Application Report
SPRAB58—July 2009

TCI648x Antenna Interface Programming

Communication Infrastructure

Brighton Feng, Jane Lu, Albert Bae

Abstract

The TMS320TCI648x antenna interface (AIF) is a peripheral module that supports transfers of baseband antenna IQ data between uplink and downlink baseband DSP processors and a high-speed SERDES interface. The AIF supports both the OBSAI RP3 and CPRI protocols.

This document describes how to use the TCI648x Antenna Interface. Example codes based on AIF operation with the TCI6488 EVM are provided.

Contents

Tables

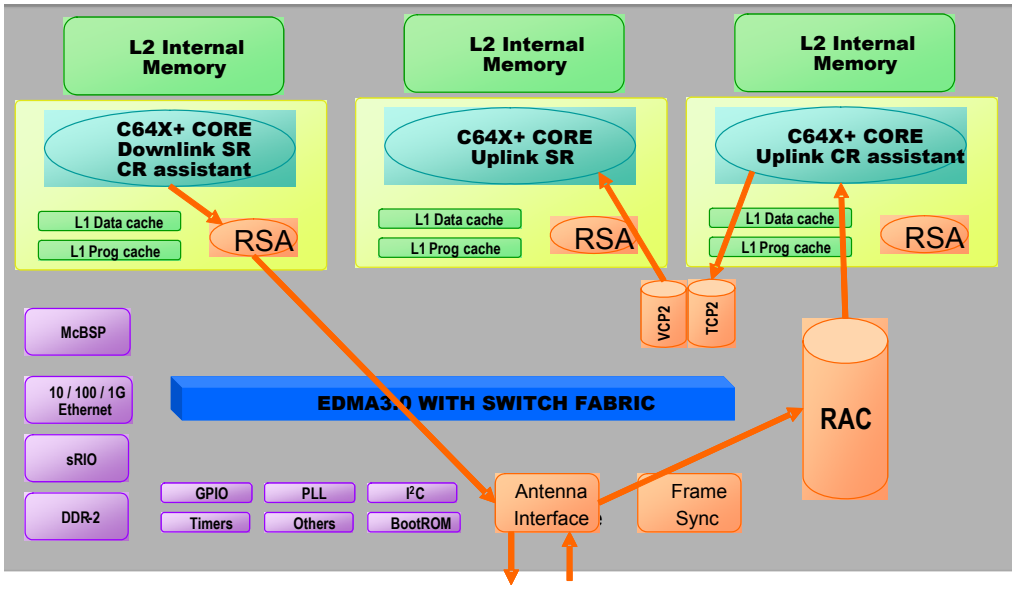
Figures

1 Introduction

The TCI6487/8 is a DSP designed to meet the requirements for wireless infrastructure baseband systems.

The DSP comprises three 1GHz C64x+ cores, several baseband accelerators, and several high-speed interfaces. All cores on the device have full access to the memory map and all the resources on the device. Figure 1 shows a block diagram of the TCI6487/8.

Figure 1 TCI6487/8 Block Diagram and Typical Data Processing Flow



The integrated antenna interface (AIF) supports both OBSAI (Open Base Station Architecture Initiative) and CPRI (Common Public Radio Interface) standards, which interface with the IF/RF (Intermediate Frequency/ Radio Frequency) module for antenna data transfer. The AIF transfers baseband IQ data with a high-speed SERDES interface.

The typical application of the TCI6488 is WCDMA baseband processing. For uplink processing, antenna data received from the antenna interface is sent to the RAC (Receiver Accelerator Coprocessor), which performs uplink chip-rate receive processing under the direction and control of a DSP core. The symbol data is sent to the VCP2 (Viterbi Coprocessor) or TCP2 (Turbo Coprocessor) for convolution or Turbo decoding under the control of the symbol rate processing core.

On the TCI6488, WCDMA transmit chip-rate processing is implemented by a DSP subsystem and its associated Rake Search/Spread Accelerator (RSA) extensions. These RSA extensions accelerate WCDMA transmit processing by performing spreading and scrambling functions; the generated antenna data is then sent out through the antenna interface.

There are six AIF links on the TCI6487/8, which support up to 96 antenna streams and allow a flexible interconnection topology. Figure 2 shows the antenna interface subsystem on the TCI6487/8.

Figure 2 Antenna Interface Subsystem on a TCI6487/8

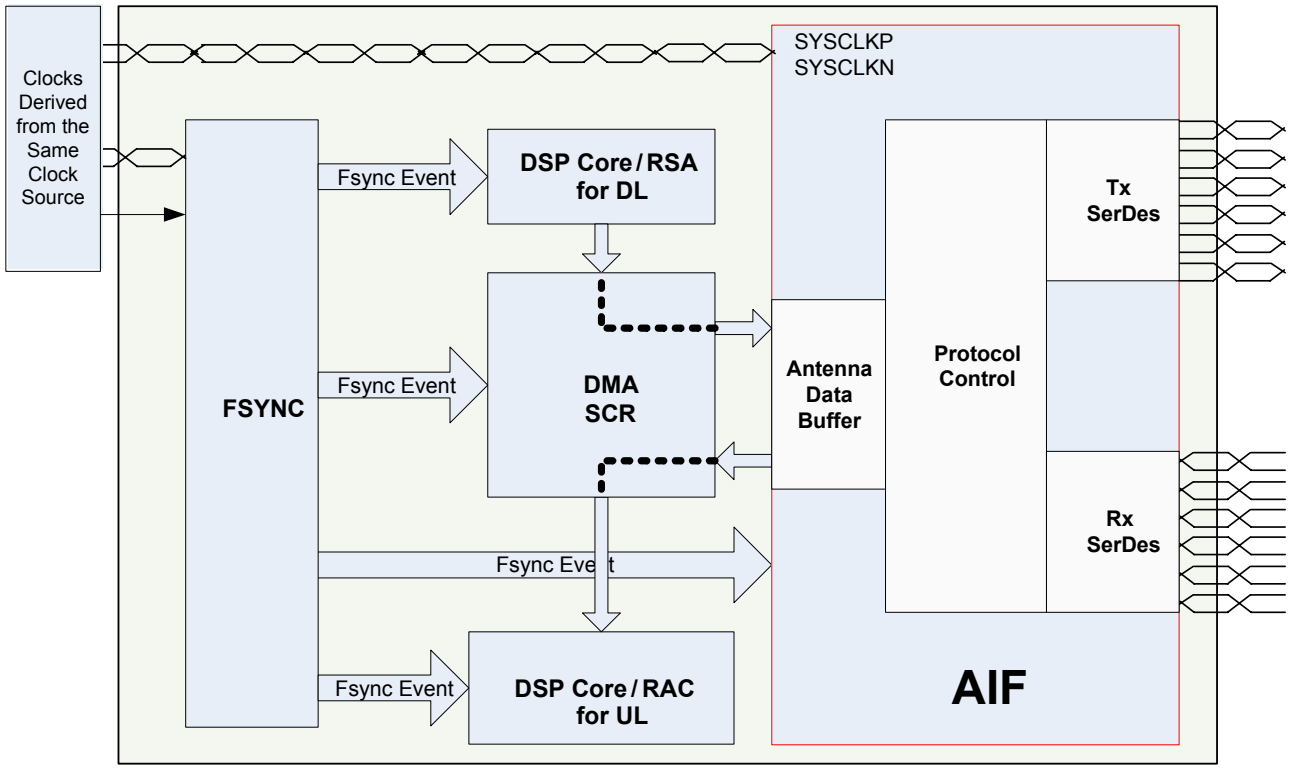
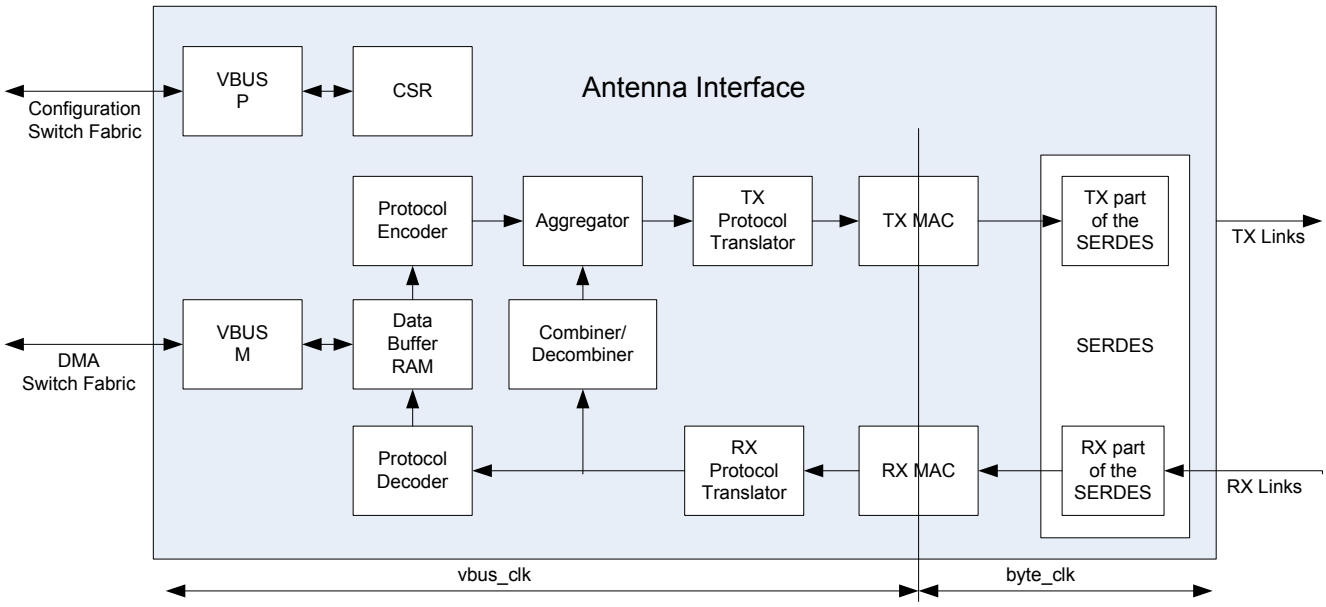


Figure 3 shows the block diagram of the AIF. For more Information about the functions of each module, see the *Antenna Interface User's Guide*.

Figure 3 AIF Block Diagram



This document describes how to program the TCI648x Antenna Interface. Example codes based on operation with the TCI6488 EVM are also provided.

2 TCI6488 EVM and AIF Test Setup

The example codes supplied with this application note are based on the TCI6488 EVM. The EVM board has two TCI6488 DSPs running at 1 GHz. Figure 4 is a block diagram of the TMS320TCI6488 EVM.

Figure 4 TCI6488 EVM Block Diagram

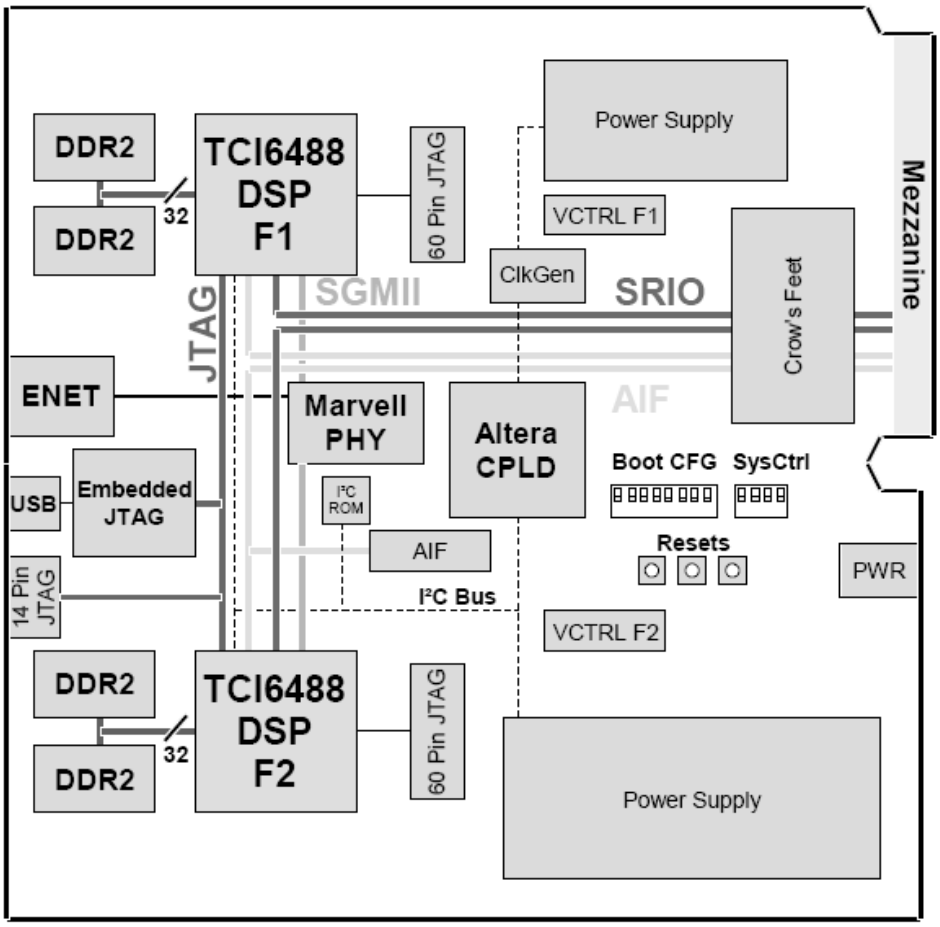
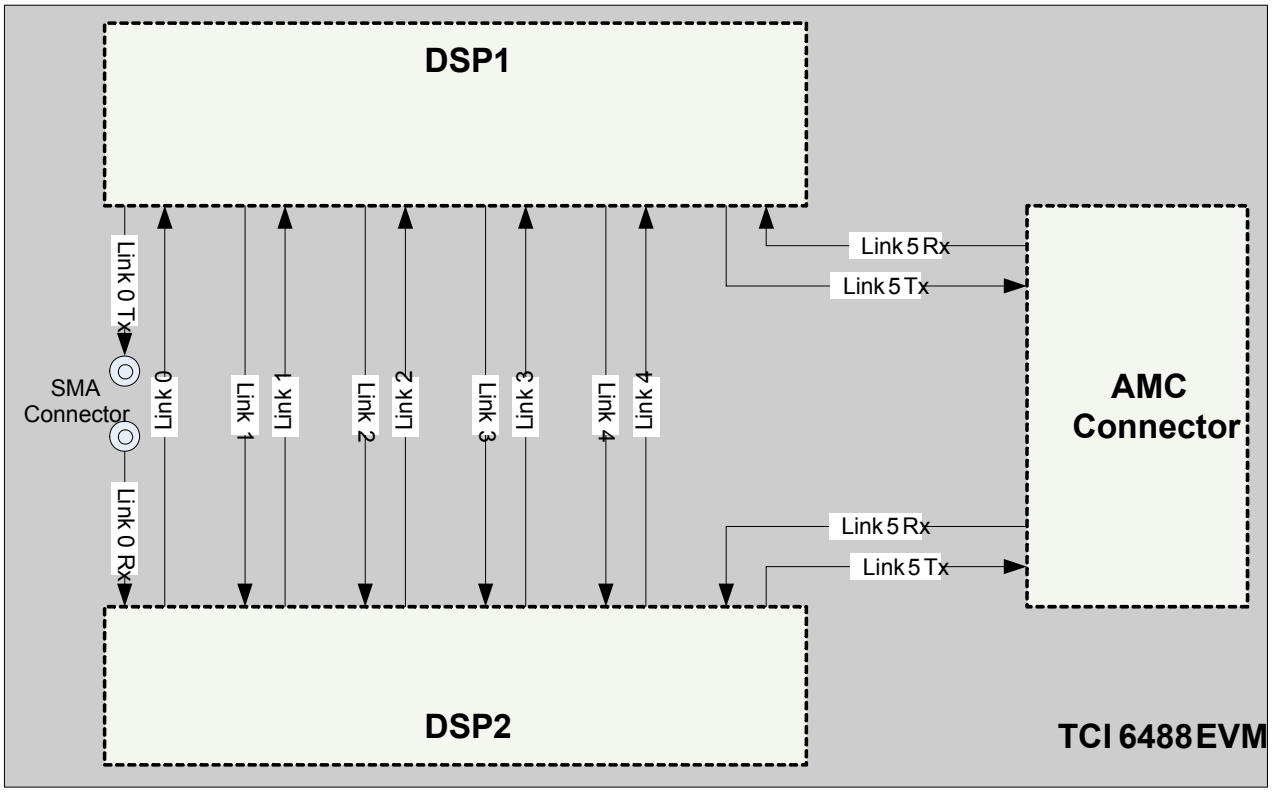


Figure 5 on page 7 shows the AIF connection between two DSPs on the TCI6488 EVM. The six AIF links are used to allow the following in the default configuration:

- Support of SMA connections for the incoming stream on DSP2 and the outgoing stream on DSP1
- Support of four full DSP-to-DSP connections. The four connections can be used to test antenna data transfer between DSPs.
- Support of one full-link connection to the AMC Connector

Figure 5 AIF Connections Between Two DSPs on TCI6488 EVM



The example codes demonstrate diversiform working modes of the AIF; the following parameters are user-configurable to verify most functions of the AIF:

- Protocol: OBSAI, CPRI
- Link rate: 1x, 2x, 4x
- Data Type: DL, UL_RSA
- Data Width
  - 7 bits (CPRI only)
  - 8 bits
  - 15 bits (CPRI only)
  - 16 bits
- Data Path
  - Internal loopback
  - External redirection
  - External redirection and aggregation with external stream (Circuit-switched data only)

Figure 6, Figure 7, and Figure 8 show different test data paths.

Figure 6 Internal Loopback Test

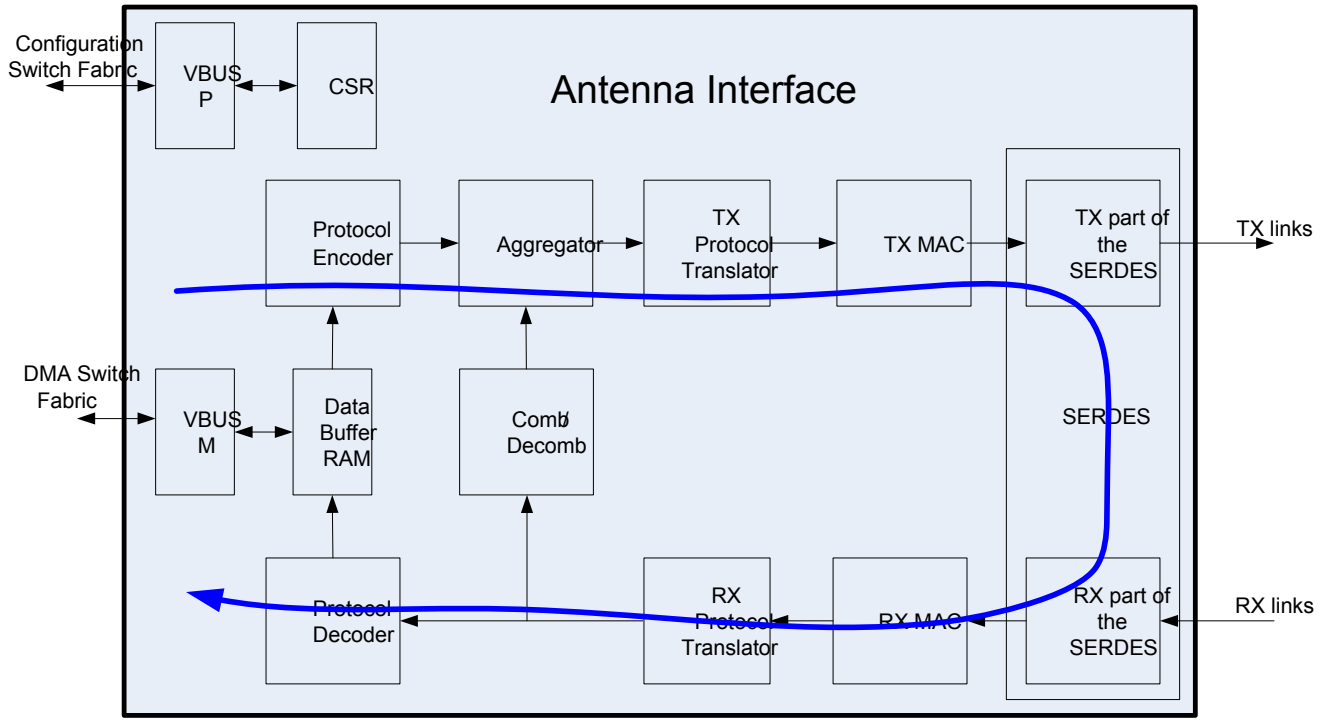


Figure 7 External Redirection Test

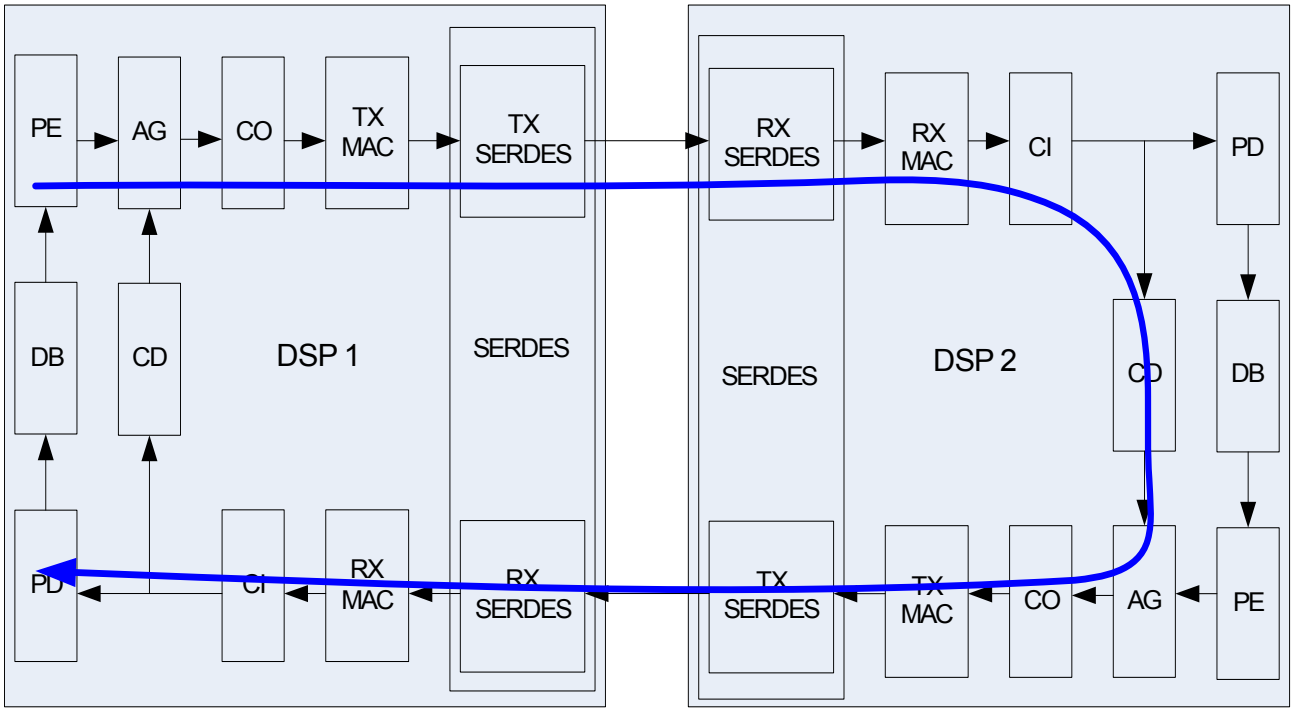
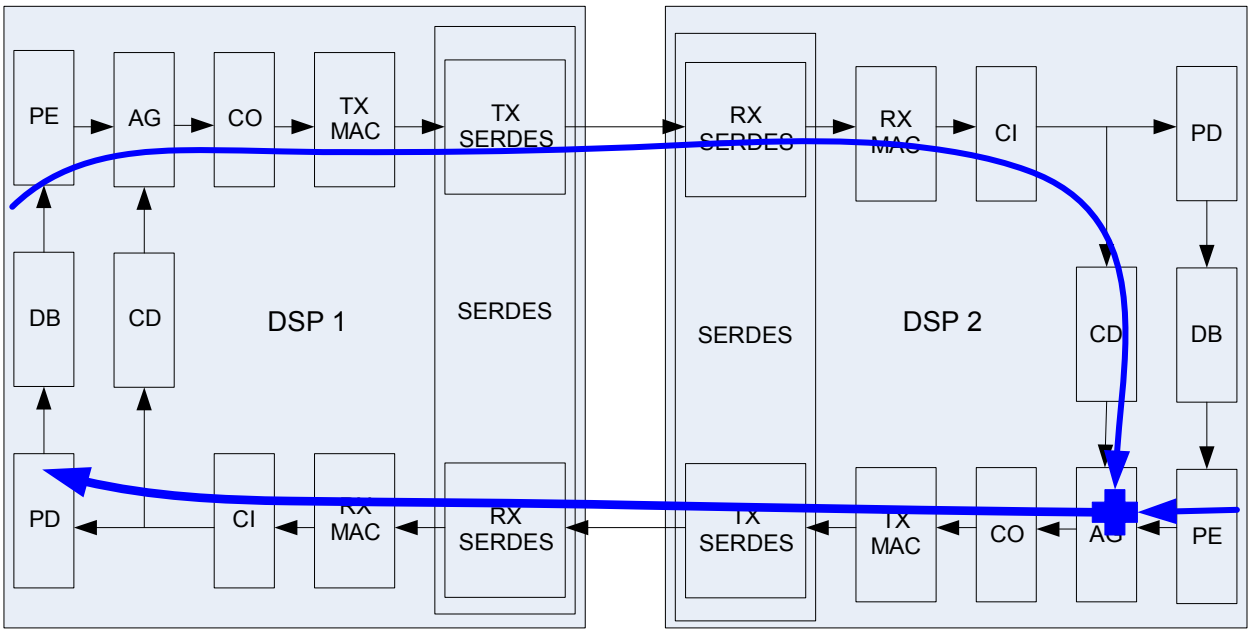


Figure 8 External Redirection and Aggregation Test



NOTE—Link 0 and link 5 of two DSPs are not directly connected to each other on this EVM; without external connections, they can be used only for internal loopback testing.

A typical test configuration is as follows:

- Link 1: 1x, 8 bits UL_RSA, internal loopback test on every DSP
- Link 2: 4x, 16 bits DL, external redirection and aggregation test between two DSPs
- Link 3: 2x, 8 bits UL_RSA, external redirection test between two DSPs
- Link 4: 2x, 16 bits DL, external redirection and aggregation test between two DSPs

NOTE—See "Using the Example Codes with this Application Note" on page 45 for more details about the examples codes.

3 Antenna Data Structure and EDMA Configuration

Data structure is the basis of software programming; understanding the antenna data structure is very important for programming the antenna interface. The EDMA configuration is determined by the antenna data buffer structure.

3.1 AIF Antenna Data Buffer Structure

There are several RAM buffers in the AIF module for antenna data transfer. Table 1 shows the AIF antenna data buffer memory map on the TCI6487/8.

Table 1 AIF Antenna Data Buffer Memory Map (Part 1 of 2)

Address	Physical Size	Mapped Size	Description
0**x**A0000000	2KB	4MB	Inbound Circuit Switched RAM for link0
0**x**A0400000	2KB	4MB	Inbound Circuit Switched RAM for link1
0**x**A0800000	2KB	4MB	Inbound Circuit Switched RAM for link2
0**x**A0C00000	2KB	4MB	Inbound Circuit Switched RAM for link3
0**x**A1000000	2KB	4MB	Inbound Circuit Switched RAM for link4
0**x**A1400000	2KB	4MB	Inbound Circuit Switched RAM for link5
0xA1800000			Reserved
0**x**A2000000	2KB	4MB	Outbound Circuit Switched RAM for link0
0**x**A2400000	2KB	4MB	Outbound Circuit Switched RAM for link1
0**x**A2800000	2KB	4MB	Outbound Circuit Switched RAM for link2
0**x**A2C00000	2KB	4MB	Outbound Circuit Switched RAM for link3
0**x**A3000000	2KB	4MB	Outbound Circuit Switched RAM for link4
0**x**A3400000	2KB	4MB	Outbound Circuit Switched RAM for link5
0xA3800000			Reserved
0**x**A4000000	Shared 9.5KB	4MB	OBSAI Inbound Packet Switched FIFO0
0**x**A4400000		4MB	OBSAI Inbound Packet Switched FIFO1
0**x**A4800000		4MB	OBSAI Inbound Packet Switched FIFO2
0**x**A4C00000		4MB	OBSAI Inbound Packet Switched FIFO3
0**x**A5000000	Shared 9.5KB	512KB	OBSAI Outbound Packet Switched FIFO0
0**x**A5080000		512KB	OBSAI Outbound Packet Switched FIFO1
0**x**A5100000		512KB	OBSAI Outbound Packet Switched FIFO2
…		…	…
0**x**A5E00000		512KB	OBSAI Outbound Packet Switched FIFO28
0**x**A5E80000		512KB	OBSAI Outbound Packet Switched FIFO29
0**x**A5F00000			Reserved
0**x**A6000000	1KB	2KB	Inbound CPRI Control Word RAM for link0
0**x**A6000800	1KB	2KB	Inbound CPRI Control Word RAM for link1
0**x**A6001000	1KB	2KB	Inbound CPRI Control Word RAM for link2
0**x**A6001800	1KB	2KB	Inbound CPRI Control Word RAM for link3
0**x**A6002000	1KB	2KB	Inbound CPRI Control Word RAM for link4
0**x**A6002800	1KB	2KB	Inbound CPRI Control Word RAM for link5
0**x**A6003000			Reserved
0**x**A7000000	1KB	2KB	Outbound CPRI Control Word RAM for link0

Table 1 AIF Antenna Data Buffer Memory Map (Part 2 of 2)

Address	Physical Size	Mapped Size	Description
0**x**A7000800	1KB	2KB	Outbound CPRI Control Word RAM for link1
0**x**A7001000	1KB	2KB	Outbound CPRI Control Word RAM for link2
0**x**A7001800	1KB	2KB	Outbound CPRI Control Word RAM for link3
0**x**A7002000	1KB	2KB	Outbound CPRI Control Word RAM for link4
0**x**A7002800	1KB	2KB	Outbound CPRI Control Word RAM for link5

The physical circuit-switched buffer size for one link is 2KB, but it is mapped to a 4MB memory range repeatedly. Linearly accessing the 4MB memory range accesses the 2KB memory buffer in a circular or wrap-around way. This mechanism simplifies the EDMA configuration when performing bigger frame size transfers to and from the DSP memory.

Each 2KB Inbound or Outbound RAM buffer holds 32 chips of antenna data for 16 streams; the 32-chip buffer per stream is also a circular buffer for AIF hardware to access while the DMA is accessing this buffer.

The data in the buffer is organized according to data type: DL (DownLink) or UL_RSA (UpLink_RSA). Figure 9 and Figure 10 show the formats of the DL and UL_RSA data types.

Figure 9 DL Four-Chips Burst Antenna Data Format in AIF RAM

	Stream 0	Stream 1	.....	Stream 14	Stream 15
Chip0~3	S0C0~3	S1C0~3	.....	S14C0~3	S15C0~3
Chip4~7	S0C4~7	S1C4~7	.....	S14C4~7	S15C4~7
Chip8~11	S0C8~11	S1C8~11	.....	S14C8~11	S15C8~11
Chip12~15	S0C12~15	S1C12~15	.....	S14C12~15	S15C12~15
Chip16~19	S0C16~19	S1C16~19	.....	S14C16~19	S15C16~19
Chip20~23	S0C20~23	S1C20~23	.....	S14C20~23	S15C20~23
Chip24~27	S0C24~27	S1C24~27	.....	S14C24~27	S15C24~27
Chip28~31	S0C28~31	S1C28~31	.....	S14C28~31	S15C28~31

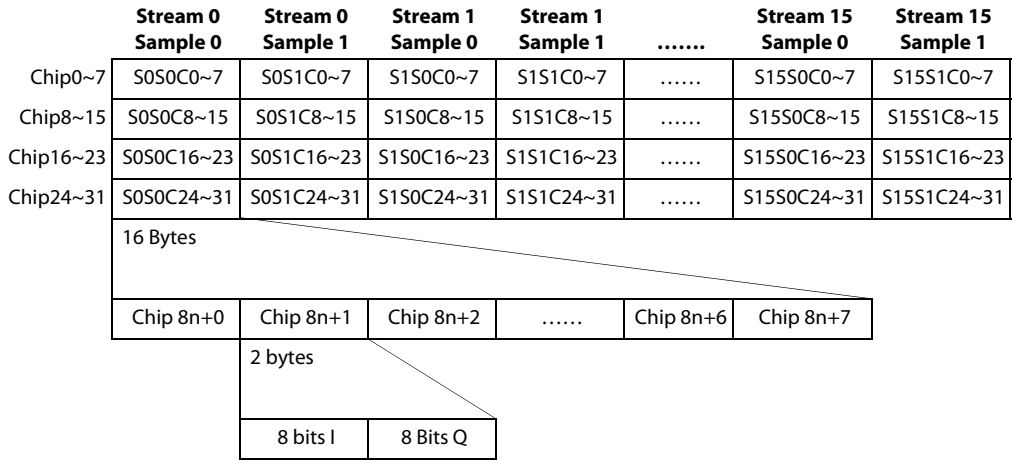
16 Bytes

Chip 4n+0	Chip 4n+1	Chip 4n+2	Chip 4n+3

4 bytes

16 bits I	16 Bits Q

Figure 10 UL_RSA Eight-Chips Burst Antenna Data Format in AIF RAM



One AIF link can support up to 16 streams. Table 2 shows the number of streams the AIF supports under different configurations. The buffer RAM organization is fixed; it does not change with the stream number. The location for the unused stream is just unused; this simplifies the implementation and programming of the AIF.

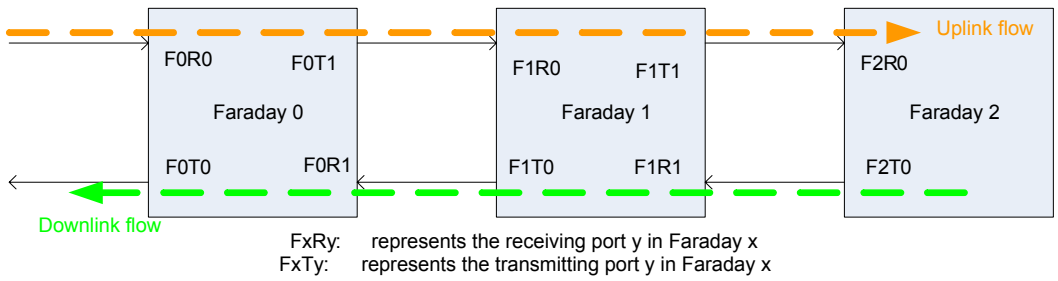
Table 2 Stream Number for Different Configurations

	1x link rate	2x link rate	4x link rate
OBSAI	4	8	16
CPRI 7/15 bit	4	8	16
CPRI 8/15 bit	3	7	15

For CPRI 7-bit or 15-bit data types, every data element still occupies 8- or 16-bit memory space in the AIF buffer, but when the data is transmitted, the AIF hardware saturates the value to the 7- or 15-bits range for aggregation or throws the MSB (Most Signification Bit) for non-aggregation operation; when 7- or 15-bit data is received, it is sign-extended and saved to an 8- or 16-bit memory unit.

NOTE—The terms *inbound* and *outbound* do not mean *uplink* and *downlink*, respectively. Data type transport through the Inbound (input port) and Outbound (output port) can be either DL or UL_RSA. As Figure 11 shows, the Uplink data flow (UL_RSA type) goes through inbound and outbound ports of several DSPs; similarly, downlink data flow (DL type) also goes through inbound and outbound ports. It is the data type (DL or UL_RSA), not whether it is *inbound* or *outbound*, that determines how the AIF buffer is organized.

Figure 11 Antenna Data Flow Through AIF Daisy Chain



DL data type must be four-chips burst and UL_RSA data type must be eight-chips burst. One burst of data is the minimum data transferred with one EDMA transfer triggered by one event; that is, one burst of data is just one row in the data format figures above.

Inbound packet-switched RAM for OBSAI is accessed through four FIFO interfaces. The 9.5KB memory can store 512 OBSAI messages; each message occupies 19 bytes, that is, 512*19= 9728 bytes= 9.5KB. The memory partition between four FIFOs is configurable through software. Inbound packet-switched messages from all six links are routed to one of the four FIFOs according to their OBSAI address and type, which are also configurable through software.

Outbound packet-switched RAM for OBSAI is accessed through 30 FIFO interfaces. The 9.5KB memory can also store 512 OBSAI messages. The memory allocation between 30 FIFOs is dynamically allocated by hardware without software intervention, which maximizes the use of the RAM. Outbound packet-switched messages to six links are fetched from one of the 30 FIFOs according their time slot configuration. Normally, each link can use five FIFOs, link 0 uses FIFO 0~4, link 1 uses FIFO 5~9… link 5 uses FIFO 25 to 29.

The physical CPRI Control Word RAM size for one link is 1KB, but it occupies 2KB memory space. For every 16 bytes used, there is 16 bytes of unused space as shown in Figure 12 through Figure 14. This requires a special EDMA configuration. To save the DSP memory space, the CPRI control frame buffer in DSP memory should be a compact 1KB buffer without unused space, and EDMA can access it linearly.

Figure 12 Data Format in CPRI Control Word RAM for 1x Link

	16 bytes for control words				
Hype Frame 0	CW0	CW1	…	CW15	16 bytes unused space
	CW16	CW17	…	CW31	16 bytes unused space
	…	…	…	…	...
	CW224	CW225	…	CW239	16 bytes unused space
	CW240	CW241	…	CW255	16 bytes unused space
Hype Frame 1	CW0	CW1	…	CW15	16 bytes unused space
	CW16	CW17	…	CW31	16 bytes unused space
	…	…	…	…	...
	CW224	CW225	…	CW239	16 bytes unused space
	CW240	CW241	…	CW255	16 bytes unused space
Hype Frame 2	CW0	CW1	…	CW15	16 bytes unused space
	CW16	CW17	…	CW31	16 bytes unused space
	…	…	…	…	...
	CW224	CW225	…	CW239	16 bytes unused space
	CW240	CW241	…	CW255	16 bytes unused space

Hype Frame 3	CW0	CW1	…	CW15	16 bytes unused space
	CW16	CW17	…	CW31	16 bytes unused space
	…	…	…	…	...
	CW224	CW225	…	CW239	16 bytes unused space
	CW240	CW241	…	CW255	16 bytes unused space

Figure 13 Data Format in CPRI Control Word RAM for 2x Link

16 bytes for control words Å@

Hype Frame 0	CW0	CW1	…	CW7	16 bytes unused space
	CW8	CW9	…	CW15	16 bytes unused space
	…	…	…	…	...
	CW240	CW241	…	CW247	16 bytes unused space
	CW248	CW249	…	CW255	16 bytes unused space
Hype Frame 1	CW0	CW1	…	CW7	16 bytes unused space
	CW8	CW9	…	CW15	16 bytes unused space
	…	…	…	…	...
	CW240	CW241	…	CW247	16 bytes unused space
	CW248	CW249	…	CW255	16 bytes unused space

Figure 14 Data Format in CPRI Control Word RAM for 4x Link

16 bytes for control words

Hype Frame 0	CW0	CW1	…	CW3	16 bytes unused space
	CW4	CW5	…	CW7	16 bytes unused space
	…	…	…	…	...
	CW248	CW249	…	CW251	16 bytes unused space
	CW252	CW253	…	CW255	16 bytes unused space

There are 256 CPRI control words in a CPRI hyperframe. Each of these control words is {1, 2, 4} bytes corresponding to {1x, 2x, 4x} link rates. As a result of this difference, the allocated RAM per segment holds {4, 2, 1} hyperframes of CPRI control words corresponding to {1x, 2x, 4x} link rates.

Normally, the AIF accesses the control RAM sequentially in a circular or wrap-around way, while at every NodeB frame boundary, the AIF resets the access address to the beginning of the control RAM. For 2x or 4x link, these two addressing rules are consistent. At the NodeB frame boundary, the circular or wrap-around mechanism just makes the next access address at the beginning of the control RAM. But there is a corner case based on these two addressing rules for 1x link. At the next NodeB frame boundary, the circular or wrap-around mechanism makes the next access address at the middle of the control RAM because the access address for hyper frame 150 (next NodeB Frame boundary) is at frame block 2 (150%4= 2, totally four frame blocks in the RAM for 1x link) of the control RAM. To adapt this corner case, the EDMA should be configured manually to reset the access address to the beginning of the control RAM at the NodeB frame boundary for 1x link.

3.2 EDMA Configuration for a DL-Type Antenna Stream in AIF RAM

For a DL-type antenna stream, EDMA is triggered every four chips, EDMA transfers four chips for all available streams with each trigger event. Table 3 shows the configuration of important EDMA parameters for a single link.

Table 3 EDMA configuration for DL data type in AIF RAM

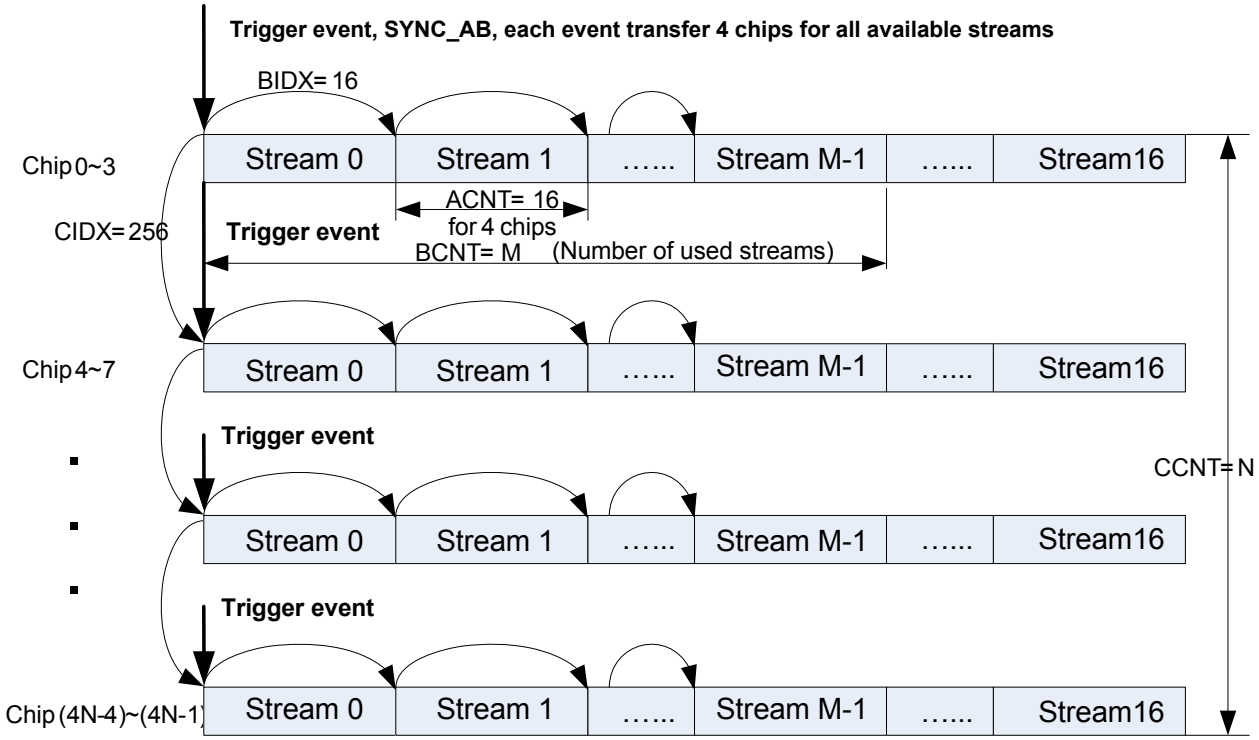
Parameters	Configuration	Comments
ACNT	16	4 (chips) x 4 (bytes), for 4 chips of one stream
BIDX	16	index to the next stream
BCNT	Number of used streams	Support maximum 16 antenna streams
CIDX	256	16 (bytes) x 16 (Streams), index to the next 4 chips for 16 streams. This is fixed value for all link speed.
CCNT	N	4N chips are transferred
Synchronization	SYNC_AB	EDMA transfers 4 chips for all available streams with each trigger event

CCNT should be as large as possible for more efficient of EDMA transfer. Although the AIF buffer size for each link is only 2KB (for 32 chips of data), it is mapped to a 4MB memory range repeatedly. Linearly accessing the 4MB memory range accesses the 2KB memory buffer in a circular or wrap-around way. So, the EDMA can linearly access 32 chips * (4MB/ 2KB) = 64K chips of data, so the maximum CCNT can be (64K chips)/(4 chips)= 16K.

Because EDMA transfers data between the AIF buffer and the other data buffer, to determine the value of CCNT should also take into account the buffer size at the other end. So, CCNT= Min(Max CCNT for AIF buffer, Max CCNT for the other buffer). The Max CCNT for the data buffers of the other end is discussed later.

Figure 15 illustrates EDMA transfer for a single link.

Figure 15 EDMA Configuration for DL Data Type in AIF RAM



3.3 EDMA configuration for UL_RSA type antenna stream in AIF RAM

For a UL_RSA-type antenna stream, EDMA is triggered every eight chips. EDMA transfers eight chips for all available streams with each trigger event. Table 4 shows the configuration of important EDMA parameters for a single link.

Table 4 EDMA Configuration for UL_RSA Data Type in AIF RAM

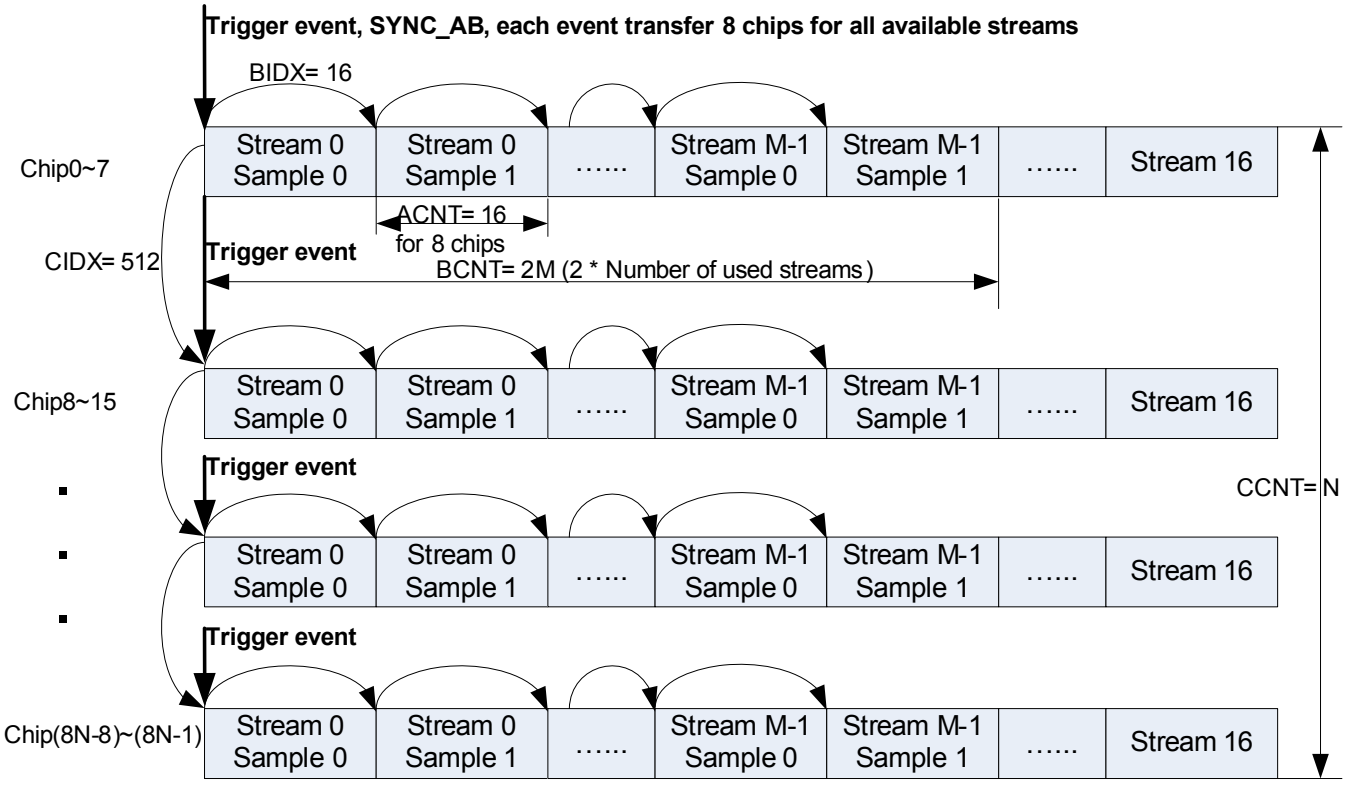
Parameters	Configuration	Comments
ACNT	16	8 (chips) x 2 (bytes), for 8 chips of one stream
BIDX	16	Index to the next sample, then to next stream
BCNT	2* Number of used streams	Every stream has two samples. Support maximum 16 antenna streams.
CIDX	512	16 (bytes) x 2 (samples) x 16 (Streams), index to the next 8 chips for 16 streams
CCNT	N	8N chips are transferred
Synchronization	SYNC_AB	EDMA transfers 8 chips for all available streams with each trigger event

CCNT should be as large as possible for more efficient EDMA transfer. Although the AIF buffer size for each link is only 2KB (for 32 chips of data), it is mapped to a 4MB memory range repeatedly. Linearly accessing the 4MB memory range accesses the 2KB memory buffer in a circular or wrap-around way. So, the EDMA can linearly access 32 chips * (4MB/2KB) = 64K chips of data, so the maximum CCNT can be (64K chips)/ (8 chips)= 8K.

Because EDMA transfers data between the AIF buffer and the other data buffer, to determine the value of CCNT should also take into account the other buffer size. So, CCNT= Min(Max CCNT for AIF buffer, Max CCNT for other buffer). The Max CCNT for the data buffers of the other end is discussed later.

Figure 16 illustrates EDMA transfer for a single link.

Figure 16 EDMA Configuration for UL_RSA Data Type in AIF RAM



3.4 EDMA Configuration for Packet-Switched FIFO

For inbound packet-switched messages, EDMA is triggered when the FIFO is not empty. For outbound packet-switched messages, EDMA should be triggered manually under control of the application software as long as the FIFO is not full.

The DSP or DMA sends outbound packet-switched messages to the AIF over the 16-byte (128 bits) Bus. The data/DMA Bus is 16 bytes wide and the packet-switched message is 19 bytes wide. A minimum of two Bus writes/reads (32 bytes) is required for every FIFO access. This also requires the message buffer in DSP memory to be a multiple of 32 bytes though only 19 of 32 bytes are used.

Table 5 shows the configuration of important EDMA parameters for single FIFO.

Table 5 EDMA Configuration for Packet-Switched FIFO

Parameters	Configuration	Comments
ACNT	32	32 bytes per message, only 19 bytes are valid
BIDX	0	Access FIFO in fixed address
BCNT	Number of message	
CIDX	0	Access FIFO in fixed address
CCNT	1	
Synchronization	SYNC_AB	

3.5 EDMA Configuration for CPRI Control Words

Because CPRI control words have regular/deterministic flow characteristics that are very similar to the antenna stream, the CPRI control words should be transferred with the antenna stream. The EDMA chain feature can be used to trigger the transfer of CPRI control words after the antenna stream transfer—that is, chain the EDMA for CPRI control words to the EDMA for antenna stream. That way, when the antenna stream is transferred every four or eight chips, the CPRI control words are also transferred every four or eight chips.

Table 6 shows the configuration of important EDMA parameters for a single link.

Table 6 EDMA Configuration for CPRI Control Words

Parameters	Configuration	Comments
ACNT	(burst chips)*(link rate) or 16	Every trigger should transfer control words for a burst chips (4 or 8 chips), every control word is 1 byte for 1x link, 2 bytes for 2x link and 4 bytes for 4x link. ACNT should no more than 16 because of the special CPRI control RAM structure. So, for burst=8 and rate=4, the ACNT=16
BCNT	16/ACNT or 2	ACNT*BCNT should be multiple of 16 bytes because of the special CPRI control RAM structure. For burst=8 and rate=4, the BCNT=2
BIDX	ACNT or 32	BIDX=ACNT to access the data for next burst. For burst=8 and rate=4, the ACNT=16, to access the next data block, it should skip the 16 unused space, thus the BIDX=32.
CIDX	ACNT+16 or 64	CIDX=ACNT should access the next burst, but need skip 16 unused space, so CIDX=ACNT+16 For burst=8 and rate=4, CIDX should be configured to access the next 32 bytes, but need skip 32 unused bytes, so CIDX= 64
CCNT	256 *(link rate)/ (ACNT*BCNT)	ACNT*BCNT*CCNT= size of one hyper frame= 256 *(link rate)
Sync	SYNC_A or SYNC_AB	Every trigger should transfer one burst of data. For burst=8 and rate=4, burst size = 32bytes = ACNT* BCNT, so, SYNC_AB should be used. For other cases, burst= ANCT, so SYNC_A should be used.

Figure 17 and Figure 18 show EDMA transfers for two typical scenarios.

Figure 17 EDMA Configuration for CPRI Control Words (8 Chips Burst, 4x Rate)

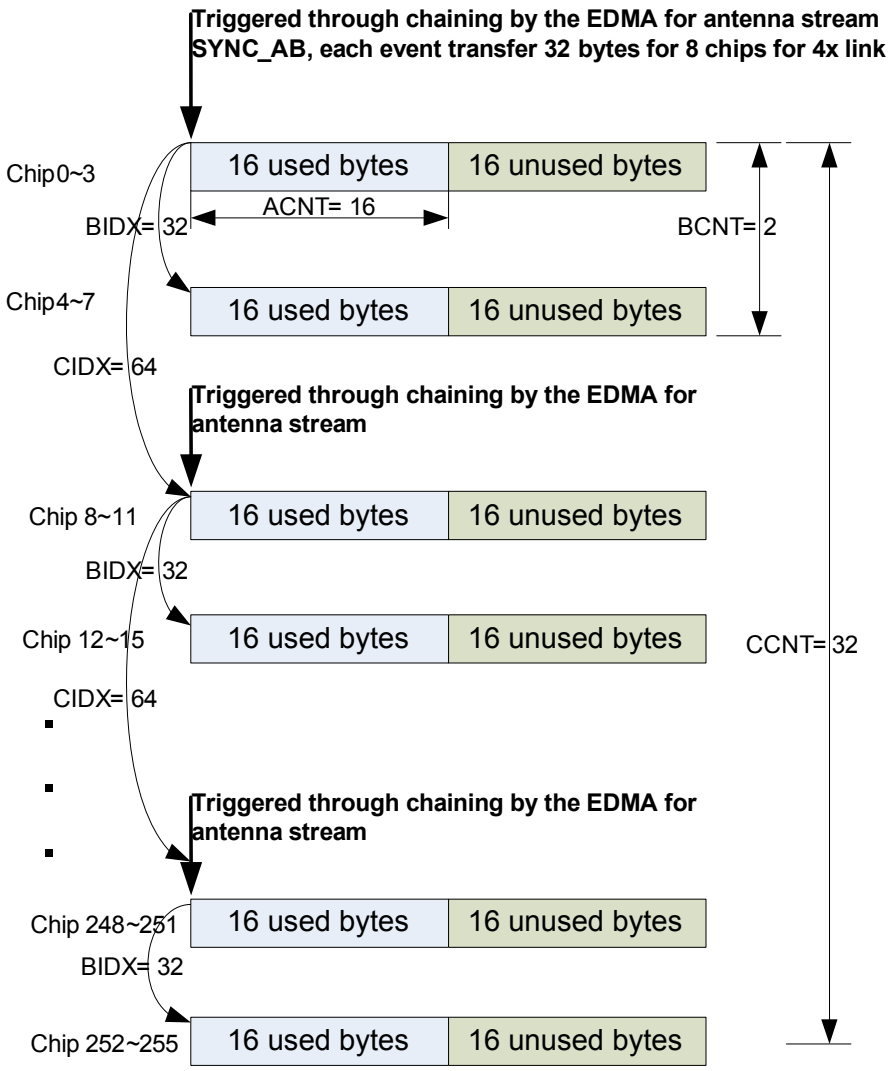
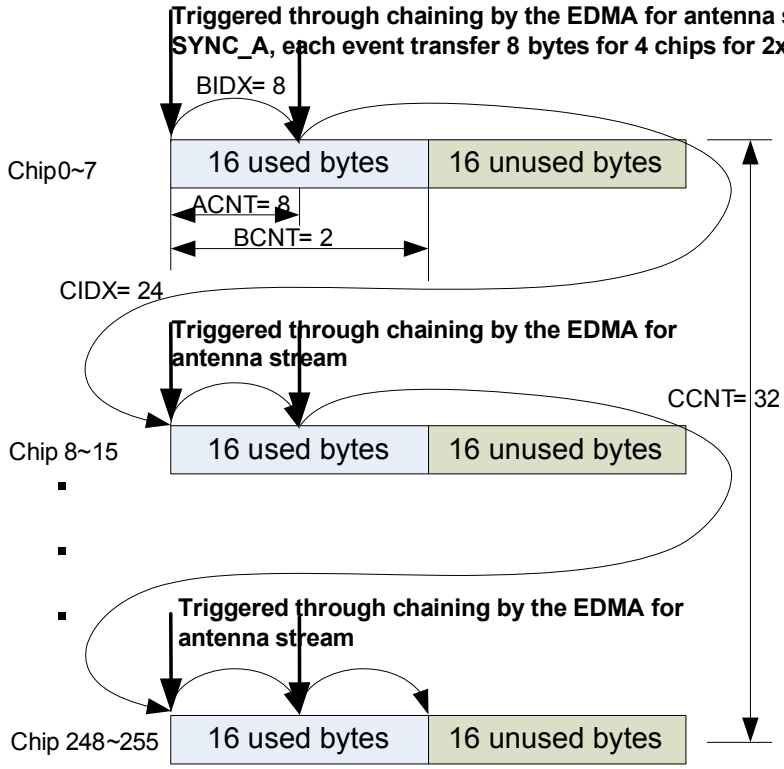


Figure 18 EDMA Configuration for CPRI Control Words (4 Chips Burst, 2x Rate)



There is a special problem for the cases with ACNT<16, as shown in Figure 18 above. Because the AIF RAM interface always swaps the byte order of 16-byte blocks, when EDMA transfers bytes 0~7 in the DSP buffer, it actually accesses bytes 8~15 in the AIF control RAM; when EDMA transfers byte 8~15 in the DSP buffer, it actually accesses bytes 0~7 in the AIF control RAM. However, the AIF accesses the control RAM sequentially after the first outbound EDMA is done, so the AIF control RAM is actually accessed by EDMA and AIF as follows:

EDMA access byte 8~15	EDMA access byte 0~7	EDMA access byte 24~31	EDMA access byte 16~23	……	
	AIF access byte 0~7	AIF access byte 8~15	AIF access byte 16~23	AIF access byte 24~31	……

So, AIF may access data which has not yet arrived at the RAM. To avoid this problem, the outbound EDMA should be started earlier as shown below; this requires a special EDMA configuration.

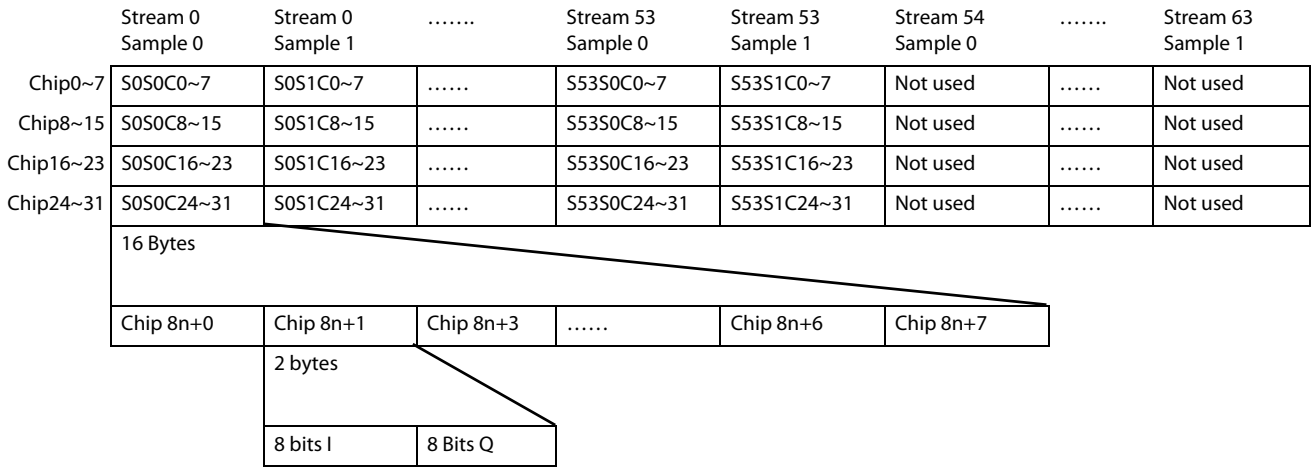
EDMA access byte 8~15	EDMA access byte 0~7	EDMA access byte 24~31	EDMA access byte 16~23	……		
		AIF access byte 0~7	AIF access byte 8~15	AIF access byte 16~23	AIF access byte 24~31	……

On the other hand, the EDMA for inbound control words should be delayed for the cases with ACNT<16.

3.6 RAC Antenna Data Buffer Structure

The RAC front-end antenna data input buffer is similar to the AIF UL_RSA data format. The RAC supports 54 streams, but the buffer is organized for 64 streams. The location for stream 54~63 is not used. Figure 19 shows how the RAC front-end buffer is organized.

Figure 19 RAC Antenna Data Buffer Structure



3.7 EDMA Configuration for RAC

The RAC antenna data buffer format is similar to the UL_RSA data type—EDMA is triggered every eight chips. EDMA transfers eight chips for multiple streams with each trigger event. Table 7 shows the configuration of important EDMA parameters.

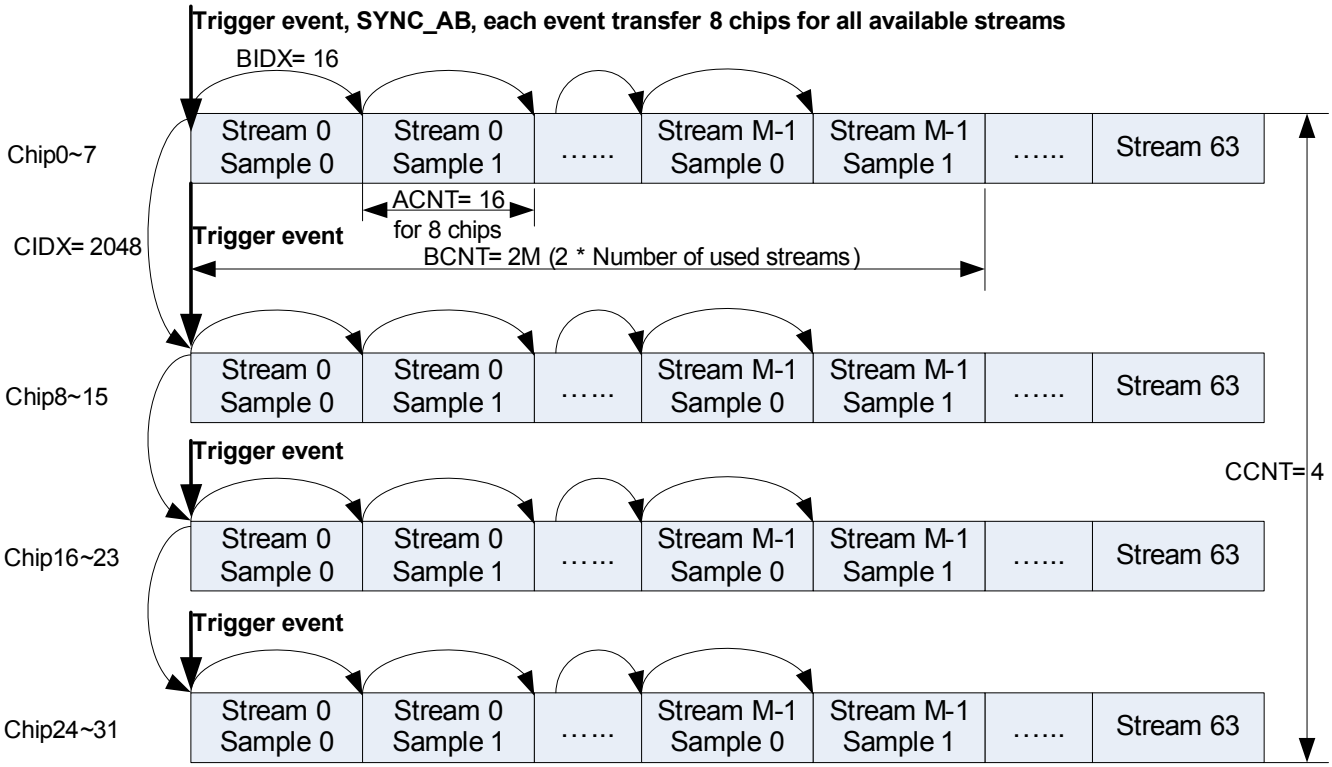
Table 7 EDMA Configuration for RAC

Parameters	Configuration	Comments
ACNT	16	8 (chips) x 2 (bytes), for 8 chips of one stream
BIDX	16	index to the next sample, then to next stream
BCNT	2* Number of used streams	Every stream has two samples. Support maximum 16 antenna streams
CIDX	2048	16 (bytes) x 2 (samples) x 64 (Streams), index to the next 8 chips for 16 streams
CCNT	4	32 chips are transferred
Synchronization	SYNC_AB	EDMA transfers 8 chips for all available streams with each trigger event

Unlike the AIF buffer, the RAC front-end buffer does not map to a larger memory range, so CCNT is fixed at four for 32 chips of data.

Figure 20 shows the EDMA transfer for RAC.

Figure 20 EDMA Configuration for RAC



3.8 Antenna Data Buffer Structure in DSP Memory

The normal data flow for outbound data is:

1. DSP processes data and stores the data in the DSP memory
2. EDMA transfers the data from DSP memory to the AIF data buffer
3. AIF fetches the data in its buffer and transmits

The normal data flow for inbound data is:

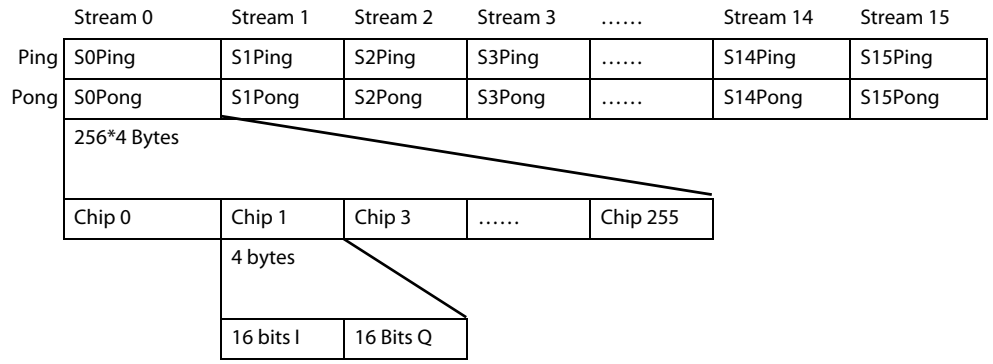
1. AIF receives antenna data and stores the data in its buffer
2. EDMA transfers the data in the AIF data buffer to DSP memory
3. DSP processes the data in its memory

The antenna data buffer in DSP memory should be organized to facilitate DSP processing. DSP processes antenna data stream by stream, and it achieves better performance if it processes more data in batches. The size of the data processed at a time is called *iteration size*. The bigger the iteration size, the better the DSP processing performance; however, a bigger iteration size results in more latency. So, it is up to the system designer to determine an acceptable tradeoff between iteration size and latency. Common iteration sizes are 64, 128, or 256 chips.

Because the antenna data buffer in DSP memory is accessed by both DSP and EDMA, ping-pong or circular buffer schemes should be used to avoid conflict between the DSP and EDMA. When DSP accesses one memory buffer, the EDMA should access the other buffer.

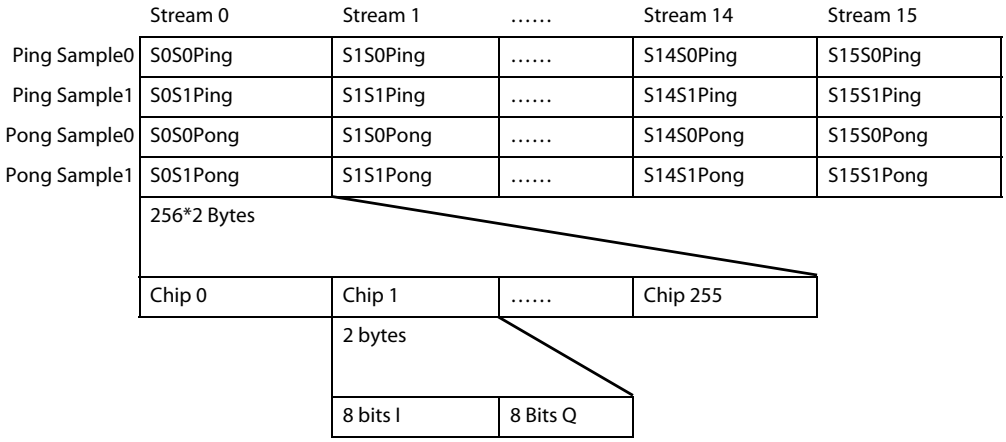
In the example provided with this application note, a ping-pong scheme is used and the iteration size corresponds to 256 chips of data. Figure 21 shows the buffer organization for DL type data in the example.

Figure 21 DL Type Data Buffer Structure Example in DSP Memory



For UL_RSA data types, two samples of every stream should be stored separately. Figure 22 shows the buffer organization for UL_RSA data in the example.

Figure 22 UL_RSA Type Data Buffer Structure Example in DSP Memory



The OBSAI packet-switched data buffer in DSP memory is fairly simple. Every message requires 32 bytes of memory (though only 19 bytes are used), the buffer size should be a multiple of 32 bytes according the message number you want to buffer. The ping-pong mechanism is required for inbound messages, but outbound messages do not require the ping-pong mechanism because the transfer is controlled by application software on the DSP core.

The CPRI control-words buffer in DSP memory is also family simple. Normally, it is a compact buffer without unused space, which should store control words for one hyper frame, for worst 4x link rate case, it is 1KB. The ping-pong mechanism is required for both inbound and outbound transfers just like circuit-switched antenna data.

3.9 EDMA Configuration for DSP data buffer for DL Type Antenna Stream

For DL-type antenna streams, EDMA is triggered every four chips. EDMA transfers four chips for all available streams with each trigger event. Table 8 shows the configuration of important EDMA parameters for a single link.

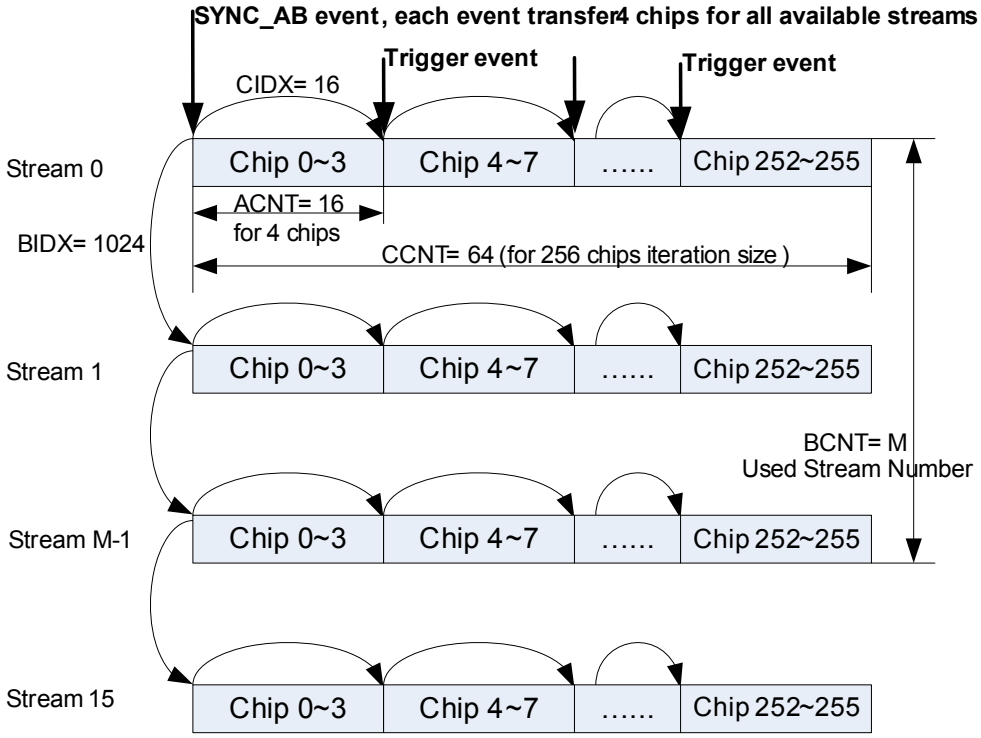
Table 8 EDMA Configuration for DSP Data Buffer for DL Data Type

Parameters	Configuration	Comments
ACNT	16	4 (chips) x 4 (bytes), for 4 chips of one stream
BIDX	16*N	index to the next stream N is the number of 4-chip bursts
BCNT	Number of used streams	Not all 16 streams are used
CIDX	16	index to the next 4 chips for 16 streams
CCNT	N (=64 for 256 chips iteration size)	4N chips are transferred
Synchronization	SYNC_AB	EDMA transfers 4 chips for all available streams with each trigger event

Because EDMA transfers data between AIF buffer and the data buffer in DSP memory, CCNT= Min (Max CCNT for the AIF buffer, Max CCNT for the DSP buffer). As described in the section above, the Max CCNT for AIF buffer for four chips burst is 16K; assume the iteration size for DSP processing is 256 chips, the Max CCNT for DSP buffer= 256/4 = 64. So, for 256 chips iteration size, the CCNT for DL data transfer between the AIF buffer and the DSP DL data buffer is 64.

Figure 23 shows an example of EDMA transfer for a single link.

Figure 23 EDMA Configuration for DSP Data Buffer for DL Data Type



3.10 EDMA Configuration for DSP Data Buffer for UL_RSA Type Antenna Stream

For a UL_RSA-type antenna stream, EDMA is triggered every eight chips. EDMA transfers eight chips for all available streams with each trigger event. Table 9 shows the configuration of important EDMA parameters for a single link.

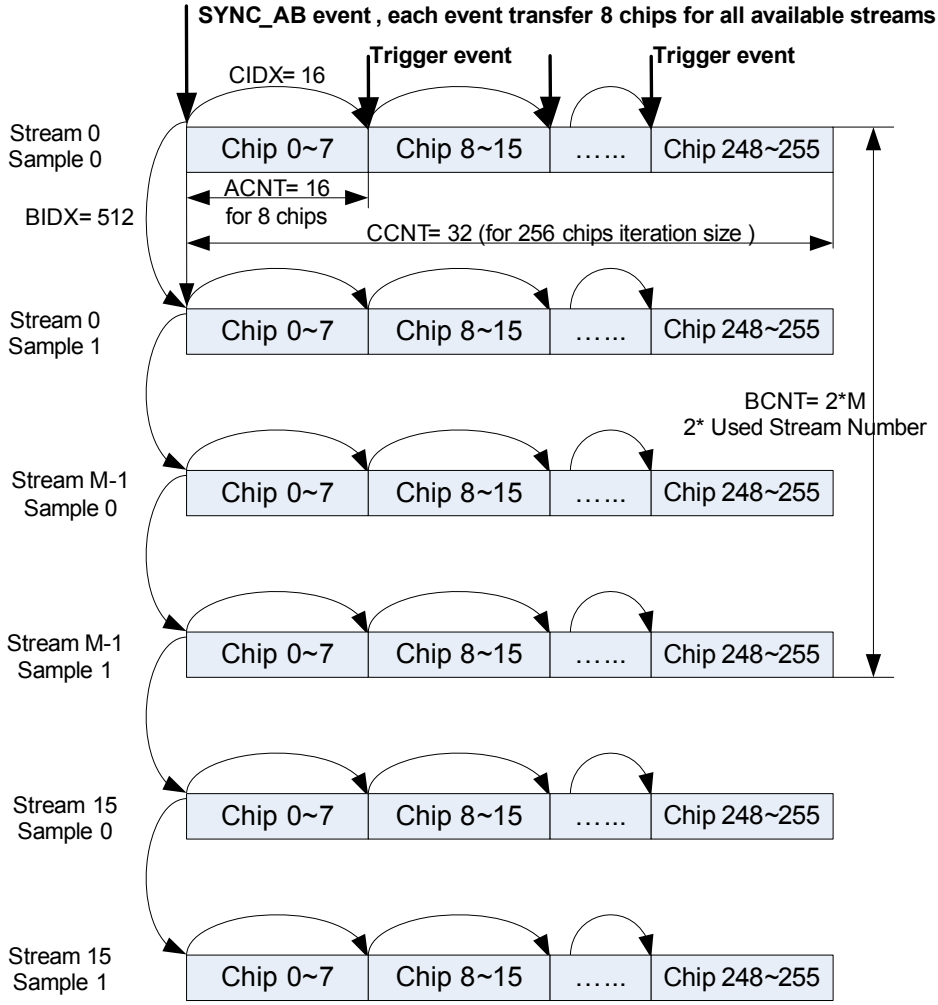
Table 9 EDMA Configuration for DSP Data Buffer for UL_RSA Data Type

Parameters	Configuration	Comments
ACNT	16	8 (chips) x 2 (bytes), for 8 chips of one stream
BIDX	16*N (Iteration size)	index to the next sample, then to next stream N is the number of 8-chip bursts
BCNT	2* Number of used streams	Every stream has two samples. Not all 16 streams are used
CIDX	16	index to the next 8 chips for 16 streams
CCNT	N (=32 for 256 chips iteration size)	8N chips are transferred
Synchronization	SYNC_AB	EDMA transfers 8 chips for all available streams with each trigger event

Because EDMA transfers data between the AIF buffer and the data buffer in DSP memory, CCNT= Min (Max CCNT for the AIF buffer, Max CCNT for the DSP buffer). As described in the section above, the Max CCNT for the AIF buffer for eight chips burst is 8K; assume the iteration size for DSP processing is 256 chips, the Max CCNT for the DSP buffer= 256/8 = 32. So, for 256 chips iteration size, the CCNT for UL_RSA data transfer between the AIF buffer and the DSP data buffer is 32.

Figure 24 shows an EDMA transfer for a single link.

Figure 24 EDMA Configuration for DSP Data Buffer for UL_RSA Data Type



3.11 EDMA Channels Used for AIF

At least two EDMA channels are required for CS (circuit-switched) antenna data transfer for one dedicated link—one for inbound and one for outbound. In total, 12 EDMA channels may be used for six links. Generally, EDMA Channels 16 to 27 are used to service the outbound/inbound AIF CS stream data transfer between the DSP memory and the AIF data buffer.

If CPRI control-words transfer is enabled, at least two EDMA channels are required for CPRI control-words transfer for one dedicated link—one for inbound and one for outbound. In total, 12 EDMA channels may be used for six links. Any free EDMA channel can be used for CPRI control words transfer. In the example in this application note, channels 52~63 are used for CPRI control-words transfer.

If OBSAI PS (packet-switched) data transfer is enabled, every outbound link requires one EDMA channel. Any free EDMA channel can be used for OBSAI PS outbound transfer. In the example, channels 58~63 are used for AIF PS outbound data transfer, one for each outbound link. These EDMA transfers are triggered manually by the

software on the DSP core. Because the OBSAI mode and CPRI mode are exclusive, channels 58~63 are used by both modes, but for different purpose. EDMA channel 40 to 42 is triggered by PS (Packet-Switched) inbound FIFO progeammable_not_empty events and used for inbound PS data transfer.

If a ping-pong mechanism is used for EDMA transfer, every EDMA channel needs at least two reload EDMA PaRAM to implement the ping-pong mechanism. Table 10 summarizes the EDMA usage in the example in this application note.

Table 10 Summary EDMA Channel Usage in the Example

Channel/PaRAM	Usage
16	Outbound CS data for link0
17	Inbound CS data for link0
18	Outbound CS data for link1
19	Inbound CS data for link1
20	Outbound CS data for link2
21	Inbound CS data for link2
22	Outbound CS data for link3
23	Inbound CS data for link3
24	Outbound CS data for link4
25	Inbound CS data for link4
26	Outbound CS data for link5
27	Inbound CS data for link5
40	Inbound PS FIFO0
41	Inbound PS FIFO1
42	Inbound PS FIFO2
52	Outbound CPRI control words for link0
53	Inbound CPRI control words for link0
54	Outbound CPRI control words for link1
55	Inbound CPRI control words for link1
56	Outbound CPRI control words for link2
57	Inbound CPRI control words for link2
58	Outbound CPRI control words for link3 Outbound PS data for link0
59	Inbound CPRI control words for link3 Outbound PS data for link1
60	Outbound CPRI control words for link4 Outbound PS data for link2
61	Inbound CPRI control words for link4 Outbound PS data for link3
62	Outbound CPRI control words for link5 Outbound PS data for link4
63	Inbound CPRI control words for link5 Outbound PS data for link5

Table 10 Summary EDMA Channel Usage in the Example

64~87	24 reload PaRAMs for CS data. (6 link * 2 channel/ link * 2 reload PaRAM/ channel = 24 reload PaRAM).
88~135	48 reload PaRAMs for CPRI control words. (6 link * 2 channel/ link * 4 reload PaRAM/ channel = 48 reload PaRAM). Every CPRI control channel uses 4 reload PaRAM because the CPRI control RAM can hold maximum 4 hyper frames for 1x links. 4 reload PaRAM are used to address these 4 different frames.
88~93	6 reload PaRAMs for inbound PS FIFOs. (3 FIFO * 1 channel/ FIFO * 2 reload PaRAM/ channel = 6 reload PaRAM).

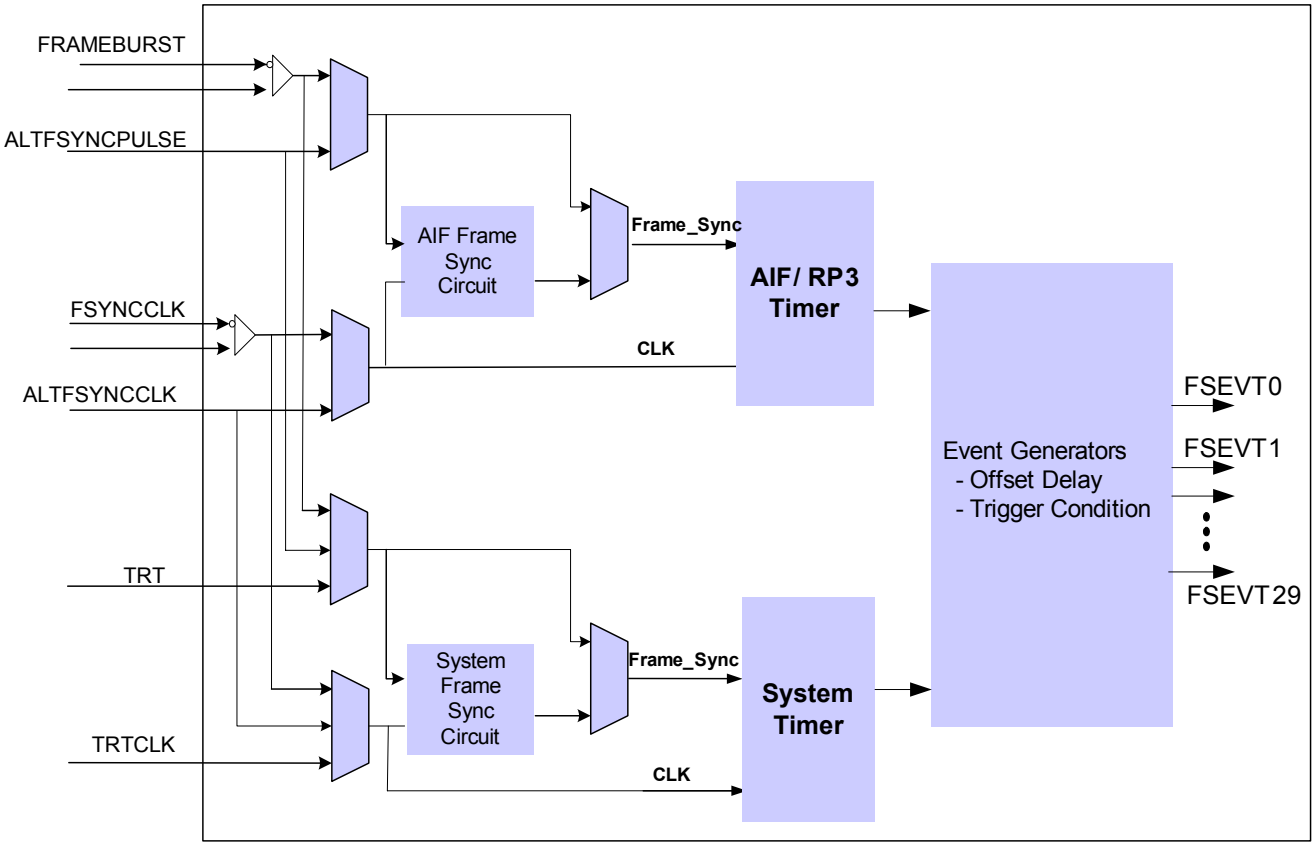
EDMA for outbound PS data does not require reload PaRAMs because it is static and it is triggered manually by application software on DSP core. Since OBSAI mode and CPRI mode are exclusive, reload PaRAM 88~92 are used by both modes, but for different purposes.

4 Frame Synchronization Configuration

The frame synchronization (FSync) module handles all UMTS and other standard-specific timing and time alignment for the TCI6487/8 device. FSync is driven by an external clock and synchronized to the system frame synchronization signal.

Figure 25 shows the block diagram of FSync. Two timers in FSync are configured to periodically generate 30 events for all other modules for antenna data processing. These events can be configured to start at any time offset with any period.

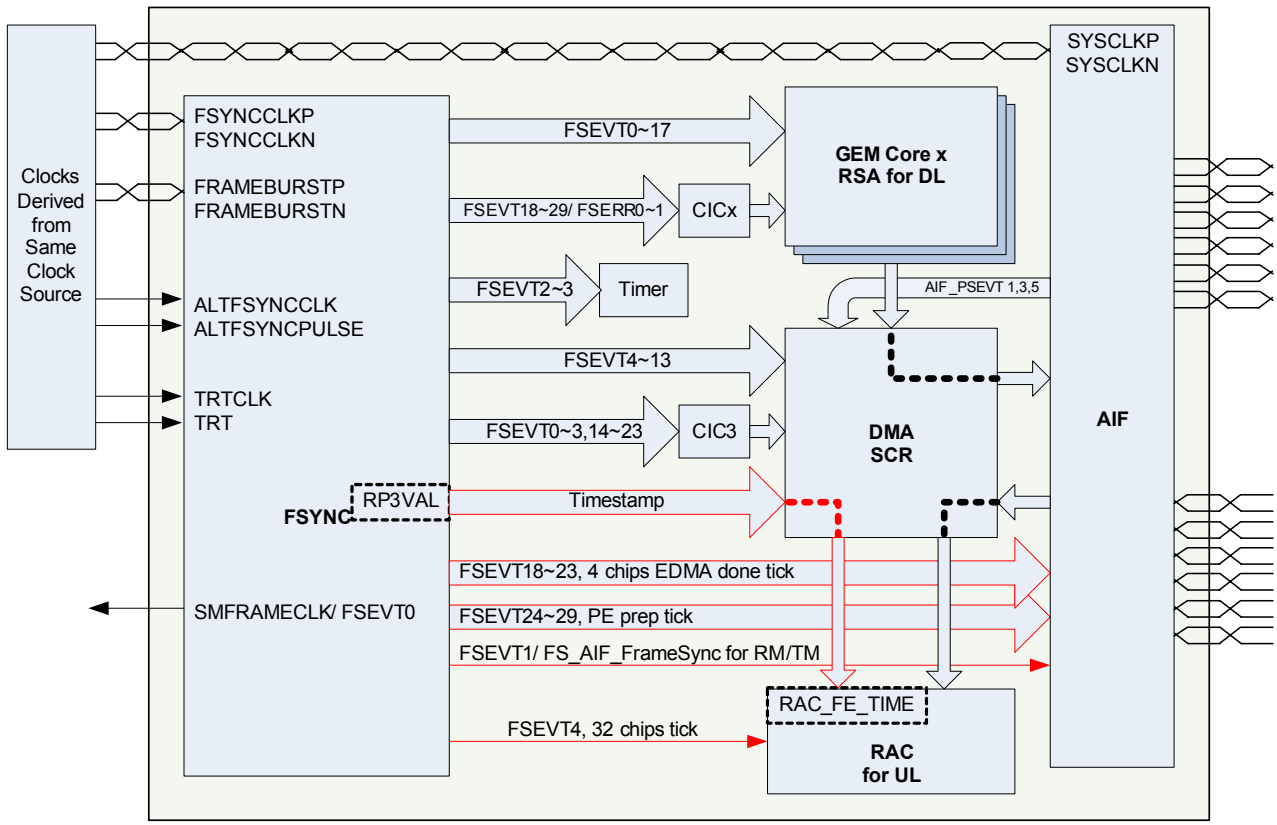
Figure 25 TCI648x FSync Block Diagram



FSync events 0~9 and events 18~29 are mask-based events; events 10~17 are count-based events. Their configuration is different, but for AIF application, AIF only cares about the period and start time offset of the event; both mask-based events and count-based events can be configured to achieve this functionality. See the *FSync User's Guide* for more information.

Timing is provided to all other subsystems via system events and VBUS reads. Figure 26 shows the block diagram of TCI6587/8 synchronization.

Figure 26 TCI6487/8 Synchronization



Almost all events are routed to the interrupt controller and EDMA controller; they can be configured to trigger any interrupt or EDMA channel. The EDMA channels for data transfer between AIF, DSP memory, and RAC are triggered by these events as well. For DL-type data, the event period is four chips; for UL_RSA-type data, the event period is eight chips. Table 11 lists the default synchronization event associated with each of the EDMA channels. For more information about EDMA events mapping, see section "8.5.1 EDMA3 Channel Synchronization Events" in the TMS320TCI6487/8 datasheet (SPRS358).

Table 11 EDMA3 Synchronization Events Associated with Fsync Module (Part 1 of 2)

EDMA Channel	Event	Event description
16	FSEVT4	Frame Synchronization Event 4
17	FSEVT5	Frame Synchronization Event 5
18	FSEVT6	Frame Synchronization Event 6
19	FSEVT7	Frame Synchronization Event 7
20	FSEVT8	Frame Synchronization Event 8
21	FSEVT9	Frame Synchronization Event 9
22	FSEVT10	Frame Synchronization Event 10
23	FSEVT11	Frame Synchronization Event 11
24	FSEVT12	Frame Synchronization Event 12

Table 11 EDMA3 Synchronization Events Associated with Fsync Module (Part 2 of 2)

EDMA Channel	Event	Event description
25	FSEVT13	Frame Synchronization Event 13
26	CIC3_EVT6	CIC_EVT_o [6] from Chip Interrupt Controller3 (Frame Synchronization Event 14)
27	CIC3_EVT7	CIC_EVT_o [7] from Chip Interrupt Controller3 (Frame Synchronization Event 15)
40	AIF_PSEVT1	PS inbound FIFO0 programmable_not_empty event
41	AIF_PSEVT3	PS inbound FIFO1 programmable_not_empty event
42	AIF_PSEVT5	PS inbound FIFO2 programmable_not_empty event

The related events can be generated flexibly by the Fsync module based on the system requirements. For example, EDMA channel 16 can be triggered every four chips to transfer outbound DL data from the DSP memory to the AIF outbound data buffer. EDMA channel 17 is triggered every eight chips to transfer UL_RSA data streams from the AIF RAM to DSP memory.

If the timing of every link is different, every link needs one event for outbound EDMA and one event for inbound EDMA, so six AIF links may need up to 12 events for triggering their data transfer EDMA. Only 10 events (FSEVT4~13) are directly routed to EDMA, other events can be routed to EDMA through CIC3, which may require extra configuration on CIC3. For more information about CIC, see chapter 7 “Interrupt Controller” of “TMS320C64x+ Megamodule Reference Guide”; configuration of CIC is similar as INTC.

If the timing of several AIF links is the same, they can share the same event for their EDMA transfer. In this case, the FSync event triggers the EDMA transfer for the first link, then chains to the EDMA for other links, i.e. the completion event of the EDMA previous link triggers the EDMA for the next link.

In the example with this application note, the EDMA for CS stream data is chained with the EDMA for CPRI control words. Because the CPRI control words are transferred along with the CS stream data, their timing is the same and they can be chained together and originally triggered by same FSync event.

Some events are hardwired to some modules, including:

- FSEVT1, Frame Synchronization for AIF, this is reference time point for all events timing for AIF, period= 1 frame.
- FSEVT18~23, EDMA done tick for PE (Protocol Encoder) of AIF, indicate EDMA has transferred data to AIF data buffer for PE to process, period= 4 chips, one event per link.
- FSEVT24~29, Frame Synchronization for PE of AIF, trigger PE to process a new frame, period= 1 frame, one event per link.
- FSEVT4, tick for RAC, trigger RAC to process 32 chips of data, period= 32 chips

Table 12 summarizes the FSync events for AIF and their configuration:

Table 12 FSync Events for AIF and Their Configuration

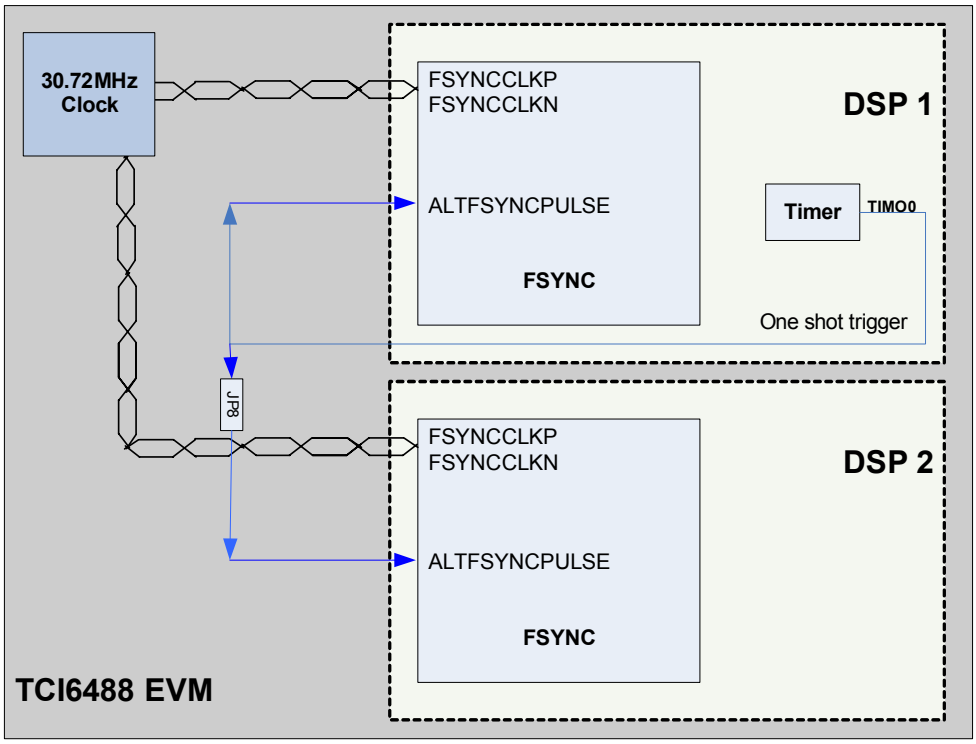
AIF Events	FSync Events #	Period	Comments
AIF Frame Synchronization	1	1 frame	One event for overall AIF as basic reference timing point
Outbound EDMA tick	Any event of 0~17	4 or 8 chips	One event for every link 4 chips period for DL data type 8 chips period for UL_RSA data type
Outbound EDMA done tick	18~23	4 chips	One event for every link
PE Frame Synchronization	24~29	1 frame	One event for every link
Inbound EDMA tick	Any event of 4~17	4 or 8 chips	One event for every link 4 chips period for DL data type 8 chips period for UL_RSA data type

To transfer antenna data between two DSPs, the two DSPs should be synchronized. Figure 27 shows the block diagram of the synchronization between two DSPs on the TCI6488 EVM. The FSync of two DSPs are driven by same differential clock source on the board.

On the Faraday EVM, the timer output pin TIMO0 of DSP1 is connected to the single end frame synchronization pin (ALTFSYNCPULSE) for FSyncs of DSP1. Jumper JP8 is a two-position jumper that allows the Sync pin of the TCI6488 DSP1 to be connected to the Sync pin of DSP2. You must short this jumper if DSP2 needs to synchronize with DSP1. To run the example in this application note, JP8 must be shorted.

In the example, DSP1 timer is configured to generate one shot pulse, which triggers both FSyncs on the DSP1 and DSP2 to run simultaneously. FSyncs on both DSPs are configured to run freely without automatic resynchronization to the external frame synchronization signal, so external periodical frame synchronization signal is not required for this test. This configuration is for testing on the EVM only; in a real system, an external periodical Frame synchronization signal should globally trigger FSync for keeping synchronization. This synchronization signal should have some deterministic timing relationship with the frame boundary of the basestation timing.

Figure 27 Synchronization of Two DSPs on the TCI6488 EVM



5 Antenna Interface Event Timings

Because the basestation is a timing-sensitive system, the AIF timing configuration is very important for the AIF to work correctly. Most timing offsets are configured through the offset of FSync events; the AIF TM (Tx MAC) timing offset is configured through the delta value in the AIF configuration registers and the AIF RM (Rx MAC) timing offset is configured through the pi value in the AIF configuration registers.

To define the AIF timing relationship, assume the following AIF daisy-chain topology:

The key AIF events relationship should be as follows:

Figure 28 AIF Events Relationship

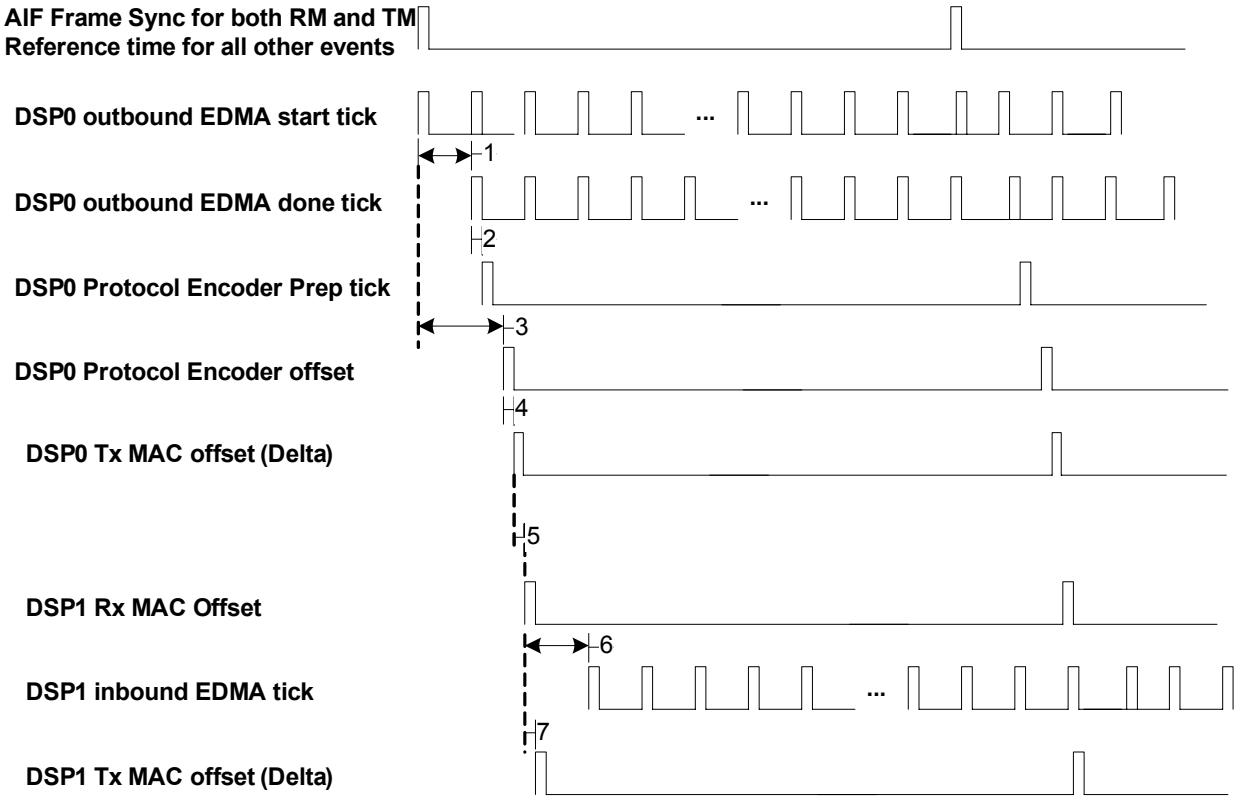


Table 13 describes the Minimal, Maximal, and recommended value for these timing parameters.

Table 13 AIF Events Timing Parameters

#	Parameter	Min	Max	Typical or Recommended	Comments
1	Delay between EDMA start and EDMA done tick	4 chips	32 chips	4 chips	**Max**= 32 chips because PE should start before the 32 chips outbound RAM overflow.
2	Delay between EDMA done and PE prep tick	0 chips	32 chips- (**Parameter** 1)	0 chips	**Max**= 32- (**Parameter** 1) chips because PE should start before the 32 chips outbound RAM overflow.
3	Delay between EDMA start and PE	(Burst Size) or (Burst Size)*2	32 chips	**Min** + 1 chips	PE should be started after at least one data burst is ready. (Burst size) is 4 chips for DL data type, 8 chips for UL_RSA data type. For CPRI or OBSAI non-aggregation link, Min= (Burst size). For OBSAI aggregation link, Min= 2*(Burst size).
4	Delay between PE and TM	22 or 28 vbus_clocks + 4 byte clocks	22 or 28 vbus_clocks + TM FIFO threshold	22 or 28 vbus_clocks + (4+TM FIFO threshold)/2 byte clock	22 vbus_clocks for OBSAI processing. 28 vbus_clocks for CPRI processing. (4 byte clocks) is the minimum number of bytes that must be written to the TM FIFO before TM starts, (TM FIFO threshold) is the programmed maximum value for TM FIFO full, the recommended value is in the middle of the minimum and maximum.
5	Delay between DSP0 TM and DSP1 RM	10 byte clocks		10 byte clocks	The delay should be composed of: TM Serdes processing delay + transfer line delay + RM Serdes processing dealy. The transfer line delay depends on customer's board layout, but it should be very small, around several ns. The RM/TM Serdes delay is about 10 byte clocks.
6	Delay between RM and inbound EDMA tick	15 vbus clocks + (Burst Size)	32 chips	**Min** + 1 chips	(Burst size) is 4 chips for DL data type, 8 chips for UL_RSA data type. **Max**= 32 chips because EDMA should start before the 32 chips inbound RAM overflow.
7	Redirection delay between RM and TM	10 or 16 vbus_clocks + 4 byte clocks	10 or 16 vbus_clocks + TM FIFO threshold	10 or 16 vbus_clocks + (4+TM FIFO threshold)/2 byte clock	10 vbus_clocks for OBSAI processing. 16 vbus_clocks for CPRI processing. (4 byte clocks) is the minimum number of bytes that must be written to the TM FIFO before TM starts, (TM FIFO threshold) is the programmed maximum value for TM FIFO full, the recommended value is in the middle of the minimum and maximum.
End of Table 13					

The typical steps to configure these timings on the first DSP of the daisy chain are:

1. Configure the offset of FSync event for outbound EDMA to start EDMA at the expected timing point.
2. Configure the offset of FSync events for outbound EDMA done and PE prep to be four chips after EDMA start.
3. Calculate the PE offset according to the delay between EDMA and PE; calculate the TM start offset according the delay between PE and TM; configure the ***delta*** in the TM configuration register with this value.

The typical steps to configure these timings on the next DSP in the daisy chain are:

1. Calculate the RM offset of the next DSP according to the delay between the TM of the previous DSP and the RM of the next DSP, configure the ***pi*** value and window size in the RM configuration registers to make the RM offset in the middle of the receive window, i.e. ***pi***= (RM offset) – (window size)/2.
2. Calculate the inbound EDMA start offset according to the delay between the RM and inbound EDMA, and configure the offset of the FSync event for inbound EDMA to start the EDMA at that timing point.
3. Calculate the TM offset of the next DSP according to the redirection delay between RM and TM; configure the ***delta*** value in the TM configuration register.
4. Calculate the PE offset of the next DSP according the delay between the TM and PE; calculate the outbound EDMA start offset according to the delay between the PE and EDMA; configure the offset of the FSync event for the outbound EDMA to start the EDMA at that timing point.
5. Configure the offset of FSync events for the outbound EDMA done and PE prep to be four chips after EDMA start.

The key point for the next DSP configuration is to make sure the streams from previous DSPs and this DSP arrive at the TM of the current DSP at the same time.

Additionally, the AIF input and output buffers are 32 chips long, and the timing difference per hop of the daisy chain is much less than 1 chip. So, the AIF buffer is large enough to support the accumulated daisy chain latency. Another approach to timing configuration is ensure that all DSPs in the chain to have aligned EDMA timing and to rely on the AIF buffer to support the timing offset, i.e. simply keep all DSPs with identical FSYNC module programming and simply change the Pi & Delta values of the AIF of different DSPs.

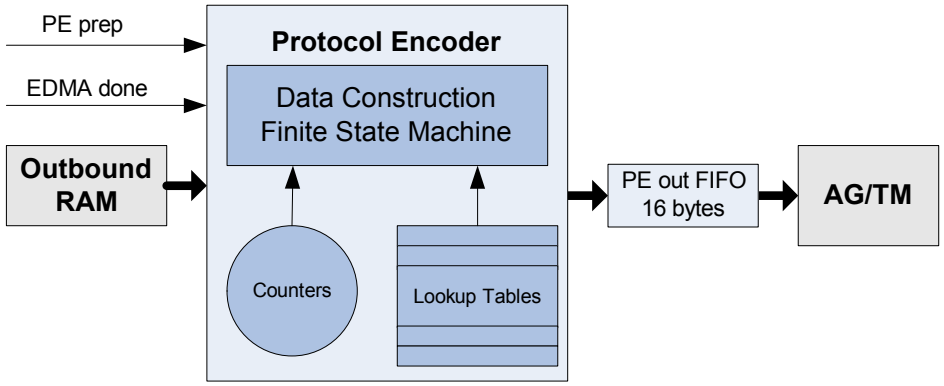
6 Antenna Interface Configuration

The AIF is very flexible; it provides many configuration options. The *Antenna Interface User's Guide* describes various AIF configurations. The following sections provide some complementary information about the configurations of several important modules of the AIF.

6.1 PE (Protocol Encoder) Configuration

Figure 29 shows the concept block diagram of the PE:

Figure 29 Protocol Encoder Block Diagram



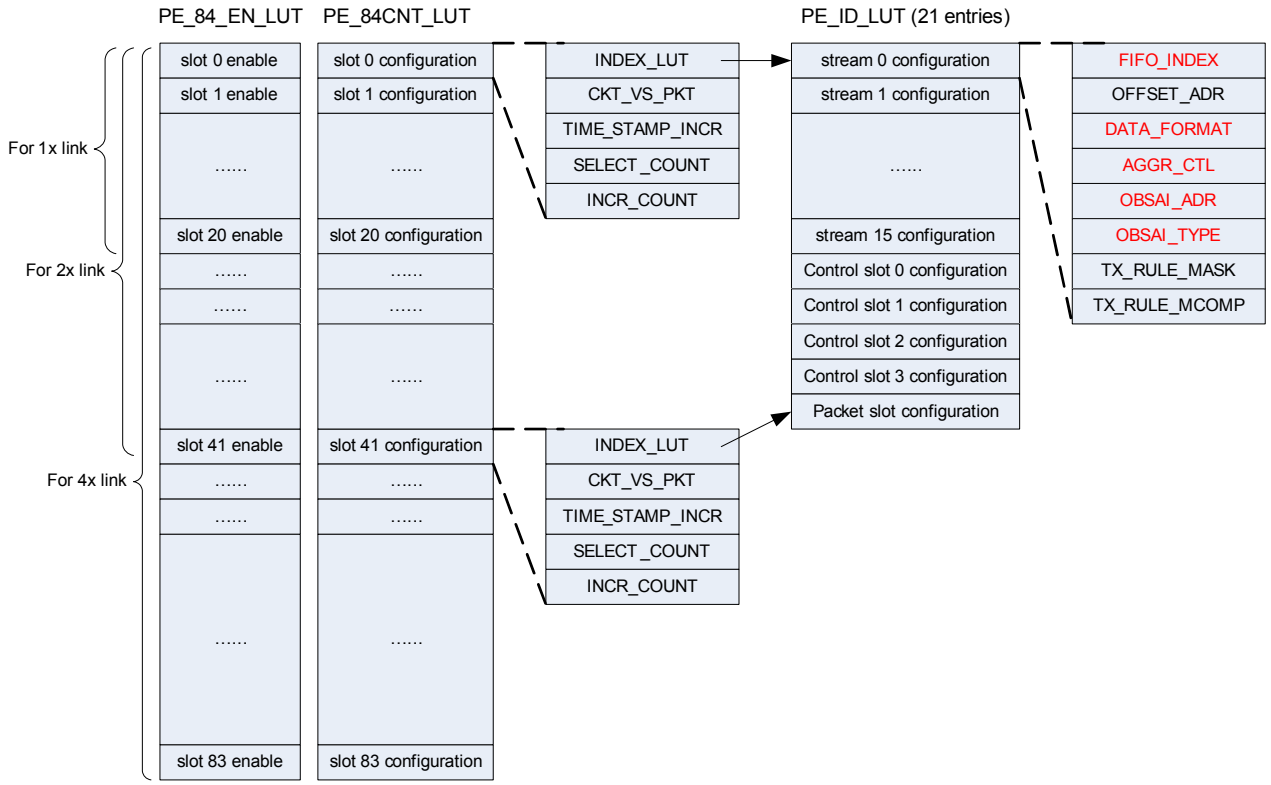
The PE prep tick from the FSync starts the PE, but real data processing happens only when:

1. There are data ready in the outbound RAM, which is indicated by the EDMA done tick from FSync, and
2. AG/TM is ready to process data, which is depends on the TM ***delta*** configuration
3. There is room in the PE out FIFO, which depends on the activities of the AG and TM.

The PE is configured through several look-up tables (LUT) as shown in Figure 30.

Two of the tables have 84 entries, which correspond to 84 message slots for OBSAI 4x modes; for 1x or 2x mode, only 21 or 42 entries are used. The PE_84_EN_LUT determines if one message slot should be enabled to transmit data; the PE_84CNT_LUT determines what data should be transmitted at that message slot. The 84 entries look-up tables are not used for CPRI mode.

Figure 30 PE Configuration Lookup Tables



Another 21 entries look-up table, PE_ID_LUT, describes 21 possible data type/formats. Normally,

- 0-15 indicate one of 16 possible streams (antenna carriers)
- 16-19 indicate one of four possible control slots
- 20 indicate packet-switched data in a data message slot

The look-up table mechanism gives more than practical flexibility to construct OBSAI messages. Any one of the 84 message slots can be used to transmit any of the 21 possible data. The INDEX_LUT filed in the PE_84CNT_LUT is a pointer to choose one of the 21 entries in the PE_ID_LUT.

The CKT_VS_PKT field in the PE_84CNT_LUT indicates the origin of the transmission data. If CKT_VS_PKT=1, PE reads data from an outbound circuit-switched RAM, the start point (initial read pointer) for PE to read data from the outbound circuit-switched RAM can be configured through OFFSET_ADR in PE_ID_LUT. You can configure it to support a special EDMA chaining operation, but for normal operation it is 0; If CKT_VS_PKT=0, PE read data from one of the outbound packet switch FIFOs, FIFO_INDEX field in PE_ID_LUT indicates which FIFO should be used for this slot.

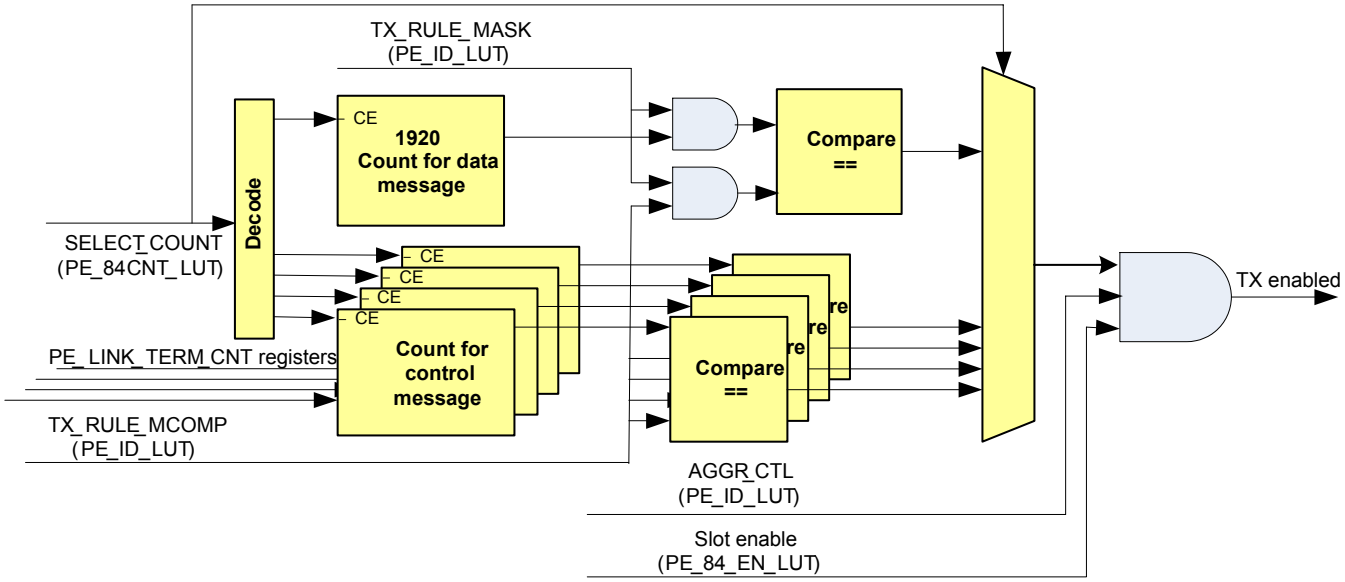
The OBSAI timestamp increments according to the TIME_STAMP_INCR field in PE_84CNT_LUT, if TIME_STAMP_INCR= 1, OBSAI timestamp increment in the next slot. This gives the programmer the flexibility to control the timestamp at any slot.

NOTE—The OBSAI header of a circuit-switched message is automatically constructed by the PE according to the configurations in these LUTs. The OBSAI packet-switched message must be constructed by application software and the full message, including header and payload, must be filled into the packet-switched FIFO. The PE does not modify any contents of the packet-switched message from FIFO and directly transports it through the AIF. Timestamp checking is performed only on OBSAI circuit-switched data. OBSAI packet-switched transfer does not need timestamps. So, application software can fill any value in the timestamp field of packet-switched message.

The SELECT_COUNT, INCR_COUNT in PE_84CNT_LUT, and TX_RULE_MASK, TX_RULE_MCOMP in PE_ID_LUT are used to further gate the OBSAI message transmission in a slot.

Figure 31 shows how the PE transmission enables rules for OBSAI mode. There are 5 counters, one for data message, four for control messages. The period of the counter for a data message is 1920, the period of the counters for control message is configured through PE_LINK_TERM_CNT registers. The counters are selected and increased according the SELECT_COUNT, INCR_COUNT in PE_84CNT_LUT. Data messages may be transmitted in a slot if ((count value)&mask == (compare value)&mask). Control messages may be transmitted in a slot if ((count value) == (compare value)). The enable signal is further qualified by PE_84_EN_LUT and AGGR_CTL in PE_ID_LUT to generate the final enable signal.

Figure 31 PE Transmission Enable Rules



Because the PE lookup table mechanism gives greater than practical flexibility, most of these features are used only for very special or complicated cases. So, for simple and normal application, you can use the default configuration of the CSL (Chip Support Library), that is,

- No data message is filtered by mask/compare block
- Slot n transmit data is described by entry (n%21) of the PE_ID_LUT. For example, slot 0 transmit data is described by entry 0 of PE_ID_LUT, slot 1 transmit data is described by entry 1 of PE_ID_LUT, slot 21 transmit data is described by entry 0 of PE_ID_LUT again.

The parameters you may need to change include: FIFO_INDEX, DATA_FORMAT, AGGR_CTL, OBSAI_ADR, OBSAI_TYPE.

For CPRI mode, only 17 of the 21 entries of PE_ID_LUT are used

- Entries 0-15 are fixed to indicate one of 16 possible streams (antenna carriers)
- Entry 16 is fixed to describe the control word

Additionally, for CPRI, only OFFSET_ADR, DATA_FORMAT and AGGR_CTL fields of PE_ID_LUT are used, and “TX enabled” is only qualified by AGGR_CTL.

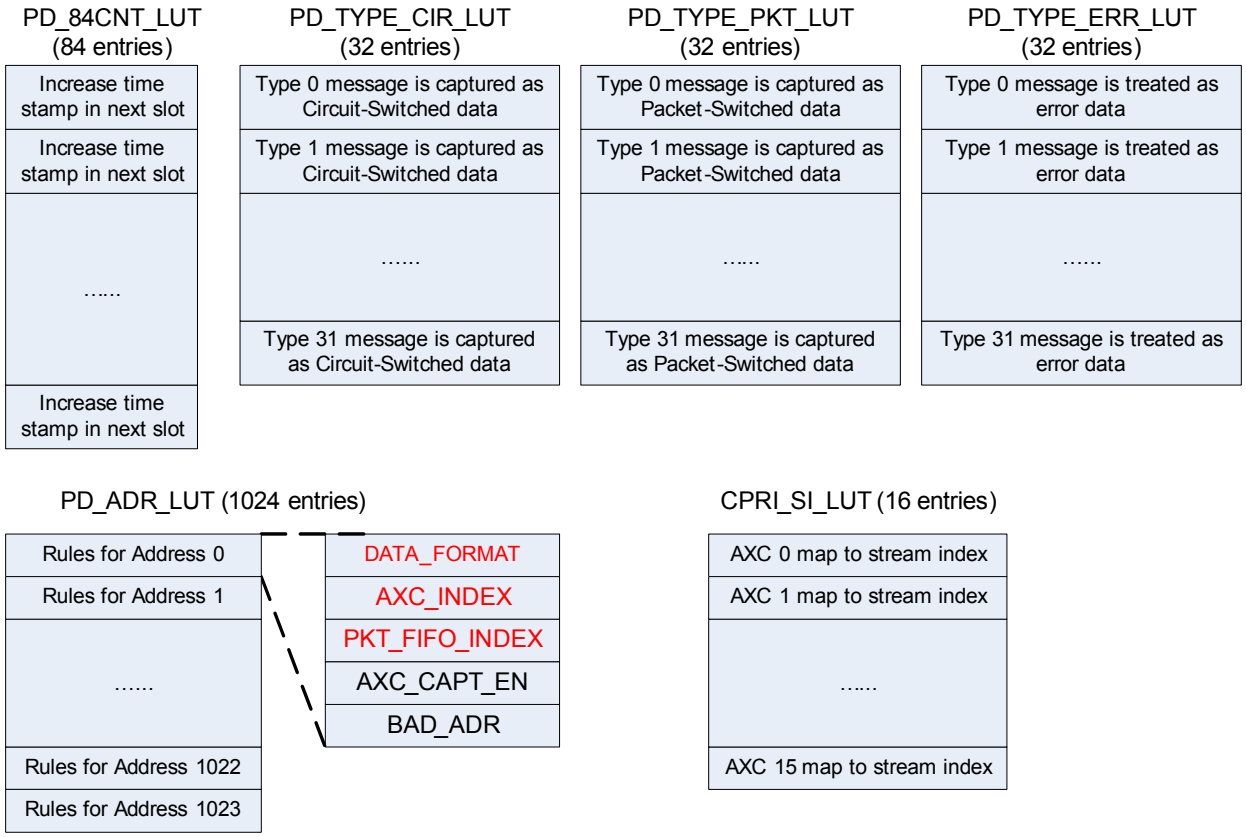
For CPRI mode, the AIF can insert control words automatically if the application does not care the control words.

If the application needs to transfer information through control words, the application should construct a hyper control frame and fill in the CPRI control words RAM. The sync&timing field will be checked by the AIF; all other fields, such as frame number and L1 inbound signaling, are ignored or replaced by the AIF because the AIF will automatically generate them. All other contents in control frame will be transferred without changing and checking.

6.2 PD (Protocol Decoder) Configuration

The PD is configured through several look-up tables (LUT) as follows:

Figure 32 PD Configuration Lookup Tables



The PD_84CNT_LUT controls the increment of the timestamp in any slot for OBSAI. The 84 different bits in the 84CNT_LUT each represent one of 84 message slots for four consecutive OBSAI message groups. When a corresponding bit is set to “1”, the timestamp counter will increment at the end of the current OBSAI message.

The PD_TYPE_XXX_LUT controls the treatment of the OBSAI message type. The OBSAI protocol defines five bit types, so every LUT has 32 entries. When a corresponding bit is set to “1” in entry *n* of PD_TYPE_CIR_LUT, the message with type *n* will be captured in the AIF inbound circuit-switched buffer. When a corresponding bit is set to “1” in entry *n* of PD_TYPE_PKT_LUT, the message with type *n* will be captured in the AIF inbound packet-switched FIFO. When a corresponding bit is set to “1” in entry *n* of PD_TYPE_ERR_LUT, the message with type *n* will be captured in the AIF inbound error FIFO. When entry *n* in all of the three PD_TYPE_XXX_LUT are “0”, the message with type *n* will be discarded by the AIF.

The PD_ADR_LUT controls the data format and receiving rules for every OBSAI address. PE rules are based on timeslot; the data format and stream number are configured for every slot. PD rules are based on address and type of OBSAI message; the data format and stream number are configured for every address. If the transmitter (PE) uses same OSBAI address for all timeslots, the receiver (PD) will receive all data into same stream buffer. This may be unexpected, so, normally, different OBSAI addresses should be configured for different transmitter timeslots.

OBSAI protocol defines a 13-bit address, but the AIF implements only a 10-bit address (corresponding to 1024 PD_ADR_LUT entries) to save memory space for the address look-up table. For flexibility, any 10 of the 13 bits of the incoming message can be used to select PD_ADR_LUT entries, which can be configured through the PD_ADR_MUX_SEL_CFG register or through the addressMask parameter of the CSL (Chips Support Library). Normal configuration is addressMask= 0x3FF, which means only the lower 10 bits are used.

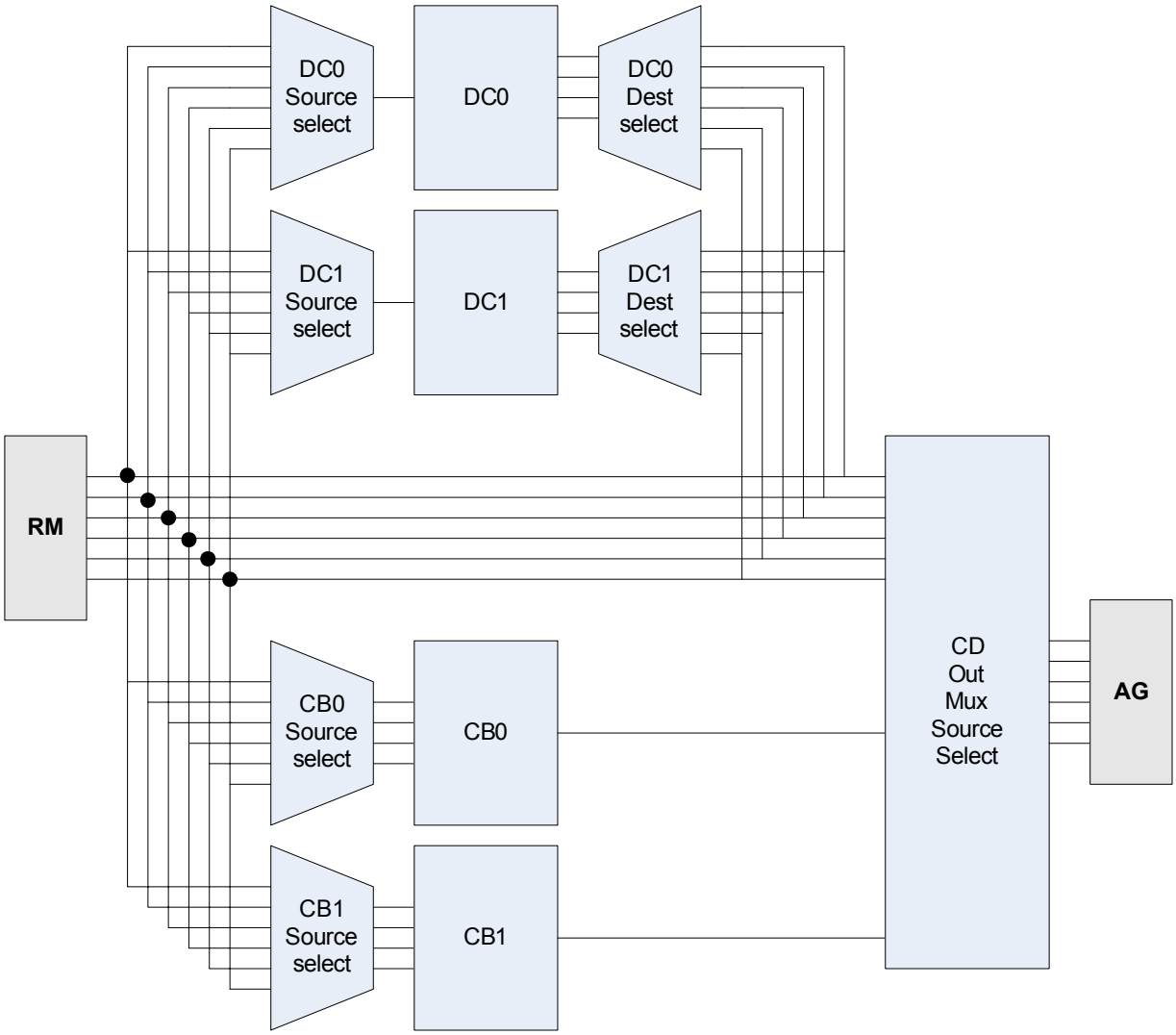
PD_84CNT_LUT, PD_TYPE_XXX_LUT, and PD_ADR_LUT are used only for OBSAI. The CPRI_SI_LUT is for CPRI only; it maps the 16 AxC to any of 16 stream locations in AIF inbound RAM. The purpose is same as AXC_INDEX in PD_ADR_LUT for OBSAI.

This look-up table mechanism gives more than practical flexibility; most of these features are used for very special or complicated cases only. So, for normal operation, use the default configuration of the CSL (Chip support library). The fields you may need to change are PD_TYPE_XXX_LUT, DATA_FORMAT, AXC_INDEX and PKT_FIFO_INDEX.

6.3 CD (Combiner/Decombiner) Configuration

Figure 33 shows the block diagram of the Combiner/Decombiner.

Figure 33 Combiner/Decombiner Block Diagram



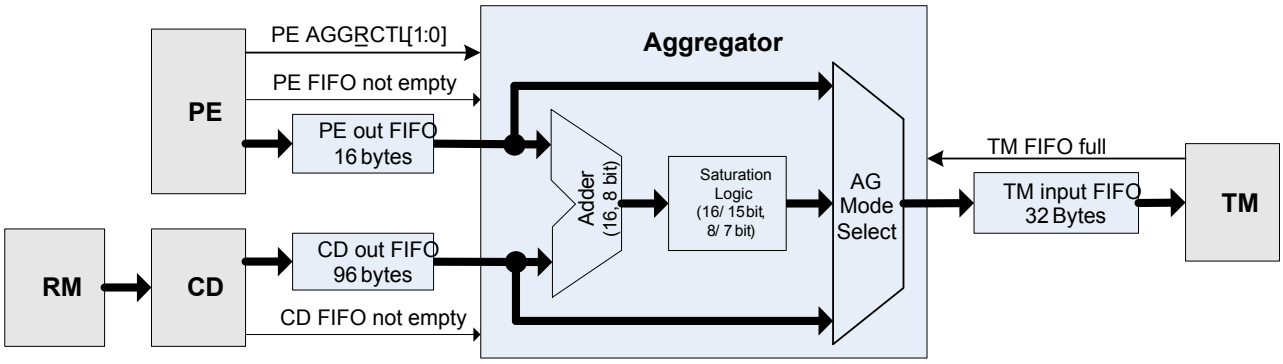
The main configuration for the CD are source/destination selections; most of these are straightforward. If decombining, each output mux source select used for decombining should select the same input link as the Decombiner. For example, if DC1 decombines input link 1 to link 2 and 3, the CD out mux source selection should be 1 instead of 2 or 3.

For combining, each of the two combiner functions provides its own programmable cb_fs_offset (like pi_offset in the RM) and valid_data_wind (valid data window). The Master Frame indicator (K28.7 byte) of each link to be combined must fall within the valid data window following the cb_fs_offset relative to frame sync.

6.4 AG (Aggregator) Configuration

Figure 34 shows the block diagram of the Aggregator.

Figure 34 Aggregator Block Diagram



The AG is throttled by

- The availability of space in the TM FIFO
- The availability of PE data in PE FIFO
- The availability of CD data in CD FIFO

AG works only when all the conditions above are met.

Saturation logic detects addition overflow and replaces the overflowed value with a value based on the selected bit width as follows:

- 7 bit data: ±63
- 8 bit data: ±127
- 15 bit data: ±16,383
- 16 bit data: ±32,767

NOTE—The saturation logic clips the AG output above the minimum negative value for each bit width (e.g.–127 instead of –128 for 8-bit data) to prevent a DC offset from occurring.

The main configurations for AG include AG mode selection and PE AGGR_CTL. If AG mode selects data from the CD only, the PE AGGR_CTL configuration has no effect. If AG mode selects to aggregate data from the CD and the PE, the PE AGGR_CTL controls the behavior of aggregator byte by byte for CPRI or message by message for OBSAI.

If AG mode selects data from the PE only, the AG runs the PE data through the summation and saturation logic in the Data Path Processor when the PE AGGR_CTL is “ADD”. In this case, the CD input to the summation logic is set to 0. This clipping can eliminate any DC offset in circuit-switched data from the PE. A PE AGGR_CTL of “INSERT” will allow the PE data to bypass the summation logic and avoid clipping. This path can be used for packet-switched data and CPRI control words. Only circuit-switched data can be aggregated.

6.5 Using the Example Codes with this Application Note

The steps to run the codes are:

1. Press the PORz Button on the EVM to Reset it. This is not required for first time after power up, but it is required to rerun the program without power cycling.
2. Conncet the CCS to the DSP.
3. Load AIF_F1 to core 0 of F1 (Faraday DSP1).
4. Load AIF_F2 to core 0 of F2 (Faraday DSP2).
5. Run F2, then run F1.
6. Check the LOG0 of DSP/BIOS on F1 for test result; the configuration and status information for every link are also shown in the LOG.

   The correct result at the end of the LOG should look like the following for OBSAI mode:

   ```
   Link1

     transferred 15000 CS data blocks, 15360000 chips CS data
     received 300 error CS data blocks, 14700 consecutive correct CS data blocks
     transferred 190000 PS messages, received 188000 PS messages
   ```

   The correct result at the end of the LOG should look like the following for CPRI mode:

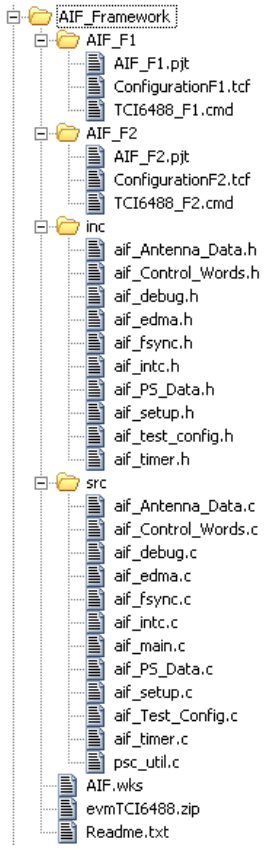
   ```
   Link1

     transferred 15000 CS data blocks, 11520000 chips CS data
     received 150 error CS data blocks, 14850 consecutive correct CS data blocks
     transferred 15000 CPRI control hyper frame, received 14850 good CPRI
     control hyper frame
   ```

NOTE—The 300 or 150 error data blocks occur at the beginning during link synchronization; this is normal.

The directory structure of the projects is as follows:

Figure 35 Directory Structure of Example Codes



There are two projects for two DSPs, the project in .\ AIF_F1 runs on F1 (Faraday DSP 1), the project in .\ AIF_F2 runs on F2 (Faraday DSP 2).

Both projects share the same codes files in the .\src and .\inc directories. The codes for different DSPs are differentiated by macro definition as follows:

```
#ifdef _EVM_F1
#include "ConfigurationF1cfg.h"
#else
#include "ConfigurationF2cfg.h"
#endif
```

The project for F1 is built with _EVM_F1 macro defined; the project for F2 is built with _EVM_F2 macro defined.

The contents of the codes are listed in Table 14.

Table 14 Source Files of the Example Codes

Files	Descriptions
aif_main	main() function and the top level scheduling codes
aif_Test_Config	This file includes basic configuration structure for test. User can modify the basic parameters to change the test mode to verify most functionalities of AIF. The codes in this file calculate many other parameters for AIF based on the basic configuration.
aif_setup	Setup AIF registers according the configuration parameters from aif_Test_Config module.
aif_fsync	Setup FSync according WCDMA frame structure; setup FSync events according the timing parameter from aif_Test_Config module. The FSync interrupt service routines increase the counter of FSync events.
aif_edma	Setup EDMA PaRAM and registers according AIF data type for all links, EDMA interrupt service routine switch ping-pong buffer and trigger DSP processing.
aif_Antenna_Data	Process antenna data. For this test, received data are compared to transmitted data, statistic the correctly transferred data or error. Then fill new data into transmit buffer for next transmitting.
aif_PS_Data	Process packet-switched data. For this test, transmitter always transfers PS data if the outbound PS FIFO is not full, the transmission EDMA for PS data is triggered by CPU manually when it is free; When AIF inbound FIFO receives messages, it triggers EDMA transfer, the EDMA ISR process the received messages, counts the received messages for every link according its address.
aif_Control_Words	Process CPRI control words. For this test, received data are compared to transmitted data, statistic the correctly transferred data or error. Then fill new data into transmit buffer for next transmitting.
aif_timer	setup timer to generate one-shot pulse to trigger the FSync of two DSPs, thus trigger the AIF of two DSPs to work synchronously.
aif_intc	setup interrupt controller to enable FSync ,EDMA and timer interrupts.
Psc_util	Various functions to enable specific module in DSP including AIF.

You can change the initialization values of the basic configuration structure in aif_Test_Config.c as follows and rebuild the project to verify most functionalities of AIF.

```
//Change this parameter to switch between OBSAI/CPRI test
CSL_AifLinkProtocol aifProtocol= CSL_AIF_LINK_PROTOCOL_CPRI;

/*CPRI control word mode*/
CSL_AifCpriCtrlWMode   cpriCtrlWordMode= CSL_AIF_CPRI_CTRL_WORD_READ_FROM_RAM;

/*Change following basic parameters to switch between different test modes
Please note, AIF link 1~4 is connected between the two DSP on EVM,
AIF link 0 on the EVM is connet to SMA connector,
AIF link 5 on the EVM is connected to AMC interface
So, without external connection link 0 and 5 can only be used for internal loopback
test
Please note, the DSP core can't process all links with 4x link rate simultaneously
*/
AifConfig aifConfig[6]=
{
    /*without external connection, link 0 can only be used for internal loop back
    test*/
    {
        0,          /*linkIndex, do not change this field*/
        0,          /*Line Enable: 1=enable, 0=disable*/
        1,          /*linkRate: 1=1x; 2=2x; 4= 4x*/

        TEST_PATH_INTERLNAL_LOOPBACK, /*test data path*/

        CSL_AIF_LINK_DATA_TYPE_DL, /*outboundDataType*/
        CSL_AIF_DATA_WIDTH_16_BIT, /*outboundDataWidth*/
        CSL_AIF_LINK_DATA_TYPE_DL, /*inboundDataType*/
        CSL_AIF_DATA_WIDTH_16_BIT, /*inboundDataWidth*/

        16,                        /*threshTxMacFifo*/
        200                        /*RX MAC frame sync window*/
    },
    /*Configuration for link 1*/
    {
```

```
            1,          /*linkIndex, do not change this field*/
            1,          /*Line Enable: 1=enable, 0=disable*/
            1,          /*linkRate: 1=1x; 2=2x; 4= 4x*/

            TEST_PATH_INTERLNAL_LOOPBACK, /*test data path*/

            CSL_AIF_LINK_DATA_TYPE_UL_RSA, /*outboundDataType*/
            CSL_AIF_DATA_WIDTH_8_BIT, /*outboundDataWidth*/
            CSL_AIF_LINK_DATA_TYPE_UL_RSA, /*inboundDataType*/
            CSL_AIF_DATA_WIDTH_8_BIT, /*inboundDataWidth*/

            16,                       /*threshTxMacFifo*/
            200                       /*RX MAC frame sync window*/
        },
        /*Configuration for link 2*/
        {
            2,          /*linkIndex, do not change this field*/
            1,          /*Line Enable: 1=enable, 0=disable*/
            2,          /*linkRate: 1=1x; 2=2x; 4= 4x*/

            TEST_PATH_REDIRECTION, /*test data path*/

            CSL_AIF_LINK_DATA_TYPE_DL, /*outboundDataType*/
            CSL_AIF_DATA_WIDTH_16_BIT, /*outboundDataWidth*/
            CSL_AIF_LINK_DATA_TYPE_DL, /*inboundDataType*/
            CSL_AIF_DATA_WIDTH_16_BIT, /*inboundDataWidth*/

            16,                       /*threshTxMacFifo*/
            200                       /*RX MAC frame sync window*/
        },
        /*Configuration for link 3*/
        {
            3,          /*linkIndex, do not change this field*/
            1,          /*Line Enable: 1=enable, 0=disable*/
            4,          /*linkRate: 1=1x; 2=2x; 4= 4x*/

            TEST_PATH_REDIRECTION, /*test data path*/

            CSL_AIF_LINK_DATA_TYPE_UL_RSA, /*outboundDataType*/
            CSL_AIF_DATA_WIDTH_8_BIT, /*outboundDataWidth*/
            CSL_AIF_LINK_DATA_TYPE_UL_RSA, /*inboundDataType*/
            CSL_AIF_DATA_WIDTH_8_BIT, /*inboundDataWidth*/

            16,                       /*threshTxMacFifo*/
            200                       /*RX MAC frame sync window*/
        },
        /*Configuration for link 4*/
        {
            4,          /*linkIndex, do not change this field*/
            1,          /*Line Enable: 1=enable, 0=disable*/
            2,          /*linkRate: 1=1x; 2=2x; 4= 4x*/

            TEST_PATH_REDIRECTION_AGGREGATION, /*test data path*/

            CSL_AIF_LINK_DATA_TYPE_DL, /*outboundDataType*/
            CSL_AIF_DATA_WIDTH_16_BIT, /*outboundDataWidth*/
            CSL_AIF_LINK_DATA_TYPE_DL, /*inboundDataType*/
            CSL_AIF_DATA_WIDTH_16_BIT, /*inboundDataWidth*/

            16,                       /*threshTxMacFifo*/
            200                       /*RX MAC frame sync window*/
        },
        /*without external connection, link 5 can only be used for internal loop back
        test*/
        {
            5,          /*linkIndex, do not change this field*/
            0,          /*Line Enable: 1=enable, 0=disable*/
            1,          /*linkRate: 1=1x; 2=2x; 4= 4x*/

            TEST_PATH_INTERLNAL_LOOPBACK, /*test data path*/

            CSL_AIF_LINK_DATA_TYPE_UL_RSA, /*outboundDataType*/
            CSL_AIF_DATA_WIDTH_8_BIT, /*outboundDataWidth*/
            CSL_AIF_LINK_DATA_TYPE_UL_RSA, /*inboundDataType*/
            CSL_AIF_DATA_WIDTH_8_BIT, /*inboundDataWidth*/

            16,                       /*threshTxMacFifo*/
            200                       /*RX MAC frame sync window*/
        }
    };
```

Because of L2 RAM bandwidth and CPU loading limitation, more than three 4x full-duplex links can not be simultaneously run for this test. But if you mix 1x, 2x, 4x link rate for different links, you can run more than three links simultaneously.

To rebuild the project with your new configurations, you may need to:

1. Change the CSL include directory
2. Install a DSP/BIOS template used by this project. Please unzip evmTCI6488.zip in the directory of the project, then copy the “evmTCI6488” directory to .\CCStudio_v3.3\bios_5_31_02\packages\ti\platforms”.

7 References

1. *TMS320TCI6487/8 Antenna Interface User's Guide* (SPRUEF4)
2. *TMS320TCI6488 Enhanced DMA (EDMA3) Controller User's Guide* (SPRUEE9)
3. *TMS320TCI6487/8 Frame Synchronization User's Guide* (SPRUEF5)
4. *TMS320C64x+ Megamodule Reference Guide* (SPRU871)
5. *TMS320TCI6487/8 datasheet* (SPRS358)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265