing and the associated programmable octets are with respect to the DAC channel of its own. The user should find the transport layer packing that are addressed for 1TX topology.

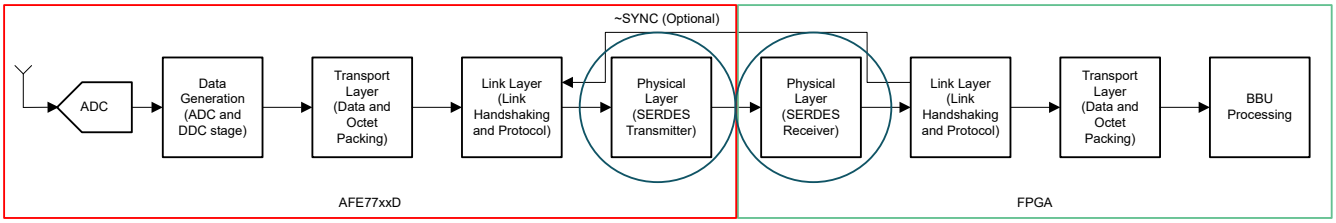


**Figure 5-2. Transport Layer of FPGA to Send Check Pattern to Link Layer of DAC**

For example, the 1x1TX of 12410 mode (or 2TX of 24410 mode) has the transport layer mapping as shown in Table 5-25, with the short pattern verifier octet pattern as shown in Table 5-26.

**Table 5-24. Transport Layer Mapping ?????**

| 0x1B6 | shorttest_ena |
|---|---|
| 1BCh | jesd_shorttest_input0 |
| 1BDh | jesd_shorttest_input0 |
| 1BEh | jesd_shorttest_input1 |
| 1BFh | jesd_shorttest_input1 |
| 1C0h | jesd_shorttest_input2 |
| 1C1h | jesd_shorttest_input2 |
| 1C2h | jesd_shorttest_input3 |
| 1C3h | jesd_shorttest_input3 |
| 1C4h | jesd_shorttest_input4 |
| 1C5h | jesd_shorttest_input4 |
| 1C6h | jesd_shorttest_input5 |
| 1C7h | jesd_shorttest_input5 |
| 1C8h | jesd_shorttest_input6 |
| 1C9h | jesd_shorttest_input6 |
| 1CAh | jesd_shorttest_input7 |
| 1CBh | jesd_shorttest_input7 |
| 1CCh | jesd_shorttest_input8 |
| 1CDh | jesd_shorttest_input8 |
| 1CEh | jesd_shorttest_input9 |
| 1CFh | jesd_shorttest_input9 |
| 1D0h | jesd_shorttest_input10 |
| 1D1h | jesd_shorttest_input10 |
| 1D2h | jesd_shorttest_input11 |

**Table 5-24. Transport Layer Mapping ????? (continued)**

| 0x1B6 | shorttest_ena |
|---|---|
| 1D3h | jesd_shorttest_input11 |
| 1D4h | jesd_shorttest_input12 |
| 1D5h | jesd_shorttest_input12 |
| 1D6h | jesd_shorttest_input13 |
| 1D7h | jesd_shorttest_input13 |
| 1D8h | jesd_shorttest_input14 |
| 1D9h | jesd_shorttest_input14 |
| 1DAh | jesd_shorttest_input15 |
| 1DBh | jesd_shorttest_input15 |

**Table 5-25. Transport Layer Mapping for 1TX of 12410 Mode**

| EXAMPLE | JESD204B TX 1x2RX_24410 MODE | | | |
|---|---|---|---|---|
| Octet | 1 | 2 | 3 | 4 |
| lane STX0 | TXA_i0[15:8] | TXA_i0[7:0] | TXA_q0[15:8] | TXA_q0[7:0] |

**Table 5-26. Mapping of Transport Layer Octet to 1TX of 12410 Mode**

| EXAMPLE | JESD204B TX 1x2RX_24410 MODE | | | |
|---|---|---|---|---|
| Octet | 1 | 2 | 3 | 4 |
| lane STX0 | octet0 | octet1 | octet2 | octet3 |

The programming and checking of the JESD204 RX logic short pattern checker involves enabling the short pattern test verifier, programming the associated octet keys to check against the JESD204B TX short pattern generator, and the alarm clearing and read back of the verifier status. TI recommends clearing the alarm register after both the JESD204B TX and RX are stable to get the most up to date reading of the verifier status.

### 5.2.4 DAC Test Pattern Mode (Non-JESD204C Specification)

The DAC logic has dedicated pattern generator block to replace incoming JESD204C data with test patterns. This can facilitate initial DAC debug testing without proper JESD204C link.

Available C Functions:

```
DacJesdConstantTestPatternEnable()
```

Enable for Send Constant Test Pattern to DAC. This test pattern is near the JESD-DUC interface. The output of the DAC will be a single tone for each band at the mixer frequency.

```
dacJesdConstantTestPatternValue()
```

Send Constant Test Pattern value to DAC. This test pattern is near the JESD-DUC interface. The output of the DAC will be a single tone for each band at the mixer frequency. The dacJesdConstantTestPatternEnable function should be called to enable this pattern.

```
DacJesdSendData()
```

This needs to be called to change from test pattern mode to normal data mode.

Besides the JESD204 constant test pattern, the AFE7769D TX signal chain itself has a tone generator before the DUC block. Configure the simple TX tone generator through the following API.

```
txSigGenTone()
```

## 6 Summary

The AFE77xxD family of RF transceiver devices supports phyiscal layer testing of the SerDes link, and also the link layer and transport layer testing of the JESD204 link. These testing are essential to ensure link stability during the overall operation of the RF system with the AFE77xxD transceiver in the design. The details of the exact setup and configuration will be AFE77xxD device dependent, and separate documents are available to describe the setup and configuration for the individual device.

# 7 References

•

# 8 Summary of C Functions