KeyStone Architecture
Turbo Encoder Coprocessor (TCP3e)

User Guide

Literature Number: SPRUGS1
November 2010

Release History

Release	Date	Chapter/Topic	Description/Comments
1.0	November 2010	All	Initial Release

Contents

Chapter 4

List of Tables

List of Figures

Preface

About This Manual

3G and 4G wireless systems require a turbo encoder as one of their forward error correction algorithms. A very cost-effective and synergistic solution has been designed using a Texas Instruments DSP with turbo encoder coprocessor. The Turbo-Encoder Coprocessor 3 (TCP3E) is a programmable peripheral for encoding of 3GPP0, LTE0 and WiMAX00 turbo codes integrated into the Texas Instruments DSP device. This document describes the operation and programming of the TCP3E.

Notational Conventions

This document uses the following conventions:

- Commands and keywords are in **boldface** font.
- Arguments for which you supply values are in *italic* font.
- Terminal sessions and information the system displays are in `screen font`.
- Information you must enter is in **`boldface screen font`**.
- Elements in square brackets ([ ]) are optional.

Notes use the following conventions:

Note—Means reader take note. Notes contain helpful suggestions or references to material not covered in the publication.

The information in a caution or a warning is provided for your protection. Please read each caution and warning carefully.

CAUTION—Indicates the possibility of service interruption if precautions are not taken.

WARNING—Indicates the possibility of damage to equipment if precautions are not taken.

Related Documentation from Texas Instruments

C66x CPU and Instruction Set Reference Guide	SPRUGH7
C66x CorePac User Guide	SPRUGW0
TMS320C6000 Programmer's Guide	SPRU198
TMS320C6000 Code Composer Studio Tutorial	SPRU301

Trademarks

Chapter 1

Introduction

1.1 About This Manual

3G and 4G wireless systems require a turbo encoder as one of their forward error correction algorithms. A very cost-effective and synergistic solution has been designed using a Texas Instruments DSP with turbo encoder coprocessor. The Turbo-Encoder Coprocessor 3 (TCP3E) is a programmable peripheral for encoding of 3GPP0, LTE0 and WiMAX00 turbo codes integrated into the Texas Instruments DSP device. This document describes the operation and programming of the TCP3E.

1.2 Notation Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labelled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.
- The term “word” describes a 32-bit value.

1.3 Related Documentation From Texas Instruments

The following documents describe the C6000™ devices and related support tools. Copies of these documents are available on the Internet at www.ti.com. Tip: Enter the literature number in the search box provided at www.ti.com.

- SPRU189 - TMS320C6000 DSP and Instruction Set Reference Guide. Describes the DSP architecture, pipeline, instruction set, and interrupts for the TMS320C6000 digital signal processors (DSPs).
- SPRU198 - TMS320C6000 Programmer’s Guide. Describes ways to optimize C and assembly code for the TMS320C6000™ DSPs and includes application program examples.
- SPRU301 - TMS320C6000 Code Composer Studio Tutorial. Introduces the Code Composer Studio™ integrated development environment and software tools.
- SPRU321 - Code Composer Studio Application Programming Interface Reference Guide. Describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.
- SPRU871 - C66x CorePac Reference Guide. Describes the TMS320C64x+ digital signal processor (DSP). Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

1.4 Trademarks

C6000, TMS320C6000, Code Composer Studio are trademarks of Texas Instruments.

All other trademarks are the property of their respective owners.

1.5 Purpose of the Peripheral

Wireless base stations currently use cost-efficient DSPs for their implementations. The system cost of the base station is a function of the number of channels that the DSP can support. As the wireless systems are converted from second generation to third and fourth generation systems, the system complexity has increased by an order of magnitude. The 3G wireless systems require a turbo encoder as one of its forward error correction algorithms. A very cost-effective and synergistic solution has been designed using a DSP and a turbo coprocessor (TCP3E).

The Turbo-Encoder Coprocessor 3 (TCP3E) is a programmable peripheral for encoding of 3GPP0, LTE0 and WiMAX00 turbo codes, integrated into the Texas Instruments DSP device. The inputs into the TCP3E are information bits, and the outputs are encoded systematic and parity bits.

Turbo codes used in 3G standards 00 are based on a parallel concatenation of Recursive Systematic Convolutional Codes (RSCC), each with a binary input. The optional Convolutional Turbo Code (CTC) included in the IEEE 802.16e standard 00 is based on duo-binary turbo codes.

The primary focus of this document is to describe how to use the TCP3E.

Note—Unless otherwise noted, references to 3GPP standards in this document imply support for the 3GPP standard up to Release 7 0. This includes WCDMA, HSUPA, HSUPA+, and TD_SCDMA.

1.6 Terminology Used in This Document

This section can be used to define any terminology or acronyms used in the document. Defining terms that are commonly known is not necessary, but some application-specific terms may need to be defined, especially for a new user.

Term	Definition
3GPP	3rd Generation Partnership Project
CRC	Cyclic Redundancy Check
CTC	Convolutional Turbo Code
LLR	Log Likelihood Ratio
LTE	Long Term Evolution
MAP	Maximum A Posteriori
RSCC	Recursive Systematic Convolutional Codes
SNR	Signal to Noise Ratio
TCP3E	Turbo-Encoder Coprocessor 3
WiMAX	Worldwide Interoperability for Microwave Access

1.7 Features

The TCP3E provides the following features:

- 3GPP, Wimax and LTE encoding
- Code rate: 1/3
- On-the-fly interleaver table generation
- Dual encode engines with input and output memories for increased throughput
- Programmable input and output format within a 32-bit word
- Block sizes supported: 40 to 8192
- Tail biting for Wimax
- CRC Encoding for LTE
- Emulation suspend support
- Runs in parallel with DSP

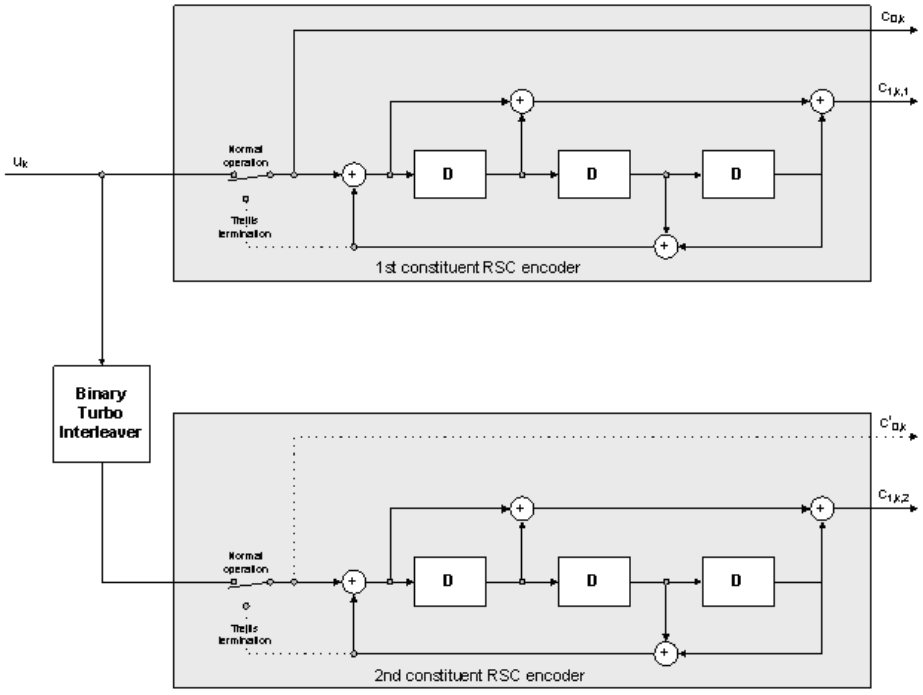
1.8 Theory

1.8.1 Binary Turbo Encoder

Binary turbo codes are block codes: a code block of *K* information bits is encoded and in the process, the output block of *N (N>K)* encoded bits is generated (a code word or code block). Binary turbo codes have been adopted in several 3GPP standards.

The encoder consists of two Recursive Systematic Convolutional (RSC) encoders. Each RSC encoder takes as an input one information bit, u_k, at a time and generates one parity bit (native code rate 1/3 for 3GPP), Figure 1-1.

Figure 1-1 Block Diagram of the Binary Turbo Encoder



The first encoder operates on the incoming binary data sequence u_k ($k = 1,2,...,K$), whereas the second encoder operates on the interleaved input sequence. The original input bit appears unchanged at the encoder output and is denoted as $c_{0,k}$ (systematic bit). In the case of the code rate 1/3, the turbo encoder for each input bit, u_k, generates three output bits: $c_{0,k}$, $c_{1,k}$, $c_{2,k}$. The interleaver design depends on the particular standard.

The initial trellis state for both constituent encoders is set to be the zero-state. There is also a provision in the standards to drive the last state of the trellis into zero-state. This is accomplished by sending some additional bits after all the information bits have been encoded. The dotted lines in Figure 1-1 become active during the trellis termination period. During the first three bit-clock cycles of the trellis termination period the control switch (Figure 1-1) of the first RCS encoder is kept in the lower position and the output of the first encoder is sent to the line. During the second three bit-clock cycles of the trellis termination period the control switch of the second RCS encoder is kept in the lower position and the output of the second encoder is sent to the line. The actual pattern of the encoded bits sent out during the trellis termination period depends on the particular standard.

The encoder is normally followed by a puncturing or a rate matching block which determines how the encoded bits will be used by the following block of the transmitter.

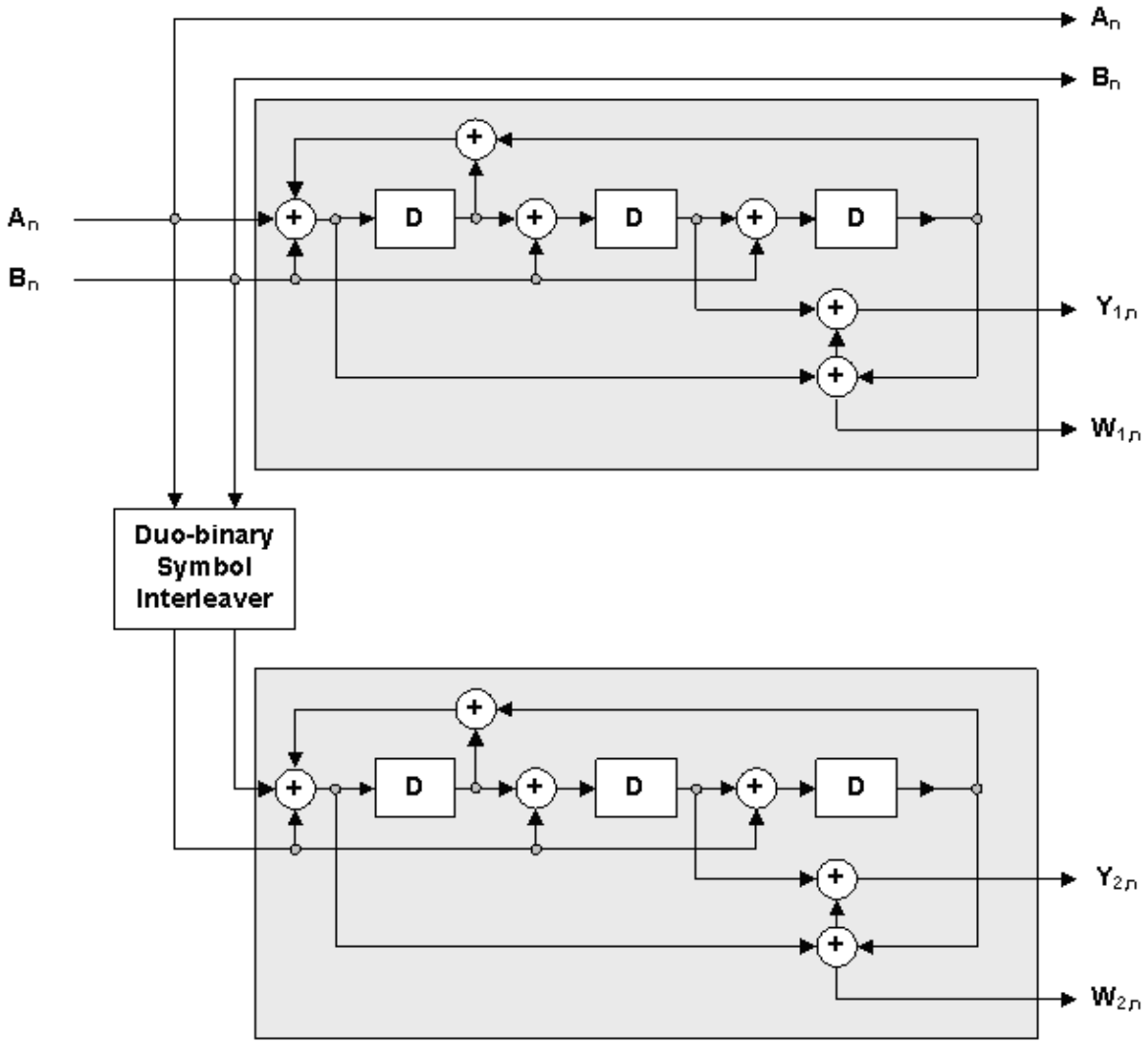
1.8.2 Duo-Binary Turbo Encoder

Duo-binary turbo codes are block codes: a code block of *K* information bits (K/2 information duo-binary symbols) is encoded and in the process the output block of *N (N>K)* encoded bits is generated (a codeword). They have been adopted in several telecommunication standards including IEEE 802.16 (WiMAX). Duo-binary turbo codes have two distinctive characteristics:

- Operation on bit pairs (duo-binary symbols) rather than on individual bits
- Utilization of the circular (tail-biting) Recursive Systematic Convolutional (RSC) component codes.

The encoder consists of two circular RSC encoders. The first RSC encoder takes as an input a pair of information bits, (A_n,B_n), representing a duo-binary symbol, at a time and generates two output parity bits ($Y_{1,n}$,$W_{1,n}$). The second RSC encoder takes as an input interleaved duo-binary symbol (A_n,B_n), and generates two output parity bits ($Y_{2,n}$,$W_{2,n}$). This encoding process is shown in Figure 1-2.

Figure 1-2 Block Diagram of the Duo-Binary Turbo Encoder



1.9 References

1. 3GPP TS 25.212 V7.5.0 (2007-05): “3GPP Technical Specification Group Radio Access Network; Multiplexing and channel coding (FDD) (Release 7); Section 4.2.3.2 Turbo coding”.
2. 3GPP TS 36.212 V2.0.0(2007-09): “3GPP Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Multiplexing and channel coding (Release 8); Section 5.1.3.2 Turbo coding”.
3. IEEE 802.16-2004: “Part16: Air Interface for Fixed Broadband Wireless Access Systems; 8.2.1.2.4 Convolutional Turbo Codes”.
4. IEEE P802.16-2004/Cor1/D5: “Corrigendum to IEEE Standard for Local and Metropolitan Area Networks: Part16: Air Interface for Fixed Broadband Wireless Access Systems; 8.4.9.3 Convolutional Turbo Codes”.
5. Change request to [2] - R1-075111 36.212 CR001; 3GPP TSG-RAN WG1 #51 Jeju, Korea, November 5 - 9, 2007.

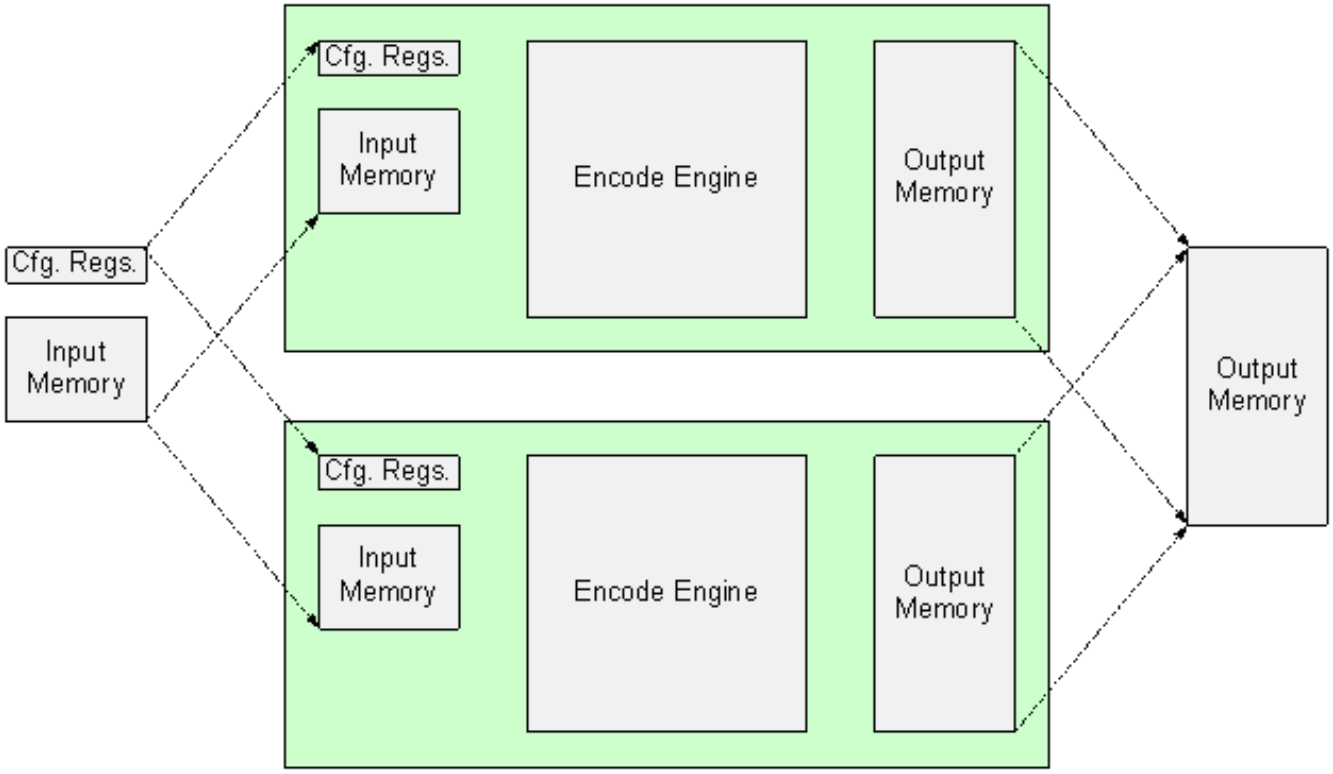
Chapter 2

Architecture

2.1 Functional Block Diagram

The functional partitioning of the TCP3E is shown in Figure 2-1. Though there are two complete encode engines inside, externally the TCP3E interface looks like a single set of input and output memories.

Figure 2-1 TCP3E Block Diagram



2.2 Description

The TCP3E contains two encode engines, each with its own set of memories both on the input and the output side. The ping/pong engine and memory selection is handled internally, so the user only sees a single input and output memory interface during normal operation. The internal logic alternates between ping and pong engines from one code block to the next.

Although it is treated as a single interface from the point-of-view of the memory map, the TCP3E VBUSP DMA interface is internally divided into separate write and read interfaces to improve throughput by allowing the transfer of data in and out simultaneously. The VBUSP config bus interface is used for accessing the control and status registers. The TCP3E issues WEVTs to the EDMA to indicate it is ready to receive data to be encoded. It issues REVTs to the EDMA to indicate that the encoded data can be read by the EDMA.

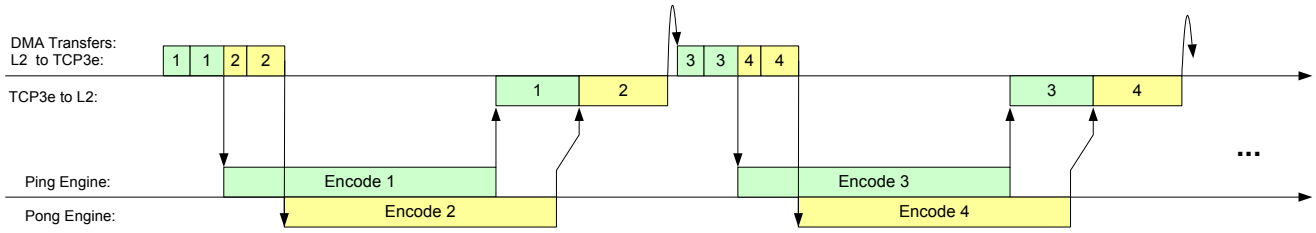
For LTE and 3GPP, 3 bits are output for every bit encoded. For WiMax, 6 bits are output for every 2 bits encoded together. The encoder achieves a throughput of ~250 Mbps in all three modes.

2.3 Throughput

The throughput of the TCP3E was calculated for the following two scenarios:

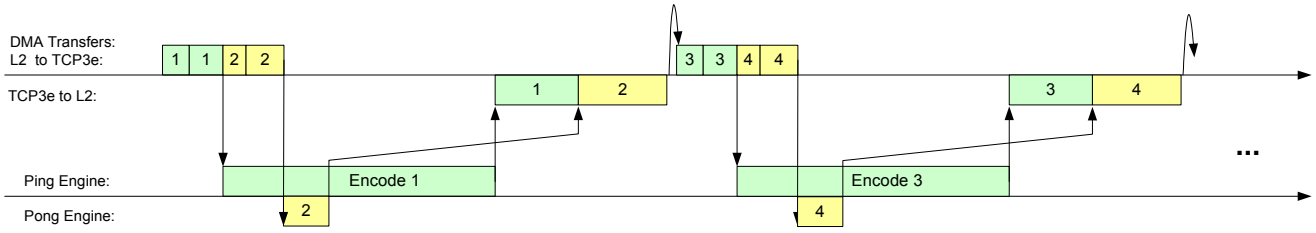
Best Case - All blocks are available, and are of the similar length (Figure 2-2).

Figure 2-2 Throughput - Best Case Scenario



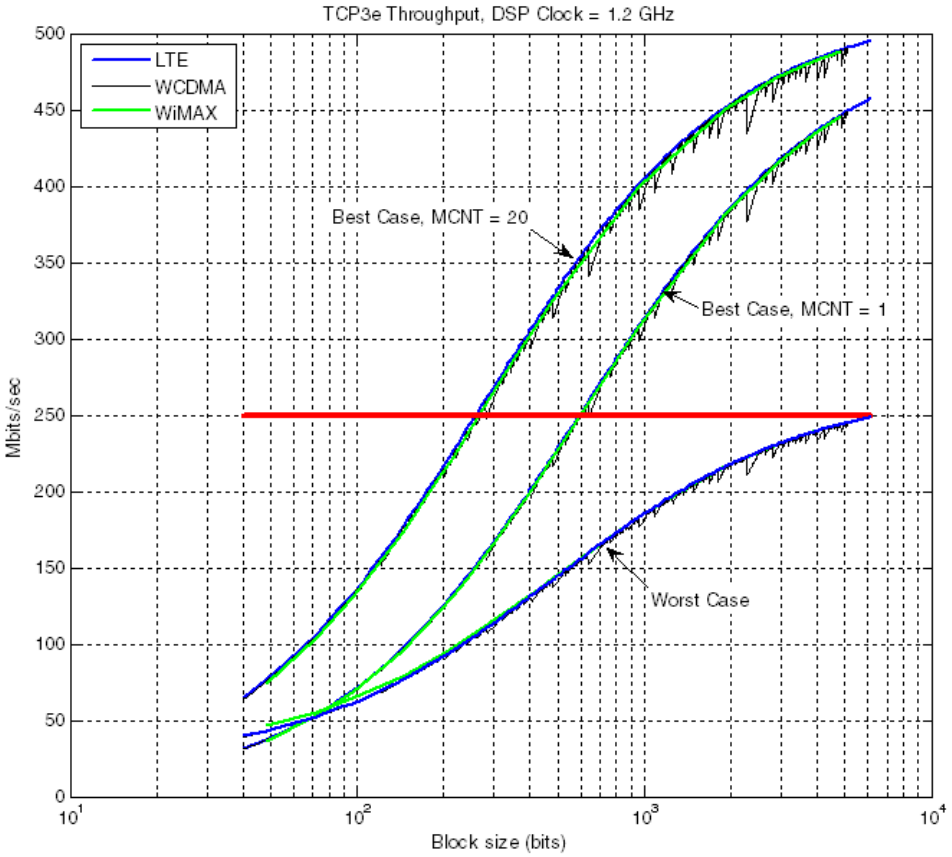
Worst Case - All blocks are available, consecutive blocks have large length difference (Figure 2-3).

Figure 2-3 Throughput - Worst Case Scenario



The throughput curve for each scenario and for 3GPP (WCDMA), LTE, and WiMax is shown in Figure 2-4.

Figure 2-4 Throughput Plot



Chapter 3

Usage

3.1 Usage Overview

This section describes how to set up and use the TCP3E from a user's point of view.

The overall usage and data flow for the TCP3E is described in the following steps.

1. DSP performs TCP3E initialization (mode, etc.) via CFG VBUS interface after TCP3E is reset (hard or soft reset).
2. DSP configures and initiates EDMA to transfer the input configuration registers for a set of MCNT code blocks via the 32-bit DMA VBUS interface. All of the input configuration registers must be written, even if they do not seem to be used in that mode or their value is not changing.
3. TCP3E issues a write event (WEVT) to EDMA to transfer the input data for the first code block to be encoded. TCP3E starts encoding the code block after it has finished loading.
4. If another encode engine is available (not currently in use) and there are more blocks to be encoded (based on MCNT), the TCP3E issues another write event (WEVT) to transfer the next code block's input data. The TCP3E starts encoding the code block after it has finished loading.
5. When the earlier code block is finished encoding, the TCP3E issues a receive event (REVT) to EDMA that causes it to transfer the encoded block out of the TCP3E. If there are more blocks to be encoded (based on MCNT), go back to step 4.
6. After the entire set of MCNT code blocks has been encoded and retrieved, the EDMA generates an interrupt to the DSP to signal that the encoding process is finished.
7. If there is another set of code blocks to be encoded, the EDMA loads the next set of input configuration registers and the processing flow goes back to step 3.

The following sections describe these steps in more detail.

3.2 Initialization

Initialization includes resetting the TCP3E (via hard or soft reset) and programming its control registers (**TCP3E_MODE**, **TCP3E_EMU**, and **TCP3E_ERR_MODE**) found in "Control and Status Registers (VBUS_CFG Bus)" on page 4-5. This step must be performed before any code blocks are encoded and is generally only repeated at system startup and when the mode is changed.

3.2.1 Reset

For initialization (including setting or changing the mode), the TCP3E must be reset via an external hard reset or by writing to the **TCP3E_SOFT_RESET** register. The contents of the embedded RAMs are not affected by either type of reset. For convenience, the soft reset has no effect on several of the registers as described in "TCP3E Soft Reset Register (TCP3E_SOFT_RESET)" on page 4-5. This means that the software does not have to reprogram those registers after a soft reset if their values do not need to be changed.

3.2.2 Mode Control

The **TCP3E_MODE** register controls different aspects of the operating mode of the TCP3E. Each field in the register is described in the following sections.

3.2.2.1 MODE Field

After resetting the TCP3E, the mode should be configured via the MODE field in the TCP3E_MODE register. "TCP3E Mode Control Register (TCP3E_MODE)" on page 4-5 describes the valid mode settings.

3.2.2.2 R_NQ_MUX Field

The R_NQ_MUX field is for future use. Currently always set it to '0'.

3.2.3 Emulation Mode Control

The **TCP3E_EMU** register controls the behavior of the TCP3E during an emulation halt. Each field in the register is described in the following sections. For more information on emulation mode usage see "Emulation Suspend Mode" on page 3-17.

Note—When halted in emulation mode, the TCP3E still responds to accesses on all of its VBUS interfaces.

3.2.3.1 FREERUN Field

The FREERUN field in the **TCP3E_EMU** register determines if an emulation halt of the DSP will also halt the TCP3E. If FREERUN is set to '1', and an emulation halt of the DSP occurs, the TCP3E will continue processing normally. If FREERUN is set to '0', and an emulation halt of the DSP occurs, then the TCP3E will be halted in a manner determined by the setting of the SOFT bit.

Note—When the FREERUN is set to '1', the SOFT field has no effect.

3.2.3.2 SOFT Field

The SOFT field in the **TCP3E_EMU** register determines how the TCP3E stops during an emulation halt. However, currently both settings of this field have the same effect.

If SOFT is set to ‘0’ or ‘1’, the TCP3E performs a ‘soft’ stop. The TCP3E is allowed to halt gracefully. It will finish processing all MCNT code blocks in the current code block set and all the results are transferred out via EDMA.

3.2.4 Error Mode Control

The **TCP3E_ERR_MODE** register is used to specify if TCP3E halts or continues running when a specific error condition is detected. See the register description in “TCP3E Error Mode Register (TCP3E_ERR_MODE)” on page 4-8 for more details.

3.3 EDMA Setup

This section describes how to set up and use the EDMA with the TCP3E.

3.3.1 EDMA Events

The TCP3E generates a write event (WEVT) to the EDMA controller when it is ready for the next code block's input data to be loaded into the TCP3E.

The TCP3E generates a receive (REVT) event to the EDMA controller when it finishes encoding a code block in order to transfer the encoded code block out of the TCP3E to system memory. If there are two code blocks being encoded, a receive event (REVT) will not be generated until the code block that was started first is finished encoding and has been transferred out of the TCP3E (even if the other code block finishes encoding earlier).

3.3.2 EDMA Programming

This section describes how to program the EDMA controller to work with the TCP3E.

3.3.2.1 Transfer Overview

The EDMA controller needs to transfer the input configuration register data as well as the input code block data from the system memory to the TCP3E. All of the input configuration registers must be loaded for each set of code blocks regardless of whether all of the values seem to be used for the current mode and regardless of whether the values are the same as the previous code block set. After the TCP3E encodes the data, the EDMA controller needs to transfer the encoded data from the TCP3E to the system memory.

Each of the following sections describes how to set up the EDMA for different scenarios.

3.3.2.2 Single Code Block PaRAM Entry Setup

This section describes how to set up the EDMA PaRAM entries to encode a single code block. The figures given (Figure 3-1 and Figure 3-2) display the PaRAM entry setup and transfer sequence and are followed by a description of the sequence of events.

Figure 3-1 Single Code Block PaRAM Entry Setup

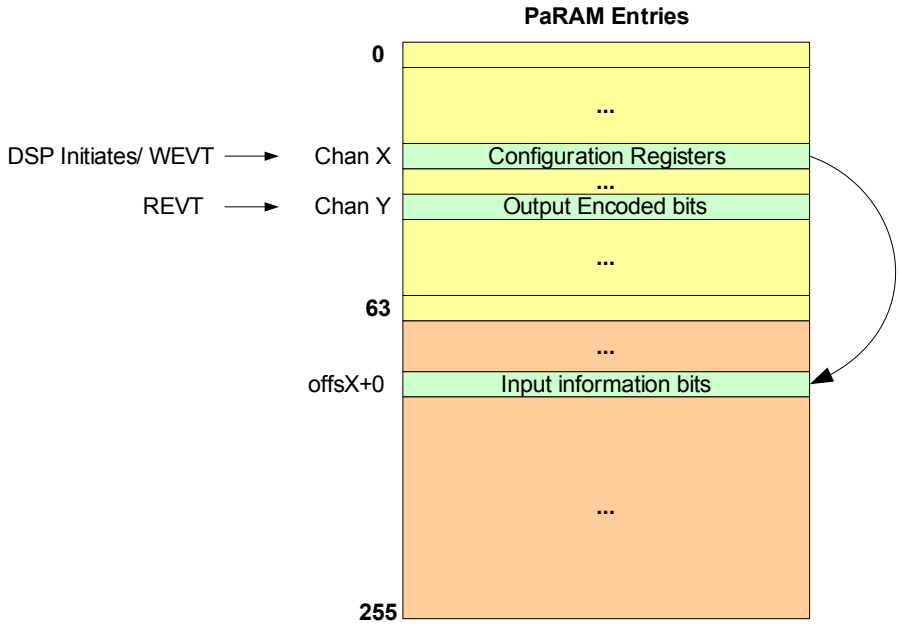
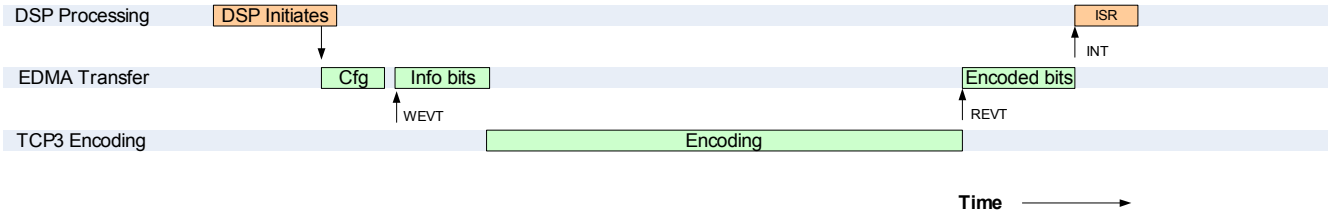


Figure 3-2 Single Code Block Transfer Sequence



The sequence of events is as follows:

1. The DSP initiates the process by setting the bit corresponding to EDMA channel X in the EDMA controller event set register.
2. EDMA transfers configuration registers and links in the entry with the input data (info bits).
3. TCP3E issues WEVT to initiate the input data transfer.
4. TCP3E encodes data and issues REVT on channel Y, to initiate encoded data transfer to DSP.
5. After the transfer completion, EDMA issues interrupt to DSP to signal the end of the encoding process.

3.3.2.3 Multiple Code Blocks of Equal Size PaRAM Entry Setup

This section describes how to set up the EDMA PaRAM entries to encode multiple code blocks of equal size. The figures given (Figure 3-3 and Figure 3-4) display the PaRAM entry setup and transfer sequence for an example with MCNT = 5.

Figure 3-3 Multiple Code Block of Equal Size PaRAM Entry Setup

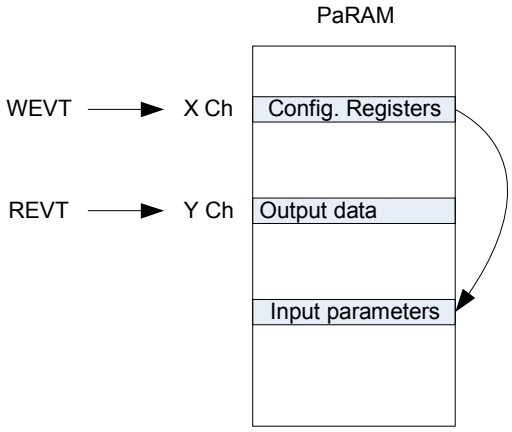
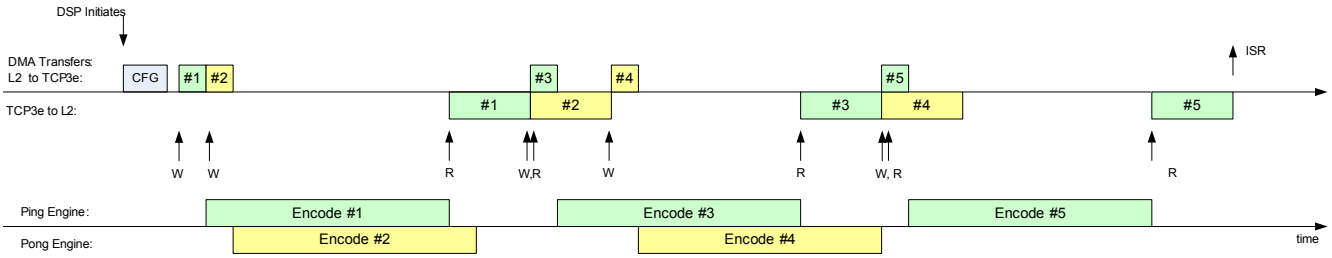
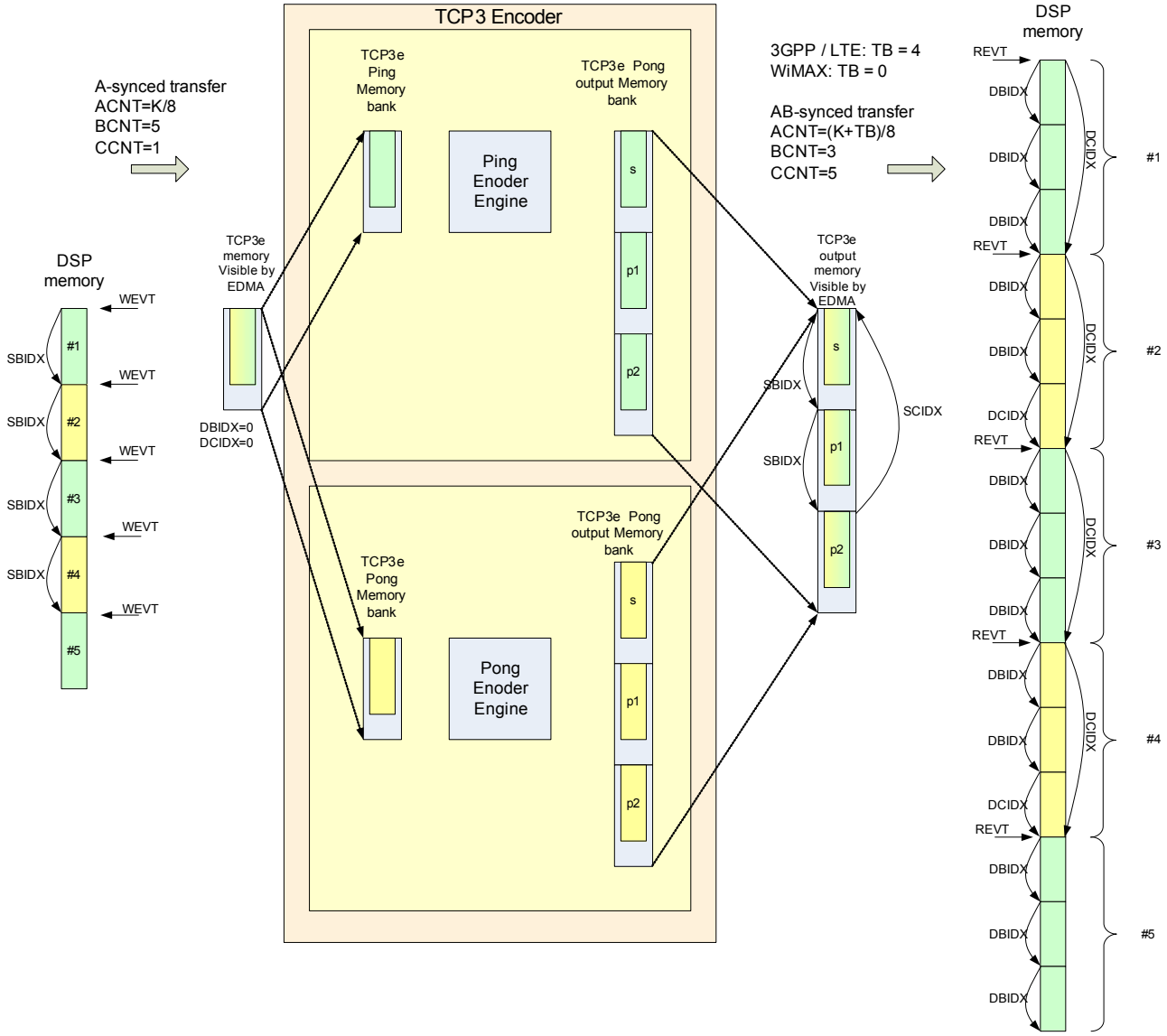


Figure 3-4 Multiple Code Block of Equal Size Transfer Sequence



The sequence of events is similar to a single code block except the PaRAM entries for input and output data are setup to move all five code blocks (one per WEVT/REVT). Figure 3-5 shows an example diagram.

Figure 3-5 Multiple Code Block of Equal Size Example Diagram



3.3.2.4 Multiple Code Blocks of Different Size PaRAM Entry Setup

This section describes how to set up the EDMA PaRAM entries to encode multiple code blocks of different size. The two figures given (Figure 3-6 and Figure 3-7) display the PaRAM entry setup and transfer sequence and are followed by a description of the operation.

Figure 3-6 Multiple Code Block of Different Size PaRAM Entry Setup

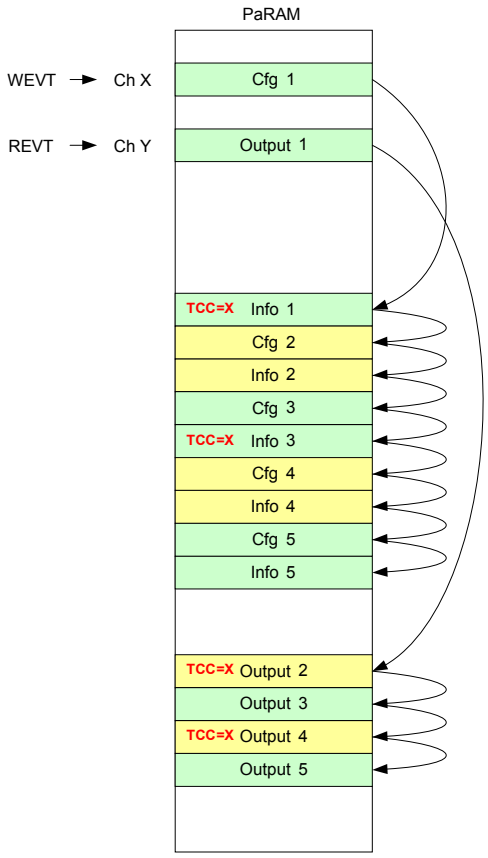
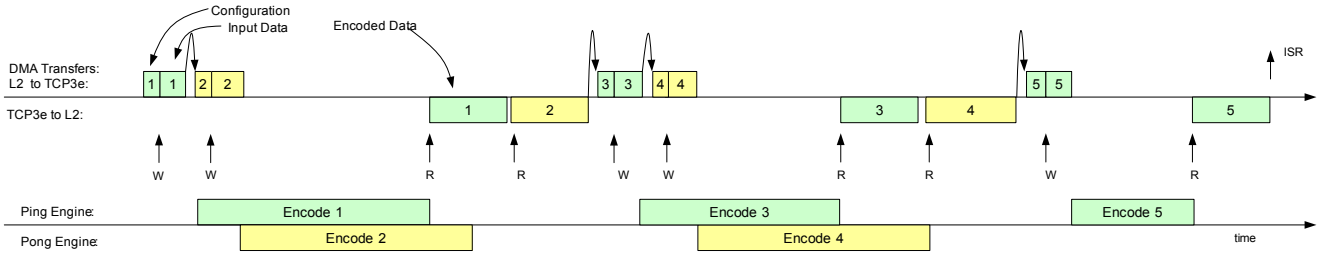


Figure 3-7 Multiple Code Block of Different Size Transfer Sequence

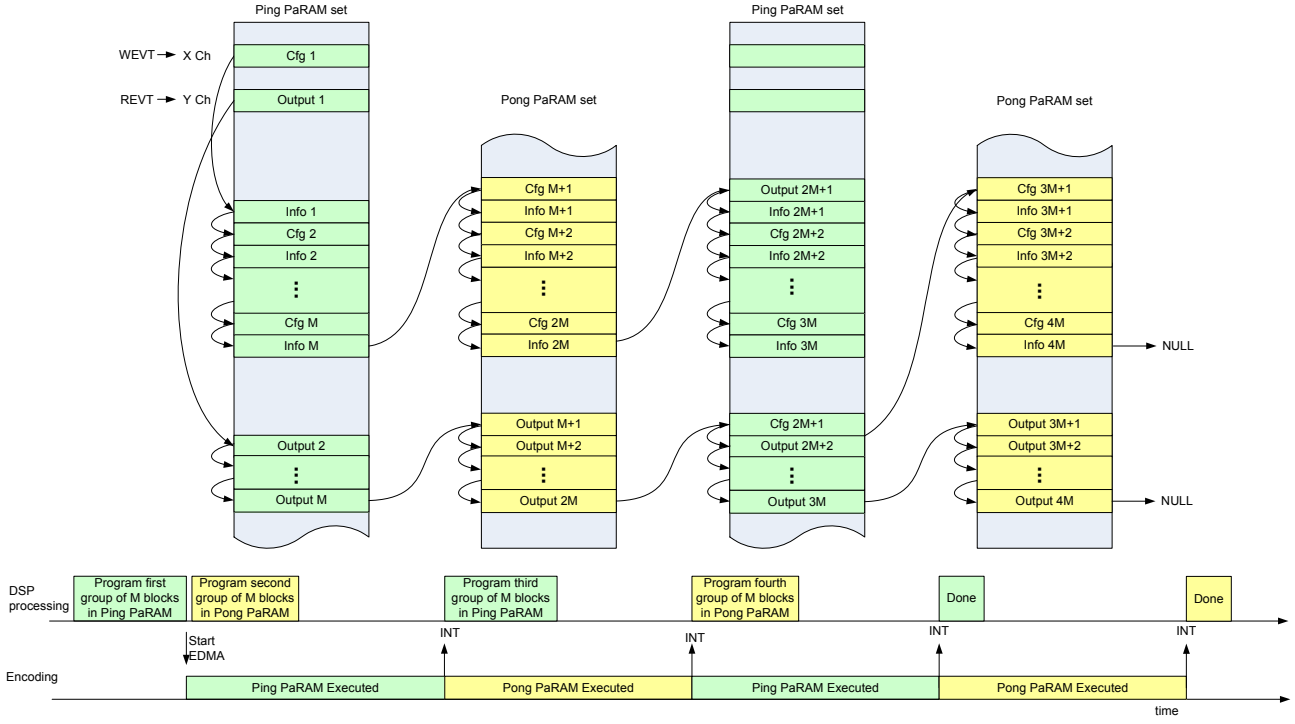


For M blocks, all having different size, 3M entries are required in the PaRAM memory. The entries 'Info 1', 'Output 2', 'Info 3' and 'Output 4', trigger the next DMA channel 'X' entry using the chaining approach. The TCP3E encodes two blocks at a time. Blocks #3 and #4 are started after both blocks, #1 and #2 are read out from the TCP3E. This scheduling methodology can be optimized beyond what is shown, but is not within the scope of this document.

3.3.2.5 Large Number of Code Blocks PaRAM Entry Setup

When the number of code blocks is too large to fit in PaRAM, the following approach can be used. The DSP programs M entries in one section of PaRAM space which will be referred to as the ping PaRAM set. While the TCP3E is processing those blocks, the DSP programs the next M entries into a second section of PaRAM space which will be referred to as the pong PaRAM set. (Note that these ping and pong sections have nothing to do with the ping and pong encode engines in the TCP3E.) The DSP then continues to alternate between the two spaces. Figure 3-8 shows an example where 4M blocks need to be processed.

Figure 3-8 Large Number of Code Blocks PaRAM Entry Setup



3.3.2.6 PaRAM Entry Details

This section provides the PaRAM entry details for each transfer.

3.3.2.6.1 Input Configuration Register Transfer

The PaRAM entry details for this transfer are as follows:

- OPT (Options):
  - ITCCHEN = 0 (Intermediate transfer complete chaining is disabled)
  - TCCHEN = 0 (Transfer complete chaining is disabled)
  - ITCINTEN = 0 (Intermediate transfer complete interrupt is disabled)
  - TCINTEN = 0 (Transfer complete interrupt is disabled)
  - TCC = 0 (Don't care since not used) (Transfer Complete Code)
  - TCCMODE = 0 (Normal Completion)
  - FWID = 0 (FIFO Width, don't care since src and dest are not FIFOs)
  - STATIC = 0 (Entry is updated as normal)
  - SYNCDIM = 0 (A-Sync. Each event triggers the transfer of ACNT elements.)
  - DAM = 0 (Dest addressing within an array increments. Dest is not a FIFO.)
  - SAM = 0 (Source addressing within an array increments. Source is not a FIFO.)
- SRC (address) = User input configuration parameters global start address
- ACNT = 12 (Number of bytes in an array) (All 3 input configuration registers must be loaded and each is 32-bits, or 4 bytes)
- BCNT = 1 (Number of arrays of length ACNT)
- DST (address) = TCP3E_CFG0 global address
- SRCBIDX = 0 (Source 2nd Dimension Index)
- DSTBIDX = 0 (Destination 2nd Dimension Index)
- BCNTRLD = 0 (BCNT reload, don't care since CCNT is 1)
- SRCCIDX = 0 (Source 3rd Dimension Index)
- DSTCIDX = 0 (Destination 3rd Dimension Index)
- CCNT = 1 (Number of frames in a block)
- LINK (address) = Address in the EDMA PaRAM of the entry associated with the corresponding code block input data transfer

3.3.2.6.2 Input Data Transfer

The PaRAM entry details for this transfer are as follows.

- OPT (Options):
  - ITCCHEN = 0 (Intermediate transfer complete chaining is disabled)
  - TCCHEN = varies, see "Multiple Code Blocks of Different Size PaRAM Entry Setup" on page 3-9 for example usage (Transfer complete chaining enable)
  - ITCINTEN = 0 (Intermediate transfer complete interrupt is disabled)
  - TCINTEN = 0 (Transfer complete interrupt is disabled)
  - TCC = varies, see "Multiple Code Blocks of Different Size PaRAM Entry Setup" on page 3-9 for example usage (Transfer Complete Code)
  - TCCMODE = 0 (Normal Completion)
  - FWID = 0 (FIFO Width, don't care since src and dest are not FIFOs)
  - STATIC = 0 (Entry is updated as normal)
  - SYNCDIM = 0 (A-Sync. Each event triggers the transfer of ACNT elements.)
  - DAM = 0 (Dest addressing within an array increments. Dest is not a FIFO.)
  - SAM = 0 (Source addressing within an array increments. Source is not a FIFO.)
- SRC (address) = User input data global start address (user input data must be 32-bit aligned)
- ACNT = ceil[(code block size - 24) / 8] for LTE when CRC generation is enabled, ceil[code block size / 8] in all other cases (Number of bytes in an array)
- BCNT = MCNT, number of code blocks in this code block set (Number of arrays of length ACNT)
- DST (address) = TCP3E Input Memory global address
- SRCBIDX = User specified offset between base addresses of input data code blocks (Source 2nd Dimension Index)
- DSTBIDX = 0 (always start at Input Memory base address) (Destination 2nd Dimension Index)
- BCNTRLD = 0 (BCNT reload, don't care since CCNT is 1)
- SRCCIDX = 0 (Source 3rd Dimension Index)
- DSTCIDX = 0 (Destination 3rd Dimension Index)
- CCNT = 1 (Number of frames in a block)
- LINK (address) = Address in the EDMA PaRAM of the entry associated with the next transfer (see examples in previous sections)

3.3.2.6.3 Output Encoded Data Transfer

The systematic, parity 0 and parity 1 data can be moved from the TCP3E to system memory as a single entry, AB-synchronized transfer. The PaRAM entry details for this transfer are as follows.

- OPT (Options):
  - ITCCHEN = 0 (Intermediate transfer complete chaining is disabled)
  - TCCHEN = varies, see"Multiple Code Blocks of Different Size PaRAM Entry Setup" on page 3-9 for example usage (Transfer complete chaining enable)
  - ITCINTEN = 0 (Intermediate transfer complete interrupt is disabled)
  - TCINTEN = User specified (Transfer complete interrupt is disabled)
  - TCC = varies, see "Multiple Code Blocks of Different Size PaRAM Entry Setup" on page 3-9 for example usage (Transfer Complete Code)
  - TCCMODE = 0 (Normal Completion)
  - FWID = 0 (FIFO Width, don't care since src and dest are not FIFOs)
  - STATIC = 0 (Entry is updated as normal)
  - SYNCDIM = 1 (AB-Sync. Each event triggers the transfer of BCNT arrays of ACNT bytes.)
  - DAM = 0 (Dest addressing within an array increments. Dest is not a FIFO.)
  - SAM = 0 (Source addressing within an array increments. Source is not a FIFO.)
- SRC (address) = TCP3E Output Memory 0 global start address
- ACNT = ceil[(code block size + numTailBitsPerSection) / 8] (numTailBitsPerSection is 4 for 3GPP and LTE, numTailBitsPerSection is 0 for WiMAX) (Number of bytes in an array)
- BCNT = 3 (Number of arrays of length ACNT)
- DST (address) = User encoded data memory global start address for this code block set
- SRCBIDX = 0x500 (Offset between sections of TCP3E Output Memory) (Source 2nd Dimension Index)
- DSTBIDX = User specified, must be at least ACNT (Destination 2nd Dimension Index)
- BCNTRLD = same value used for BCNT (BCNT reload)
- SRCCIDX = 0 (always start at Output Memory 0 base address) (Source 3rd Dimension Index)
- DSTCIDX = User specified, must be at least 3 * ACNT (Destination 3rd Dimension Index)
- CCNT = MCNT, number of code blocks in this code block set (Number of frames in a block)
- LINK (address) = Address in the EDMA PaRAM of the EDMA parameters associated with the next transfer (see examples in previous sections)

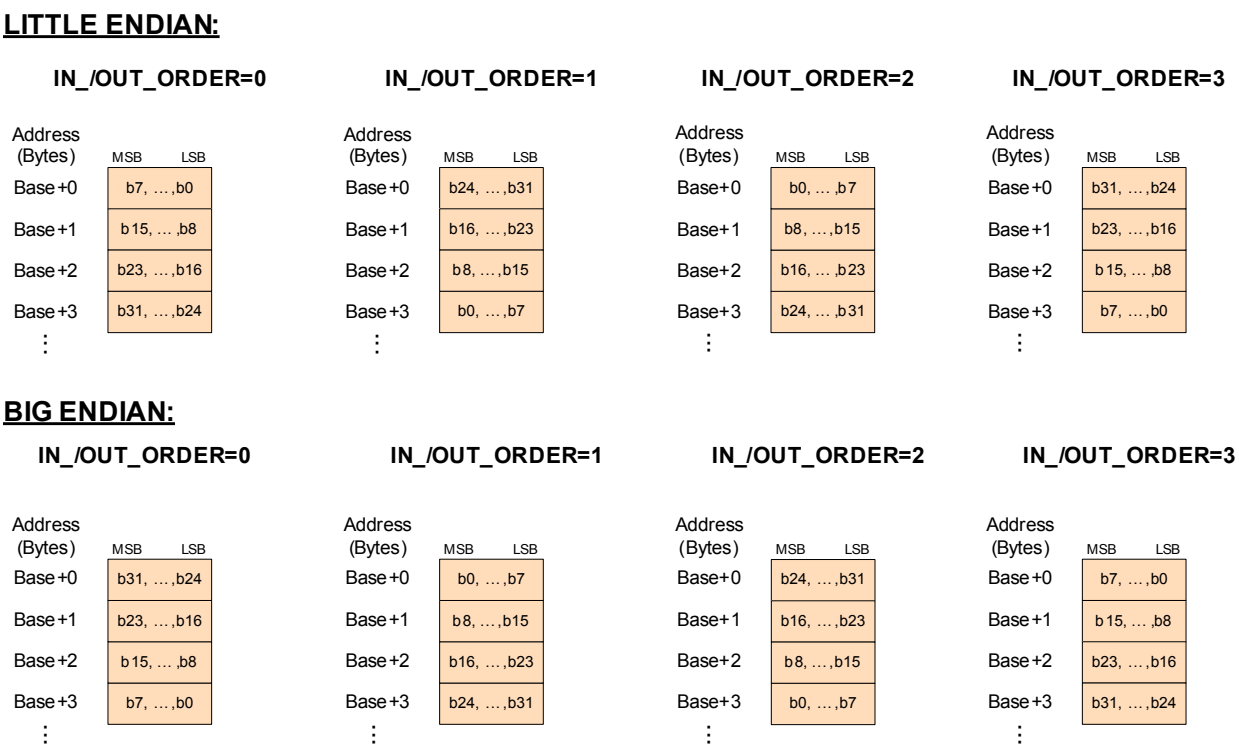
3.4 Input/Output Data Format

This section discusses the input and output data formats used in the TCP3E.

3.4.1 Endianness/Ordering Considerations

The input configuration parameters IN_ORDER and OUT_ORDER in the **TCP3E_CFG0** register are set depending on the DSP endianess and data packing order in the system memory. Their effect is shown in Figure 3-9 where b0 is the first data bit in time.

Figure 3-9 TCP3E Input/Output Data Ordering



3.4.2 Tail Bits for 3GPP/LTE

The TCP3E outputs three streams of data per code block: Systematic, Parity0, and Parity1:

Systematic Stream = $d_k^{(0)} = x_k$

Parity0 Stream = $d_k^{(1)} = z_k$

Parity1 Stream = $d_k^{(2)} = z'_k$

for k = 0,1,2,…,K-1

The tail bits are packed at the end of the streams as follows:

$$d_K^{(0)} = x_K \quad d_{K+1}^{(0)} = z_{K+1} \quad d_{K+2}^{(0)} = x'_K \quad d_{K+3}^{(0)} = z'_{K+1}$$

$$d_K^{(1)} = z_K \quad d_{K+1}^{(1)} = x_{K+2} \quad d_{K+2}^{(1)} = z'_K \quad d_{K+3}^{(1)} = x'_{K+2}$$

$$d_K^{(2)} = x_{K+1} \quad d_{K+1}^{(2)} = z_{K+2} \quad d_{K+2}^{(2)} = x'_{K+1} \quad d_{K+3}^{(2)} = z'_{K+2}$$

3.5 Error Detection and Interrupt

The TCP3E generates an error interrupt when it senses an error condition that has been enabled to produce an interrupt in the **TCP3E_EINT_EN** register. Error conditions include programming errors, such as setting a control parameter in a register to an invalid value, and incorrect DSP accesses to the TCP3E memories. The registers associated with the TCP3E error interrupt are described in "TCP3E Error Interrupt Registers (VBUS_CFG Bus)" on page 4-8.

When an error condition is detected, the TCP3E will either halt or continue running, based on the setting of the **TCP3E_ERR_MODE** register (see "TCP3E Error Mode Register (TCP3E_ERR_MODE)" on page 4-8). If the TCP3E halts on an error condition the DSP is expected to correct the error, reprogram the external DMA controller if possible, then restart the TCP3E. The TCP3E restart is accomplished by performing a TCP3E soft reset (by writing a '1' to **TCP3E_SOFT_RESET** register) and then reloading the input configuration registers with the correct parameters. If the TCP3E is set to continue running after an error condition and an error occurs, the encode results are not predictable.

Here are some additional rules that the TCP3E error interrupt follows:

- If an error condition occurs and the corresponding interrupt raw status bit is already set to a '1', the occurrence of the second error condition will not be saved.
- If the DSP attempts to clear an interrupt raw status bit at the same time it is being set again, the raw status bit will remain set to a '1'.
- In emulation suspend mode, the incorrect memory access error conditions are not generated.

Whether or not an error interrupt results in halting the TCP3E, the management of the error interrupt registers is as follows:

1. The DSP identifies the source of the interrupt by reading the TCP3E Error Interrupt Enabled Status register. Note that the time lapse between issuing the interrupt and the DSP reading this register could allow additional errors to be registered.
2. If the error(s) has not caused TCP3E to halt, the DSP clears the interrupt(s) by writing to the TCP3E Error Interrupt Clear register. If no new error conditions have occurred since the reading of the TCP3E Error Interrupt Enabled Status register, the TCP3E error interrupt output is negated. Otherwise, the new error conditions will cause the error interrupt output to remain asserted.
3. If the error(s) have caused the TCP3E to halt, a soft reset must be issued at this time to clear the error source(s). There is no need to write to the TCP3E Error Interrupt Clear register. The TCP3E error interrupt output is negated and will remain negated until the next error condition.
4. The DSP then writes an 8-bit vector to the TCP3E End of Interrupt register to acknowledge that it has serviced the TCP3E error interrupt condition(s).
5. If the TCP3E error interrupt output is still asserted, the Interrupt Distributor Component will generate another interrupt to the DSP.

3.6 Debug/Emulation Support

3.6.1 Emulation Suspend Mode

TCP3E supports emulation suspend mode to enable system level debug. In emulation suspend mode, the DSP is able to write to and read from every memory and register in the TCP3E. The registers in TCP3E are all 32-bit registers, so the same address mapping can be used for normal and emulation suspend mode.

Since emulation suspend mode accesses are 32-bit, and the TCP3E memories are also on 32-bit boundaries, there are no additional bitwidth issues for accessing the memories in emulation suspend mode. However, to allow full access to all memories, the emulation suspend mode RAM memory map is different from the normal mode RAM mapping. During normal mode, the ping and pong memories are accessed at the same address as the selection is done internally. For emulation suspend mode, since both of these memories have to be accessible, separate memory addressing has to be used. Check the memory map (section 4.2) for details. Also, all memory accesses in emulation suspend mode will be performed as if the IN_ORDER and OUT_ORDER fields in the TCP3E_CFG0 register were set to '0', meaning that no bit reversal or byte swapping will be performed.

In emulation suspend mode, it is assumed that the EDMA can still service outstanding REVTs and WEVTs. The TCP3E halts due to emulation suspend after all MCNT blocks in the current code block set are done encoding. After emulation suspend mode is over, when the TCP3E resumes operation, it starts again from the ping encode engine no matter where it ended during emulation suspend.

3.6.2 Debug Transaction Considerations (EMUDBG)

Certain features of software debuggers, like the “watch windows” found in Code Composer Studio, may result in accesses to TCP3E RAMs while the TCP3E is actively encoding a block. These accesses are not supported since the data read from the RAMs while the TCP3E is operating would not be correct and may corrupt the operation of the TCP3E.

The mechanism used to distinguish debugger transactions on the VBUSP is the emudbg signal. It is not supported by the TCP3E.

Chapter 4

Registers and Memories

4.1 Registers and Memories

The range of defined addresses for the VBUS_CFG and VBUS_DMA interfaces is shown in Table 4-1. Within each range are many unsupported address locations. Each interface is placed at a unique base address in the global address map.

Table 4-1 VBUS Interface Address Ranges

VBUS Interface	Defined Address Range
VBUS CFG	0x0C00 – 0x0C3C
VBUS DMA	0x0000 – 0x3E00

4.2 Memory Map

Table 4-2 shows the memory mapped control and status registers accessible via the config address space.

Table 4-2 Control and Status Registers (VBUS_CFG)

VBUS_CFG Offset Address	Register Name	Function
0x0C00	TCP3E_PID	TCP3E Peripheral Identification Register
0x0C04	TCP3E_MODE	TCP3E Mode Register
0x0C08	TCP3E_EMU	TCP3E Emulation Register
0x0C0C	TCP3E_SOFT_RESET	TCP3E Soft Reset
0x0C10	TCP3E_STAT	TCP3E Stat Register
0x0C1C	TCP3E_ERR_MODE	TCP3E Error Mask Register
0x0C20	TCP3E_EINT_STAT	TCP3E Error Interrupt Raw Status
0x0C24	TCP3E_EINT_SET_STAT	TCP3E Error Interrupt Set Status Register
0x0C28	TCP3E_EINT_CLR	TCP3E Error Interrupt Clear Register
0x0C2C	TCP3E_EINT_EN_STAT	TCP3E Error Interrupt Enabled Status Register
0x0C30	TCP3E_EINT_EN	TCP3E Error Interrupt Enable Register
0x0C34	TCP3E_EINT_EN_SET	TCP3E Error Interrupt Enable Set Register
0x0C38	TCP3E_EINT_EN_CLR	TCP3E Error Interrupt Enable Clear Register
0x0C3C	TCP3E_EOI	TCP3E End of Interrupt Register
End of Table 4-2		

Table 4-3 shows the memory mapped codeblock configuration registers accessible via the data address space.

Table 4-3 Codeblock Configuration Registers (VBUS_DMA)

Vbus_dma Offset Address	Register Name	Function
0x0C20	TCP3E_CONFIG_PARAM0	TCP3E Config Parameter Register 0
0x0C24	TCP3E_CONFIG_PARAM1	TCP3E Config Parameter Register 1
0x0C28	TCP3E_CONFIG_PARAM2	TCP3E Config Parameter Register 2
End of Table 4-3		

Table 4-4 shows the memory mapped RAM accessible via the data address space in normal operational mode.

Table 4-4 Memory Mapped RAM in Normal Mode (VBUS_DMA)

VBUS_DMA Offset Start Address	Name	Address Range	Length, bytes
0x1400	Input Memory	0x1400-0x17FC	1024 (0x400)
0x2000	Output Memory 0 (Sys)	0x2000-0x2400	1028 (0x404)
0x2500	Output Memory 1 (Par0)	0x2500-0x2900	1028 (0x404)
0x2A00	Output Memory 2 (Par1)	0x2A00-0x2E00	1028 (0x404)
End of Table 4-4			

Table 4-5 shows the memory mapped RAM accessible via the data address space in emulation suspend mode (for debug).

Table 4-5 Memory Mapped RAM in Emulation Suspend (VBUS_DMA)

VBUS_DMA Offset Start Address	Name	Address Range	Length (bytes)
0x1000	Ping Input Memory 1 (Non-interleaved)	0x1000-0x13FC	1024(0x400)
0x1400	Ping Input Memory 0 (Interleaved)	0x1400-0x17FC	1024 (0x400)
0x1800	Pong Input Memory 0 (Interleaved)	0x1800-0x1BFC	1024(0x400)
0x1C00	Pong Input Memory 1 (Non-interleaved)	0x1C00-0x1FFC	1024 (0x400)
0x2000	Ping Output Memory 0 (Sys)	0x2000-0x2400	1028 (0x404)
0x2500	Ping Output Memory 1 (Par0)	0x2500-0x2900	1028 (0x404)
0x2A00	Ping Output Memory 2 (Par1)	0x2A00-0x2E00	1028 (0x404)
0x3000	Pong Output Memory 0 (Sys)	0x3000-0x3400	1028 (0x404)
0x3500	Pong Output Memory 1 (Par0)	0x3500-0x3900	1028 (0x404)
0x3A00	Pong Output Memory 2 (Par1)	0x3A00-0x3E00	1028 (0x404)
End of Table 4-5			

4.3 Constant Registers (VBUS CFG Bus)

The only register with a constant, read-only value is the TCP3E_PID register.

4.3.1 TCP3E Peripheral ID Register (TCP3E_PID)

TCP3E_PID

Table 4-6 Peripheral ID Register

Bit	Name	R/W	Value	Description
5:0	MINOR	R	RLS	Minor revision (Y) code
7:6	CUSTOM	R	RLS	Custom version code
10:8	MAJOR	R	RLS	Major revision (X) code
15:11	RTL	R	RLS	RTL Version (R) code
27:16	FUNCTION	R	0x808	Function code assigned to TCP3E
29:28	Rsvd	R	00	Reserved
31:30	SCHEME	R	01	Current scheme
End of Table 4-6				

The peripheral identification register (TCP3E_PID) is a constant register that contains the ID and ID revision number for that peripheral. The TCP3E_PID stores version information used to identify the peripheral. All bits within this register are read-only (writes have no effect) meaning that the values within this register are hard-coded with the appropriate values and do not change from their reset state. The lower 16-bits of TCP3E_PID are the revision ID of the TCP3E. Since this is a dynamic value based upon the version number of the RTL, it is not kept up to date here. Instead, please refer to the release notes for the TCP3E releases.

4.4 Control and Status Registers (VBUS_CFG Bus)

The following sections describe the TCP3E control and status registers.

4.4.1 TCP3E Mode Control Register (TCP3E_MODE)

TCP3E_MODE

Table 4-7 TCP3E Mode Control Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
2:0	MODE	R/W		TCP3E mode:
			0	No mode selected
			1	3GPP mode
			2	LTE mode
			4	WiMax mode
				NOTE: • The TCP3E must be reset via external hard reset or by writing a '1' to the soft reset register before changing the mode.
3	Rsrvd	R	0	Reserved (always reads 0)
4	R_NQ_MUX	R/W	0	For future usage, currently always set to '0'.
31:5	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-7				

This register should be written to with the value for the desired mode after reset. Setting it to an invalid value or not setting it before running the encoder will result in an error. The TCP3E must be reset via external hard reset or by writing a '1' to the soft reset register before changing the mode.

4.4.2 TCP3E Emulation Control Register (TCP3E_EMU)

TCP3E_EMU

Table 4-8 Emulation Control Register (reset value 0x0003)

Bit	Name	R/W	Value	Description
0	FREERUN	R/W		FREERUN bit:
			0	TCP3E responds to the emulation suspend signal it is monitoring and operates according to the setting of the SOFT bit.
			1	TCP3E ignores emulation suspend signals and runs normally.
1	SOFT	R/W		SOFT bit:
			0	Hard stop: Same as soft stop.
			1	Soft stop: TCP3E completes all MCNT encodes in the current code block set and halts gracefully.
31:2	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-8				

4.4.3 TCP3E Soft Reset Register (TCP3E_SOFT_RESET)

This register is used to perform a TCP3E soft reset. All registers are reset during soft reset EXCEPT the following:

- TCP3E_MODE register

- TCP3E_EMU register
- TCP3E_ERR_MODE register
- TCP3E_EINT_EN register

TCP3E_SOFT_RESET

Table 4-9 TCP3E Soft Reset Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
31:0	SOFT_RESET	W		TCP3E soft reset control:
			1	Perform TCP3E soft reset
			Any other value	No effect
				NOTE: To soft reset TCP3E, just write '1' to this register. This register will always read as '0'.
End of Table 4-9				

4.4.4 TCP3E Status Register (TCP3E_STAT)

TCP3E_STAT

Table 4-10 TCP3E Status Register (reset value 0x0000) (Part 1 of 2)

Bit	Name	R/W	Value	Description
0	EMUL_HALT	R		Indicates that TCP3E is halted due to emulation suspend.
			0	TCP3E NOT halted
			1	TCP3E IS halted
1	BLK_SIZE_GT_SPEC	R		Indicates that block size is greater than 4800 but less than or equal to 8192 for Wimax OR greater than 6144 but less than or equal to 8192 for LTE.
			0	NOT in range
			1	IS in range
2	PONG_OUT_BUF_ACTIVE			Indicates that pong output buffer is active.
		R	0	Pong output buffer NOT active
			1	Pong output buffer IS active (TCP3E has issued an revt to read pong encoded data, but the read process has not completed)
3	PONG_ENC_ACTIVE			Indicates that pong encode is active.
		R	0	Pong encode NOT active
			1	Pong encode IS active (pong input buffer has been filled, but the encode engine has not yet completed encoding the data and storing it in the pong output buffer)
4	PONG_IN_BUF_ACTIVE			Indicates that pong input buffer is active.
		R	0	Pong input buffer NOT active
			1	Pong input buffer IS active (TCP3E has issued an wevt to input pong data, but the write process to the pong input buffer has not completed)
5	PING_OUT_BUF_ACTIVE			Indicates that ping output buffer is active.
		R	0	Ping output buffer NOT active
			1	Ping output buffer IS active (TCP3E has issued an revt to read ping encoded data, but the read process has not completed)

Table 4-10 TCP3E Status Register (reset value 0x0000) (Part 2 of 2)

Bit	Name	R/W	Value	Description
6	PING_ENC_ACTIVE			Indicates that ping encode is active.
		R	0	Ping encode NOT active
			1	Ping encode IS active (ping input buffer has been filled, but the encode engine has not yet completed encoding the data and storing it in the ping output buffer)
7	PING_IN_BUF_ACTIVE			Indicates that ping input buffer is active.
		R	0	Ping input buffer NOT active
			1	Ping input buffer IS active (TCP3E has issued an wevt to input ping data, but the write process to the ping input buffer has not completed)
8	ERROR_HALT	R		Indicates that TCP3E is halted due to an error condition.
			0	NOT halted due to an error condition
			1	IS halted due to an error condition
31:9	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-10				

4.5 TCP3E Error Interrupt Registers (VBUS_CFG Bus)

The following sections describe the TCP3E error interrupt registers. Table 4-11 describes each error condition that may produce an error interrupt.

Table 4-11 TCP3E Error Condition Descriptions

Bit Index in Registers	Error Condition	Description
0	ILLEGAL_MODE	Mode is set to an illegal value (not 1,2 or 4). This error condition may occur when writing to the TCP3E_MODE register or when the TCP3E starts a new encode operation.
1	MCNT_ZERO	MCNT is zero. This error condition may occur when writing to the TCP3E_CFG0 register.
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL	Block size too large or too small. This error condition may occur when writing to the TCP3E_CFG0 register.
3	MODE_WRITE_DURING_ENCODE	Mode set during encode operation. This error condition may occur when writing to the TCP3E_MODE register.
4	INPUT_MEM_READ	Input buffer read in non-emulation mode. This error condition may occur when reading from the TCP3E input memory.
5	INPUT_BUFFER_WRITE_ACCESS	Write to input buffer beyond current access range. This error condition may occur when writing to the TCP3E input memory.
6	OUTPUT_BUFFER_WRITE	Write to output buffer in non-emulation mode. This error condition may occur when writing to the TCP3E output memory.
7	OUTPUT_BUFFER_ACCESS_RANGE	Read from output buffer beyond current access range. This error condition may occur when reading from the TCP3E output memory.
End of Table 4-11		

4.5.1 TCP3E Error Mode Register (TCP3E_ERR_MODE)

This register specifies if TCP3E halts or continues running when a specific error condition is detected. There is one bit for each error condition. Even though the TCP3E can be programmed to continue operating after an error has occurred, the encoded results are not guaranteed to be valid in that case. Bits 0 to 3 are reserved since those four error conditions will always halt the TCP3E.

TCP3E_ERR_MODE

Table 4-12 TCP3E Error Mode Register (reset value 0x0000) (Part 1 of 2)

Bit	Name	R/W	Value	Description
3:0	Rsrvd	R	0	Reserved (always reads 0) These error conditions are not maskable: ILLEGAL_MODE MCNT_ZERO BLKSIZE_TOO_LARGE_OR_TOO_SMALL MODE_WRITE_DURING_ENCODE
4	ERR_HALT_EN4	R/W		INPUT_MEM_READ error condition: 0 TCP3E will continue running after error condition is detected 1 TCP3E will HALT after error condition is detected
5	ERR_HALT_EN5	R/W		INPUT_BUFFER_WRITE_ACCESS error condition: 0 TCP3E will continue running after error condition is detected 1 TCP3E will HALT after error condition is detected

Table 4-12 TCP3E Error Mode Register (reset value 0x0000) (Part 2 of 2)

Bit	Name	R/W	Value	Description
6	ERR_HALT_EN6	R/W	0 1	OUTPUT_BUFFER_WRITE error condition: TCP3E will continue running after error condition is detected TCP3E will HALT after error condition is detected
7	ERR_HALT_EN7	R/W	0 1	OUTPUT_BUFFER_ACCESS_RANGE error condition: TCP3E will continue running after error condition is detected TCP3E will HALT after error condition is detected
31:8	Rsrvd	R	N/A	Reserved (always reads 0)
End of Table 4-12				

4.5.2 TCP3E Error Interrupt Raw Status Register (TCP3E_EINT_STAT)

The **TCP3E Error Interrupt Raw Status** register contains the interrupt status bit for each error condition before any enable processing. It is described in Table 4-13. The interrupt status bits are set when an error condition occurs or when a bit is set in the **TCP3E Error Interrupt Set** register (for debug to allow software to set an interrupt status bit). They are cleared when a '1' is written to the associated bit in the **TCP3E Error Interrupt Clear** register.

TCP3E_EINT_STAT

Table 4-13 TCP3E Error Interrupt Raw Status Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_ERR	R	0 1	ILLEGAL_MODE error condition: Error condition NOT detected Error condition IS detected
1	MCNT_ZERO_ERR	R	0 1	MCNT_ZERO error condition: Error condition NOT detected Error condition IS detected
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_ERR	R	0 1	BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition: Error condition NOT detected Error condition IS detected
3	MODE_WRITE_DURING_ENCODE_ERR	R	0 1	MODE_WRITE_DURING_ENCODE error condition: Error condition NOT detected Error condition IS detected
4	INPUT_MEM_READ_ERR	R	0 1	INPUT_MEM_READ error condition: Error condition NOT detected Error condition IS detected
5	INPUT_BUFFER_WRITE_ACCESS_ERR	R	0 1	INPUT_BUFFER_WRITE_ACCESS error condition: Error condition NOT detected Error condition IS detected
6	OUTPUT_BUFFER_WRITE_ERR	R	0 1	OUTPUT_BUFFER_WRITE error condition: Error condition NOT detected Error condition IS detected
7	OUTPUT_BUFFER_ACCESS_RANGE_ERR	R	0 1	OUTPUT_BUFFER_ACCESS_RANGE error condition: Error condition NOT detected Error condition IS detected
31:8	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-13				

4.5.3 TCP3E Error Interrupt Set Register (TCP3E_EINT_SET_STAT)

The **TCP3E Error Interrupt Set** register is used for debug to allow software to set an interrupt status bit. Writing a '1' here causes the associated interrupt status bit in the **TCP3E Error Interrupt Raw Status** register to be set to a '1'. Writing a '0' has no effect.

TCP3E_EINT_STAT

Table 4-14 TCP3E Error Interrupt Raw Status Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_SET	W		ILLEGAL_MODE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
1	MCNT_ZERO_SET	W		MCNT_ZERO error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_SET	W		BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
3	MODE_WRITE_DURING_ENCODE_SET	W		MODE_WRITE_DURING_ENCODE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
4	INPUT_MEM_READ_SET	W		INPUT_MEM_READ error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
5	INPUT_BUFFER_WRITE_ACCESS_SET	W		INPUT_BUFFER_WRITE_ACCESS error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
6	OUTPUT_BUFFER_WRITE_SET	W		OUTPUT_BUFFER_WRITE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
7	OUTPUT_BUFFER_ACCESS_RANGE_SET	W		OUTPUT_BUFFER_ACCESS_RANGE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Raw Status register
31:8	Rsrvd	W	N/A	Reserved
End of Table 4-14				

4.5.4 TCP3E Error Interrupt Clear Register (TCP3E_EINT_CLR)

The **TCP3E Error Interrupt Clear** register is a write-only register that mirrors the interrupt status bits in the **TCP3E Error Interrupt Raw Status** register. It is described in Table 4-15. Writing a '1' here causes the associated interrupt status bit in the **TCP3E Error Interrupt Raw Status** register to be cleared to a '0'. This register is used by software to clear an interrupt condition after it has been serviced. Writing a '0' has no effect. Reading this register will return the value in the **TCP3E Error Interrupt Raw Status** register.

TCP3E_EINT_CLR

Table 4-15 TCP3E Error Interrupt Clear Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_CLR	W		ILLEGAL_MODE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
1	MCNT_ZERO_CLR	W		MCNT_ZERO error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_CLR	W		BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
3	MODE_WRITE_DURING_ENCODE_CLR	W		MODE_WRITE_DURING_ENCODE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
4	INPUT_MEM_READ_CLR	W		INPUT_MEM_READ error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
5	INPUT_BUFFER_WRITE_ACCESS_CLR	W		INPUT_BUFFER_WRITE_ACCESS error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
6	OUTPUT_BUFFER_WRITE_CLR	W		OUTPUT_BUFFER_WRITE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
7	OUTPUT_BUFFER_ACCESS_RANGE_CLR	W		OUTPUT_BUFFER_ACCESS_RANGE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Raw Status register
31:8	Rsrvd	W	N/A	Reserved
End of Table 4-15				

4.5.5 TCP3E Error Interrupt Enabled Status Register (TCP3E_EINT_EN_STAT)

The **TCP3E Error Interrupt Enabled Status** register is a read-only register that is the logical AND of the **TCP3E Error Interrupt Raw Status** register and the **TCP3E Error Interrupt Enable** register. It is described in Table 4-16.

TCP3E_EINT_EN_STAT

Table 4-16 TCP3E Error Interrupt Enabled Status Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_STAT	R		ILLEGAL_MODE error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
1	MCNT_ZERO_STAT	R		MCNT_ZERO error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_STAT	R		BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
3	MODE_WRITE_DURING_ENCODE_STAT	R		MODE_WRITE_DURING_ENCODE error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
4	INPUT_MEM_READ_STAT	R		INPUT_MEM_READ error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
5	INPUT_BUFFER_WRITE_ACCESS_STAT	R		INPUT_BUFFER_WRITE_ACCESS error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
6	OUTPUT_BUFFER_WRITE_STAT	R		OUTPUT_BUFFER_WRITE error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
7	OUTPUT_BUFFER_ACCESS_RANGE_STAT	R		OUTPUT_BUFFER_ACCESS_RANGE error condition:
			0	Error condition NOT detected
			1	Error condition IS detected
31:8	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-16				

4.5.6 TCP3E Error Interrupt Enable Register (TCP3E_EINT_EN)

The **TCP3E Error Interrupt Enable** register contains the enables for each of the interrupt status bits in the **TCP3E Error Interrupt Enabled Status** register. It is described in Table 4-17. An interrupt status bit can only be set in the **TCP3E Error Interrupt Enabled Status** register and cause an interrupt if it is enabled.

Interrupt enable bits are set by writing to the **TCP3E Interrupt Error Interrupt Enable Set** register. Interrupt enable bits are cleared by writing to the **TCP3E Interrupt Error Interrupt Enable Clear** register.

TCP3E_EINT_EN

Table 4-17 TCP3E Error Interrupt Enable Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_EN	R		ILLEGAL_MODE error condition:
			0	Disabled
			1	Enabled
1	MCNT_ZERO_EN	R		MCNT_ZERO error condition:
			0	Disabled
			1	Enabled
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_EN	R		BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition:
			0	Disabled
			1	Enabled
3	MODE_WRITE_DURING_ENCODE_EN	R		MODE_WRITE_DURING_ENCODE error condition:
			0	Disabled
			1	Enabled
4	INPUT_MEM_READ_EN	R		INPUT_MEM_READ error condition:
			0	Disabled
			1	Enabled
5	INPUT_BUFFER_WRITE_ACCESS_EN	R		INPUT_BUFFER_WRITE_ACCESS error condition:
			0	Disabled
			1	Enabled
6	OUTPUT_BUFFER_WRITE_EN	R		OUTPUT_BUFFER_WRITE error condition:
			0	Disabled
			1	Enabled
7	OUTPUT_BUFFER_ACCESS_RANGE_EN	R		OUTPUT_BUFFER_ACCESS_RANGE error condition:
			0	Disabled
			1	Enabled
31:8	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-17				

4.5.7 TCP3E Error Interrupt Enable Set Register (TCP3E_EINT_EN_SET)

The **TCP3E Error Interrupt Enable Set** register is used to set the interrupt enable bits in the **TCP3E Error Interrupt Enable** register. It is described in Table 4-18. Writing a '1' here causes the associated interrupt enable bit in the **TCP3E Error Interrupt Enable** register to be set to a '1'. Writing a '0' has no effect.

TCP3E_EINT_EN_SET

Table 4-18 TCP3E Error Interrupt Enable Set Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_EN_SET	W		ILLEGAL_MODE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
1	MCNT_ZERO_EN_SET	W		MCNT_ZERO error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_EN_SET	W		BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
3	MODE_WRITE_DURING_ENCODE_EN_SET	W		MODE_WRITE_DURING_ENCODE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
4	INPUT_MEM_READ_EN_SET	W		INPUT_MEM_READ error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
5	INPUT_BUFFER_WRITE_ACCESS_EN_SET	W		INPUT_BUFFER_WRITE_ACCESS error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
6	OUTPUT_BUFFER_WRITE_EN_SET	W		OUTPUT_BUFFER_WRITE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
7	OUTPUT_BUFFER_ACCESS_RANGE_EN_SET	W		OUTPUT_BUFFER_ACCESS_RANGE error condition:
			0	No effect
			1	Corresponding bit is set in TCP3E Error Interrupt Enable register
31:8	Rsrvd	W	N/A	Reserved
End of Table 4-18				

4.5.8 TCP3E Error Interrupt Enable Clear Register (TCP3E_EINT_EN_CLR)

The **TCP3E Error Interrupt Enable Clear** register is a write-only register that mirrors the interrupt enable bits in the **TCP3E Error Interrupt Enable** register. Writing '1' causes the associated interrupt enable in the **TCP3E Error Interrupt Enable** register to be cleared to a '0' and disables the associated interrupt status bit. Writing a '0' has no effect. Reading this register will return the value in the **TCP3E Error Interrupt Enable** register. It is described in Table 4-19.

TCP3E_EINT_EN_CLR

Table 4-19 TCP3E Error Interrupt Enable Clear Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
0	ILLEGAL_MODE_EN_SET	W		ILLEGAL_MODE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
1	MCNT_ZERO_EN_CLR	W		MCNT_ZERO error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
2	BLKSIZE_TOO_LARGE_OR_TOO_SMALL_EN_CLR	W		BLKSIZE_TOO_LARGE_OR_TOO_SMALL error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
3	MODE_WRITE_DURING_ENCODE_EN_CLR	W		MODE_WRITE_DURING_ENCODE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
4	INPUT_MEM_READ_EN_CLR	W		INPUT_MEM_READ error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
5	INPUT_BUFFER_WRITE_ACCESS_EN_CLR	W		INPUT_BUFFER_WRITE_ACCESS error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
6	OUTPUT_BUFFER_WRITE_EN_CLR	W		OUTPUT_BUFFER_WRITE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
7	OUTPUT_BUFFER_ACCESS_RANGE_EN_CLR	W		OUTPUT_BUFFER_ACCESS_RANGE error condition:
			0	No effect
			1	Corresponding bit is cleared in TCP3E Error Interrupt Enable register
31:8	Rsrvd	W	N/A	Reserved
End of Table 4-19				

4.6 Common Interrupt Registers (VBUS CFG Bus)

The only common interrupt register in the TCP3E is the TCP3E_EOI register described here.

4.6.1 TCP3E End of Interrupt Register (TCP3E_EOI)

The **TCP3E End of Interrupt** register supports the End of Interrupt (EOI) interface between the TCP3E and the external Interrupt Distributor Component. It is a common register that is used for servicing the TCP3E interrupts. This register is written by software to acknowledge the interrupt has been cleared so another interrupt of the same type can be generated by the Interrupt Distributor Component. This register is written after a TCP3E interrupt has been cleared via the associated Interrupt Clear register. It is described in Table 4-20. Writes to this register have no effect on the internal interrupt processing of the TCP3E. The specific value to be written into this register is found in the chip-level documentation.

TCP3E_EOI

Table 4-20 TCP3E End of Interrupt Register (reset value 0x0000)

Bit	Name	R/W	Value	Description
7:0	EOI_VECTOR	R/W	0x00 - 0xFF	End of interrupt vector value that distinguishes which interrupt is being acknowledged. This value is output on the external eoi_vector interface of the TCP3E. TCP3E End of Interrupt Vector to acknowledge clear of Raw Status bit register
31:8	Rsrvd	R	0	Reserved (always reads 0)
End of Table 4-20				

4.7 Input Configuration Registers (VBUS DMA Bus)

The following sections describe the TCP3E input configuration registers. These 32-bit registers are accessible via the 32-bit VBUSP_DMA interface.

4.7.1 TCP3E Input Configuration Register 0 (TCP3E_CFG0)

The TCP3E_CFG0 input configuration register is described in Table 4-21.

TCP3E_CFG0

Table 4-21 TCP3E Input Config Register 0 (reset value 0x0000)

Bit	Name	R/W	Value	Description
11:0	MCNT	R/W	1 - 4095	Number of code-blocks that need to be processed (setting value to 0 is illegal)
12	Rsrvd	R	0	Reserved (always reads 0)
26:13	BLOCK_SIZE	R/W	40 - 5114 40 - 6144 48 - 4800	Un-encoded code block size (in bits) (includes the 24-bit CRC for LTE when CRC generation is enabled) WCDMA mode LTE mode WiMAX mode
28:27	IN_ORDER	R/W		Controls format conversion of input data: 0 - No bit reversal on the input 32-bit word 1 - Bit reverse within the 32-bit input word 2 - Bit reverse within each byte of the 32-bit input word 3 - Byte swap, no bit reverse on 32-bit input word
30:29	OUT_ORDER	R/W		Controls format conversion of output data: 0 - No bit reversal on the output 32-bit word 1 - Bit reverse within the 32-bit output word 2 - Bit reverse within each byte of the 32-bit output word 3 - Byte swap, no bit reverse on 32-bit output word
31	GEN_LTE_CRC	R/W		Only used for LTE mode to determine if CRC will be generated or not. This flag is ignored if not in LTE mode. Note that when this flag is enabled in LTE mode, the amount of input data transferred via EDMA will be 24 bits less than the specified BLOCK_SIZE. 0 - CRC NOT generated 1 - CRC IS generated
End of Table 4-21				

4.7.2 TCP3E Input Configuration Register 1 (TCP3E_CFG1)

The TCP3E_CFG1 input configuration register is described in Table 4-22. This register provides two of the constants required for LTE and WiMax interleaver table generation. The field definitions are different for each mode. This register must be set to 0 in 3GPP mode for proper operation of the encoder.

TCP3E_CFG1

Table 4-22 TCP3E Input Config Register 1 (reset value 0x0000)

Bit	Name	R/W	Value	Description
15:0	INTLV_PAR0	R/W		3GPP mode : must be set to 0 LTE mode : f1 WiMAX mode: P0
31:16	INTLV_PAR1	R/W		3GPP mode : must be set to 0 LTE mode : f2 WiMAX mode: P1
End of Table 4-22				

4.7.3 TCP3E Input Configuration Register 2 (TCP3E_CFG2)

The TCP3E_CFG2 input configuration register is described in Table 4-23. This register provides the remaining two constants required by the WiMax interleaver table generation. This register must be set to 0 in 3GPP and LTE modes for proper operation of the encoder.

TCP3E_CFG2

Table 4-23 TCP3E Input Config Register 2 (reset value 0x0000)

Bit	Name	R/W	Value	Description
15:0	INTLV_PAR2	R/W		3GPP mode : must be set to 0 LTE mode : must be set to 0 WiMAX mode: P2
31:16	INTLV_PAR3	R/W	-	3GPP mode : must be set to 0 LTE mode : must be set to 0 WiMAX mode: P3
End of Table 4-23				

Index

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	Communications and Telecom	www.ti.com/communications
DSP	dsp.ti.com	Computers and Peripherals	www.ti.com/computers
Clocks and Timers	www.ti.com/clocks	Consumer Electronics	www.ti.com/consumer-apps
Interface	interface.ti.com	Energy	www.ti.com/energy
Logic	logic.ti.com	Industrial	www.ti.com/industrial
Power Mgmt	power.ti.com	Medical	www.ti.com/medical
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Space, Avionics & Defense	www.ti.com/space-avionics-defense
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video and Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless-apps

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