TMS320C64x DSP Turbo-Decoder Coprocessor (TCP) Reference Guide

Literature Number: SPRU534B
September 2004

Texas Instruments

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 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.

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.

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
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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

Preface

Read This First

About This Manual

Channel decoding of high bit-rate data channels found in third generation (3G) cellular standards requires decoding of turbo-encoded data. The turbo-decoder coprocessor (TCP) in some of the digital signal processor (DSPs) of the TMS320C6000™ DSP family has been designed to perform this operation for IS2000 and 3GPP wireless standards. This document describes the operation and programming of the TCP.

Notational 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.

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.

TMS320C6000 CPU and Instruction Set Reference Guide (literature number SPRU189) describes the TMS320C6000™ CPU architecture, instruction set, pipeline, and interrupts for these digital signal processors.

TMS320C6000 DSP Peripherals Overview Reference Guide (literature number SPRU190) describes the peripherals available on the TMS320C6000™ DSPs.

TMS320C64x Technical Overview (SPRU395) gives an introduction to the TMS320C64x™ DSP and discusses the application areas that are enhanced by the TMS320C64x VelociTI™.

TMS320C6000 Programmer's Guide (literature number SPRU198) describes ways to optimize C and assembly code for the TMS320C6000™ DSPs and includes application program examples.

TMS320C6000 Code Composer Studio Tutorial (literature number SPRU301) introduces the Code Composer Studio™ integrated development environment and software tools.

Code Composer Studio Application Programming Interface Reference Guide (literature number SPRU321) describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.

TMS320C6x Peripheral Support Library Programmer's Reference (literature number SPRU273) describes the contents of the TMS320C6000™ peripheral support library of functions and macros. It lists functions and macros both by header file and alphabetically, provides a complete description of each, and gives code examples to show how they are used.

TMS320C6000 Chip Support Library API Reference Guide (literature number SPRU401) describes a set of application programming interfaces (APIs) used to configure and control the on-chip peripherals.

Trademarks

Code Composer Studio, C6000, C62x, C64x, C67x, TMS320C6000, TMS320C62x, TMS320C64x, TMS320C67x, and VelociTI are trademarks of Texas Instruments.

Contents

Figures

Tables

Turbo-Decoder Coprocessor (TCP)

Channel decoding of high bit-rate data channels found in third generation (3G) cellular standards requires decoding of turbo-encoded data. The turbo-decoder coprocessor (TCP) in some of the digital signal processor (DSPs) of the TMS320C6000™ DSP family has been designed to perform this operation for IS2000 and 3GPP wireless standards. This document describes the operation and programming of the TCP.

1 Features

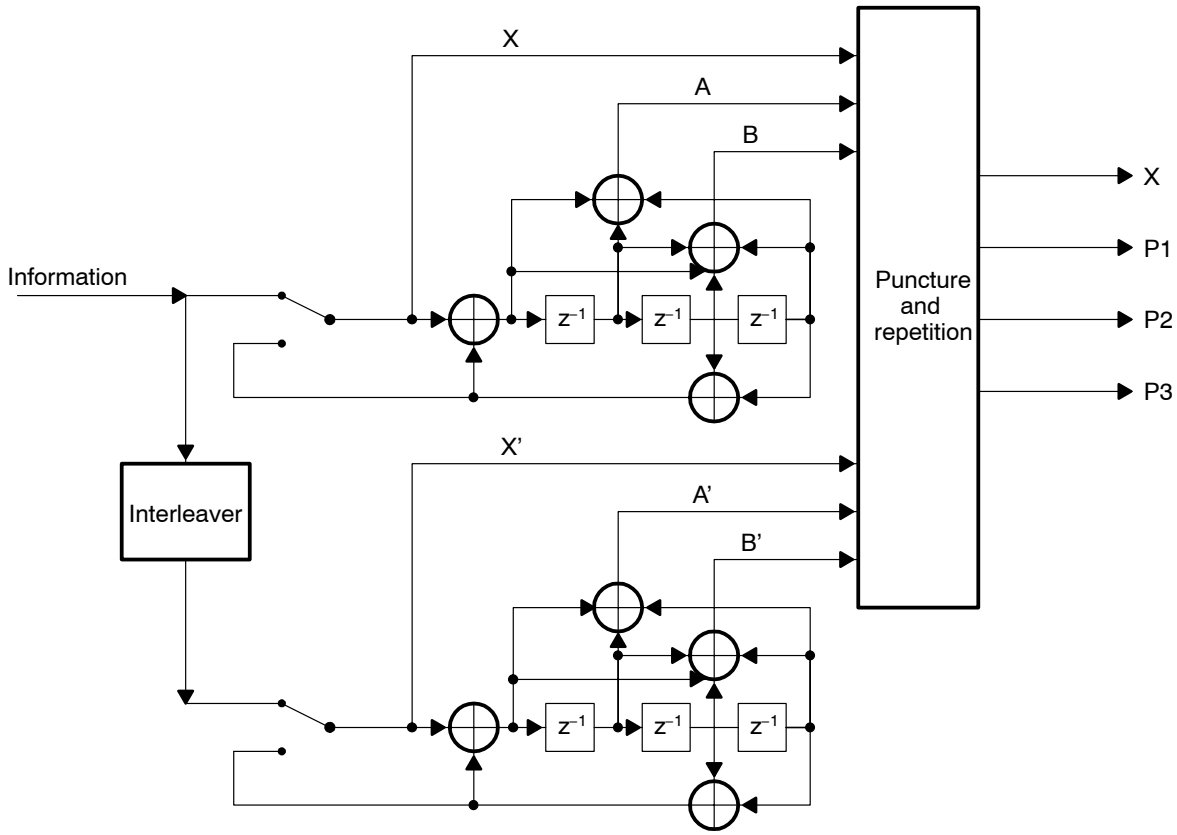
The TCP provides:

- High performance:
  - Very low processing delay because of the highly paralleled architecture allowing 8 iterations of a 2Mbps 3GPP channel to be decoded in less than 2 ms and a 1Mbps IS-2000 channel in less than 3.5ms.
  - Processing delay can be further reduced by enabling a stopping criteria algorithm.
  - TCP and DSP can run full speed in parallel.
- System cost optimization:
  - Reduces board space and power consumption by performing turbo-decoding on-chip.
  - Communication between the DSP and the TCP is performed through a high performance DMA engine, the enhanced DMA (EDMA).
  - TCP uses its own optimized working memories.
- High flexibility to cope with standard evolutions:
  - Accepts all IS2000, 3GPP rates and polynomials.
  - Accepts any frame length from 40 (3GPP minimum frame size) up to 20730 (IS2000 maximum frame size).
  - Can use any kind of interleaver.
  - Frees-up DSP resources.

2 Introduction

Encoding is done as shown in Figure 1. The 3GPP and IS2000 turbo encoders employ two recursive, systematic, convolutional (RSC) encoders connected in parallel, with an interleaver, the turbo interleaver, preceding the second recursive convolutional encoder. The two recursive convolutional codes are called the constituent encoders of the turbo code and have a constraint length K = 4.

Figure 1. 3GPP and IS2000 Turbo Encoder Block Diagram



Switches in upper position for information bits and in lower position for tail bits

The outputs of the constituent encoders are punctured and repeated (F denotes the frame size; X and X' are systematic data; A, B, A', and B' are parity data; X', A', and B' are the interleaved versions of X, A, and B data):

- Data rate 1/2 ($2 \times F$ bits):

  $X_0A_0X_1A'_1X_2A_2X_3A'_3\ldots$

- Data rate 1/3 ($3 \times F$ bits):

  $X_0A_0A'_0X_1A_1A'_1\ X_2A_2A'_2\ X_3A_3A'_3\ \ldots$

- Date rate 1/4 ($4 \times F$ bits):

  $X_0A_0B_0B'_0X_1A_1A'_1B'_1X_2A_2B_2B'_2X_3A_3A'_3B'_3\ldots$

For the tail bits, the sequence is:

- IS2000 tail rate 1/2 and 3GPP tail rate 1/3: 12 bits

  $X_FA_FX_{F+1}A_{F+1}X_{F+2}A_{F+2}X'_FA'_tX'_{t+1}A'_{t+1}X'_{t+2}A'_{t+2}$

- IS2000 tail rate 1/3: 18 bits (systematic bit repeated twice)

  $X_FX_FA_FX_{F+1}X_{F+1}A_{F+1}X_{F+2}X_{F+2}A_{F+2}X'_FX'_FA'_FX'_{F+1}X'_{F+1}A'_{F+1}X'_{F+2}X'_{F+2}$
  A'_{F+2}

- IS2000 tail rate 1/4: 24 bits (systematic bit repeated twice)

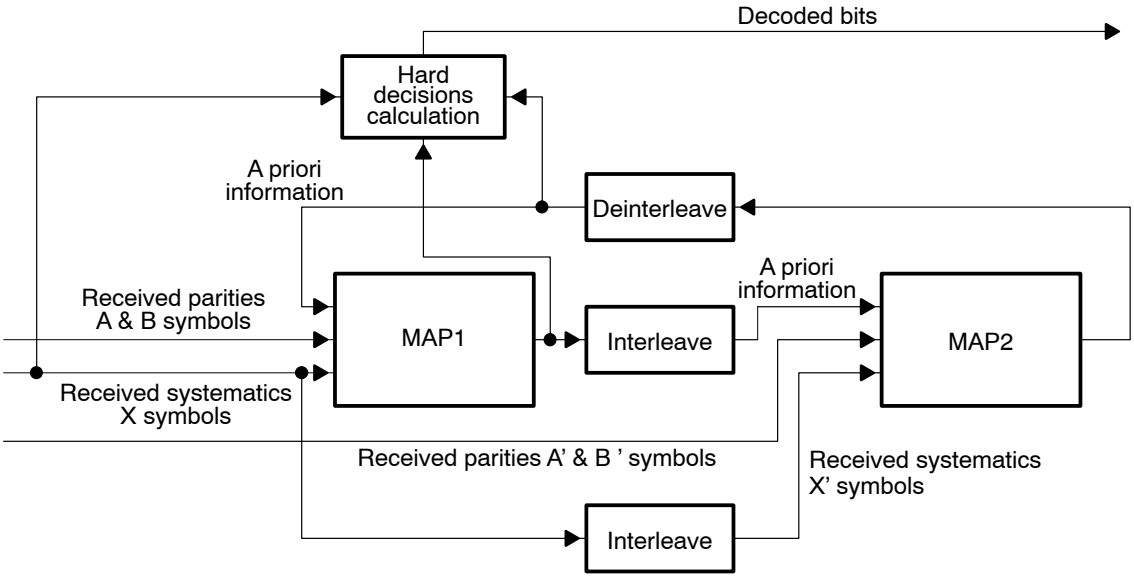
  $X_FX_FA_FB_F\ X_{F+1}X_{F+1}A_{F+1}B_{F+1}\ X_{F+2}X_{F+2}A_{F+2}B_{F+2}X'_FX'_FA'_FB'_F$
  $X'_{F+1}X'_{F+1}A'_{F+1}B'_{F+1}\ X'_{F+2}X'_{F+2}A'_{F+2}B'_{F+2}$

The decoding process is an iterative algorithm based on simple decoders individually matched to the RSC codes. A generic 3GPP and IS2000 turbo decoder is shown in Figure 2.

Each decoder sends *a posteriori* likelihood estimates of the decoded bits to the other decoder, and uses the corresponding estimates from the other decoder as *a priori* likelihood. The *a priori* information is seen as beforehand knowledge, meaning that some messages are more likely to occur than others. *A posteriori* information adds to the *a priori* information the knowledge gained by the decoding.

The uncoded information bits (corrupted by the noisy channel) are available to each decoder to initialize the *a priori* likelihoods. The decoders use the MAP (Maximum a Posteriori) bitwise decoding algorithm that requires the same number of states as the well known Viterbi algorithm. The turbo decoder iterates between the outputs of the two constituent decoders until it reaches satisfactory convergence. The final output is a hard-quantized version of the likelihood estimates of the decoders.

Figure 2. 3GPP and IS2000 Turbo Decoder Block Diagram



3 Overview

The DSP controls the operation of the TCP (Figure 3) using memory-mapped registers. The DSP typically sends and receives data using synchronized EDMA transfers through the 64-bit EDMA bus. The TCP sends two synchronization events to the EDMA: a receive event (TCPREVT) and a transmit event (TCPXEVT).

The processing unit implements the MAX*-LOG-MAP approximation of the BCJR algorithm (see L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theor*y, vol. IT–20, pp. 284–287, Mar. 1974).

The TCP has two fundamental modes: standalone (SA) and shared processing (SP).

In SA mode, the TCP iterates a given number of times and outputs hard decisions. In SP mode, the TCP executes a single MAP decode and outputs extrinsic information (soft information). SA mode is *typically* used for frame sizes smaller or equal to 5114; whereas, SP mode must be used for frames strictly larger than 5114. Table 1 describes which mode to use depending on the frame size.

The TCP input data corresponds to channel log-likelihood ratios scaled on 8 bits and output data to hard-decisions (SA mode) or extrinsics (SP mode) scaled on 7 bits.

Figure 3. TCP Block Diagram

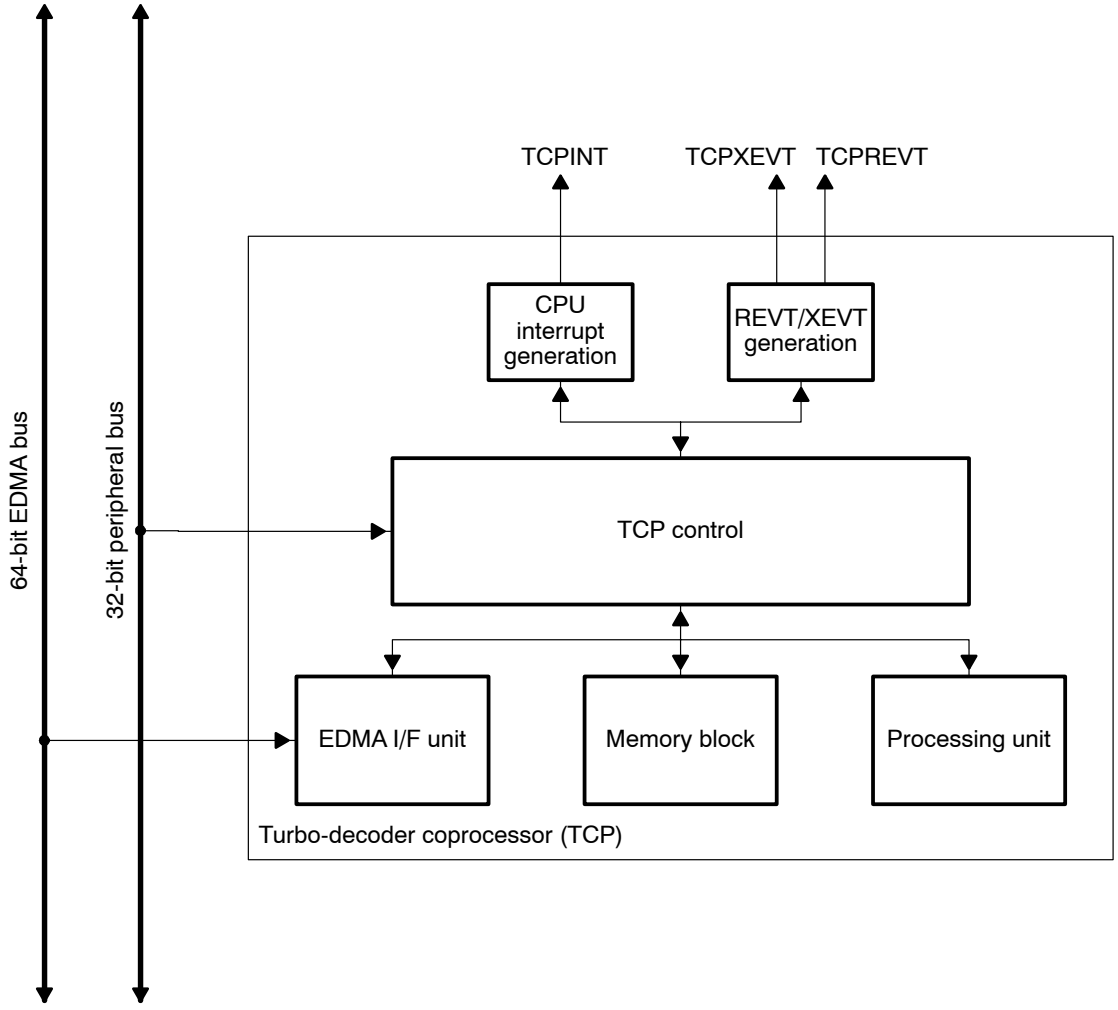


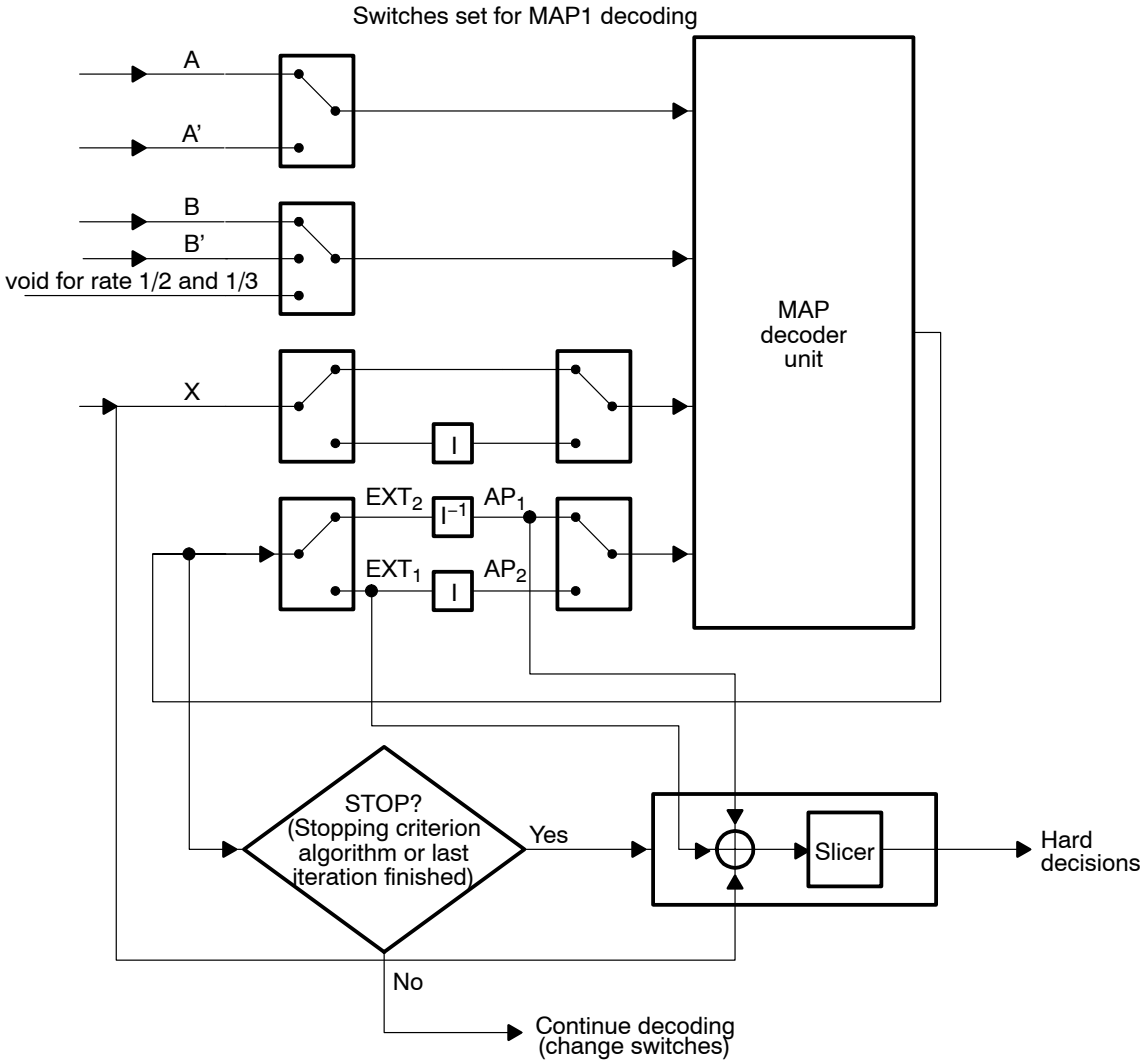
Table 1. Frame Sizes for Standalone (SA) Mode and Shared-Processing (SP) Mode

Frame Size (F)	TCP Mode
40 R F R 5114	Standalone or shared processing
F > 5114	Shared processing

4 Standalone (SA) Mode

In standalone (SA) mode, the DSP sends the systematic and parity data, and the interleaver table. The TCP then works independently of the DSP (stand-alone), iterates a user-defined maximum number of times, and outputs hard decision data. In this mode, minimum DSP processing is required. A stopping criteria can be enabled to reduce the processing delay (see section 4.3). The SA mode is shown Figure 4.

Figure 4. Standalone (SA) Mode Block Diagram



EXT_1 = extrinsics after MAP1 AP_1 = a priori for MAP1
EXT_2 = extrinsics after MAP2 AP_2 = a priori for MAP2

4.1 Input Data Format

4.1.1 Systematic and Parity Data

Symbols (data) have to be quantized on 8 bits as (5,3) numbers, that is, SIIII.FFF (where S = sign bit, I = integer, F = fractional bit). Depending on the rate, Table 2, Table 3, and Table 4 show how data must be organized in the DSP memory The base address must be double-word aligned. For big-endian configuration, see the TCP endian register (TCPEND) in section 6.15.

Table 2. Rate 1/2 Systematic/Parity Data: Periodicity of 4 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	A'1	X1	A0	X0
Base + 4h	A'3	X3	A2	X2
Base + 8h	…			

Table 3. Rate 1/3 Systematic/Parity Data: Periodicity of 3 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X1	A'0	A0	X0
Base + 4h	A2	X2	A'1	A1
Base + 8h	…			A'2

Table 4. Rate 1/4 Systematic/Parity Data: Periodicity of 8 Bytes

	Data – Little Endian Representation			
Address (bytes)	**MSB**			**LSB**
Base	B'0	B0	A0	X0
Base + 4h	B'1	A'1	A1	X1
Base + 8h	B'2	B2	A2	X2
Base + Ch	B'3	A'3	A3	X3
Base + 10h	…			

4.1.2 Interleaver Indexes

Each index is a 13-bit value being effectively saved as 16 bits right-justified. Given an index j, an interleaver table t, and a data x, the interleaved data x is given as $x' = x[t[j]]$. Table 5 shows how data must be organized in the memory Base address must be double-word aligned. For big-endian configurations, see the TCP endian register (TCPEND) in section 6.15.

Table 5. Interleaver Data

	Data	
Address (bytes)	**MSB**	**LSB**
Base	Index1	Index0
Base + 4h	Index3	Index2
Base + 8h	…	

4.2 Output Data Format

Hard decisions coming from the TCP are 32-bit word-packed with the symbol 0 hard decision bit in the least-significant-bit position. Their destination storage base address must be double-word aligned. Moreover the buffer length must contain an even number of words.

4.3 Stopping Criteria

A stopping criteria is enabled in the TCP input configuration register 2 (TCPIC2), in section 6.3, by writing a threshold value. When enabled (nonzero value), the stopping criteria algorithm calculates a SNR ratio based on the extrinsic data and compares it with the threshold. If the estimated SNR ratio is greater than or equal to the threshold, then the TCP computes directly the hard decisions, thus possibly stopping before the maximum number of iterations has been reached and therefore improving the processing delay.

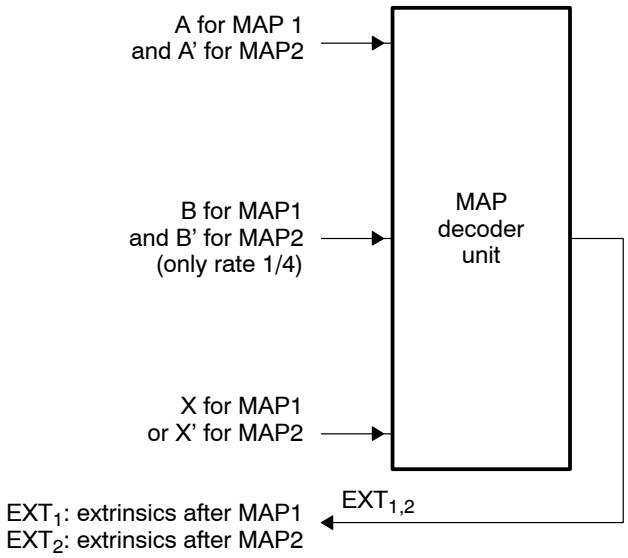
Larger thresholds give better results but require more iterations. Smaller thresholds require fewer iterations and can give poorer results in terms of Bit Error Rates (BER).

You can read from the output parameters the actual number of iterations run.

5 Shared-Processing (SP) Mode

In shared-processing (SP) mode, the DSP sends systematic and parity data, and a priori data. The TCP performs one single MAP decode and outputs extrinsic data. A priori data for MAP1 is obtained by deinterleaving the extrinsic data from previous MAP2, and a priori data for MAP2 is obtained by interleaving the extrinsics data from previous MAP1. An overview of the SP mode is shown in Figure 5. Note that the systematic and parity data to be sent to the TCP has to be *demultiplexed* from the original flow described in section 5.1. The input data demultiplexing, interleaving, deinterleaving operations, hard decision calculation, and any stopping criteria algorithm must be performed by the DSP.

Figure 5. Shared-Processing (SP) Mode Block Diagram



5.1 Input Data Format

5.1.1 Systematics and Parity Data

The original systematic and parity data is organized as described in section 2. The DSP has to split the data for MAP1 as shown in Table 6, Table 7, and Table 8; and for MAP2 as shown in Table 9, Table 10, and Table 11. Base address must be double-word aligned. Interleaved systematic data (X') must be calculated by the DSP given the interleaver table (see section 4.1.2). For big-endian configuration, refer to section 7.

Table 6. Rate 1/2 Systematic/Parity Data for MAP1: Periodicity of 8 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X3	X2	X1	X0
Base + 4h	0	0	A2	A0
Base + 8h	X7	X6	X5	X4
Base + Ch	0	0	A6	A4
Base + 10h	…			

Table 7. Rate 1/3 Systematic/Parity Data for MAP1:Periodicity of 8 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X3	X2	X1	X0
Base + 4h	A3	A2	A1	A0
Base + 8h	X7	X6	X5	X4
Base + Ch	A7	A6	A5	A4
Base + 10h	…			

Table 8. Rate1/4 Systematic/Parity Data for MAP1:Periodicity of 20 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X3	X2	X1	X0
Base + 4h	A3	A2	A1	A0
Base + 8h	X5	X4	B2	B0
Base + Ch	A5	A4	X7	X6
Base + 10h	B6	B4	A7	A6
Base + 14h	…			

Table 9. Rate 1/2 Systematic/Parity Data for MAP2: Periodicity of 8 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X'3	X'2	X'1	X'0
Base + 4h	0	0	A'3	A'1
Base + 8h	X'7	X'6	X'5	X'4
Base + Ch	0	0	A'7	A'5
Base + 10h	…			

Table 10. Rate 1/3 Systematic/Parity Data for MAP2: Periodicity of 8 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X'3	X'2	X'1	X'0
Base + 4h	A'3	A'2	A'1	A'0
Base + 8h	X'7	X'6	X'5	X'4
Base + Ch	A'7	A'6	A'5	A'4
Base + 10h	…			

Table 11. Rate1/4 Systematic/Parity Data for MAP2: Periodicity of 20 Bytes

	Data			
Address (bytes)	**MSB**			**LSB**
Base	X'3	X'2	X'1	X'0
Base + 4h	B'3	B'2	B'1	B'0
Base + 8h	X'5	X'4	A'3	A'1
Base + Ch	B'5	B'4	X'7	X'6
Base + 10h	A'7	A'5	B'7	B'6
Base + 14h	…			

5.1.2 A priori Data

A priori data for MAP1 and MAP2 must be organized as described in Table 12 (the base address must be double-word aligned). For big-endian configuration, refer to section 7.

Table 12. A priori Data

	Data			
Address (bytes)	**MSB**			**LSB**
Base	AP3	AP2	AP1	AP0
Base + 4h	AP7	AP6	AP5	AP4
Base + 8h				

5.2 Output Data Format

The TCP delivers 32-bit word-packed extrinsic data. Each extrinsic is a (5,2) number, that is, SIIII.FF (where S = sign bit, I = integer, F = fractional bit) and is right-justified with 0 in the most-significant-bit position. Their destination storage base address must be double-word aligned. Moreover the buffer length must contain an even number of words.

6 Registers

The TCP contains several memory-mapped registers accessible by way of the CPU, QDMA, and EDMA. A peripheral-bus access is faster than an EDMA-bus access for isolated accesses (typically when accessing control registers). EDMA-bus accesses are intended to be used for EDMA transfers and are meant to provide maximum throughput to/from the TCP.

The memory map is listed in Table 13. All TCP memories (systematic and parity, interleaver, hard decisions, a priori, and extrinsic) are regarded as FIFOs by the DSP, meaning you do not have to perform any indexing on the addresses.

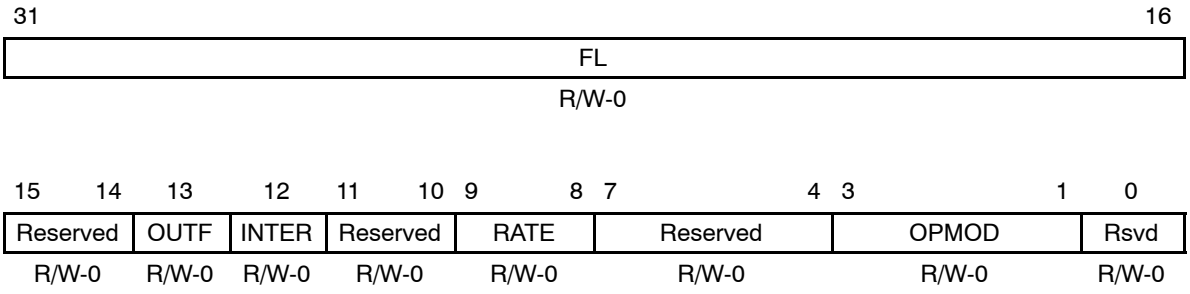
Table 13. TCP Registers

Start Address (hex)				
EDMA Bus	**Peripheral Bus**	**Acronym**	**Register Name**	**Section**
5800 0000	01BA 0000	TCPIC0	TCP input configuration register 0	6.1
5800 0004	01BA 0004	TCPIC1	TCP input configuration register 1	6.2
5800 0008	01BA 0008	TCPIC2	TCP input configuration register 2	6.3
5800 000C	01BA 000C	TCPIC3	TCP input configuration register 3	6.4
5800 0010	01BA 0010	TCPIC4	TCP input configuration register 4	6.5
5800 0014	01BA 0014	TCPIC5	TCP input configuration register 5	6.6
5800 0018	01BA 0018	TCPIC6	TCP input configuration Register 6	6.7
5800 001C	01BA 001C	TCPIC7	TCP input configuration register 7	6.8
5800 0020	01BA 0020	TCPIC8	TCP input configuration register 8	6.9
5800 0024	01BA 0024	TCPIC9	TCP input configuration register 9	6.10
5800 0028	01BA 0028	TCPIC10	TCP input configuration register 10	6.11
5800 002C	01BA 002C	TCPIC11	TCP input configuration register 11	6.12
5800 0030	01BA 0030	TCPOUT	TCP output parameters register	6.13
5802 0000	–	TCPSP	TCP systematics and parities memory	9.3.1
5804 0000	–	TCPEXT	TCP extrinsics memory	9.3.1
5806 0000	–	TCPAP	TCP a priori memory	9.3.1
5808 0000	–	TCPINTER	TCP interleaver memory	9.2.1
580A 0000	–	TCPHD	TCP hard decisions memory	9.2.1
–	01BA 0038	TCPEXE	TCP execution register	6.14
–	01BA 0040	TCPEND	TCP endian register	6.15
–	01BA 0050	TCPERR	TCP error register	6.16
–	01BA 0058	TCPSTAT	TCP status register	6.17

6.1 TCP Input Configuration Register 0 (TCPIC0)

The TCP input configuration register 0 (TCPIC0) is shown in Figure 6 and described in Table 14. TCPIC0 is used to configure the TCP.

Figure 6. TCP Input Configuration Register 0 (TCPIC0)



Legend: R/W = Read/Write; -*n* = value after reset

Table 14. TCP Input Configuration Register 0 (TCPIC0) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–16	FL	OF(*value*)	40–20730	Frame length. Number of symbols in the frame to be decoded (not including tail symbols).
15–14	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
13	OUTF	OF(*value*)		Output parameters read flag (SA mode only; in SP mode, must be cleared to 0).
		NO	0	No REVT generation. Output parameters are not read via EDMA.
		YES	1	REVT generation. Output parameters are read via EDMA.
12	INTER	OF(*value*)		Interleaver write flag.
		NO	0	Interleaver table is not sent to the TCP (required for SP mode)
		YES	1	Interleaver table is sent to the TCP (required for SA mode)
11–10	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation TCP_IC0_*field_symval*

Table 14. TCP Input Configuration Register 0 (TCPIC0) Field Descriptions (Continued)

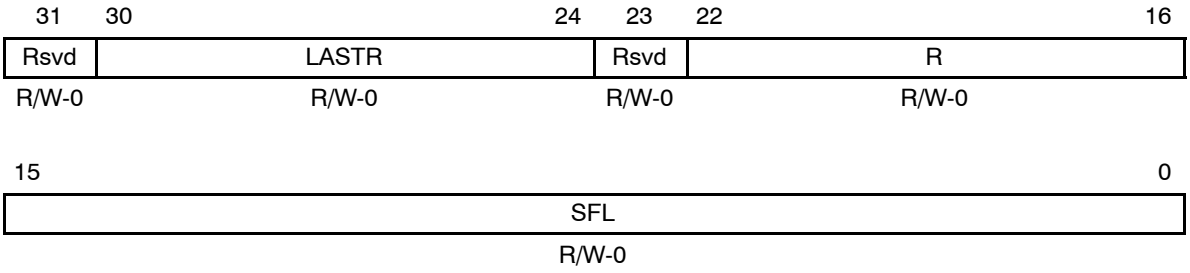
Bit	*field*†	*symval*†	Value	Description
9–8	RATE	OF(*value*)	0–3h	Code rate.
		–	0	Reserved
		1_2	1h	Rate 1/2
		1_3	2h	Rate 1/3
		1_4	3h	Rate 1/4
7–4	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3–1	OPMOD	OF(*value*)	0–7h	Operational mode.
		SA	0	SA mode
		–	1h–3h	Reserved
		MAP1A	4h	SP mode MAP1 (first iteration)
		MAP1B	5h	SP mode MAP1 (any other iteration)
		–	6h	Reserved
		MAP2	7h	SP mode MAP2
0	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation TCP_IC0_*field_symval*

6.2 TCP Input Configuration Register 1 (TCPIC1)

The TCP input configuration register 1 (TCPIC1) is shown in Figure 7 and described in Table 15. TCPIC1 is used to configure the TCP.

Figure 7. TCP Input Configuration Register 1 (TCPIC1)



Legend: R/W = Read/Write; -*n* = value after reset

Table 15. TCP Input Configuration Register 1 (TCPIC1) Field Descriptions

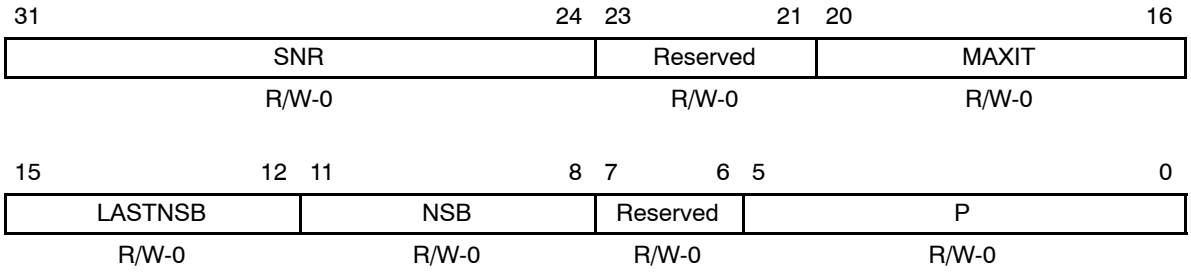
Bit	***field***†	***symval***†	Value	Description
31	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
30–24	LASTR	OF(*value*)		Last subframe reliability length – 1: (SP mode only; don't care in SA mode). Number of symbols – 1 to be used in the reliability portion the last subframe.
23	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
22–16	R	OF(*value*)	39–127	Reliability length – 1 (from 39 to 127). Number of symbols – 1 to be used in the reliability portion of a frame or subframe.
15–0	SFL	OF(*value*)	98–5114	Subframe length (from 98 to 5114): (SP mode only; don't care in SA mode). Number of symbols in a subframe including the header and tail prolog symbols. Maximum is 5114.

† For CSL implementation, use the notation TCP_IC1_*field_symval*

6.3 TCP Input Configuration Register 2 (TCPIC2)

The TCP input configuration register 2 (TCPIC2) is shown in Figure 8 and described in Table 16. TCPIC2 is used to configure the TCP.

Figure 8. TCP Input Configuration Register 2 (TCPIC2)



Legend: R/W = Read/Write; -*n* = value after reset

Table 16. TCP Input Configuration Register 2 (TCPIC2) Field Descriptions

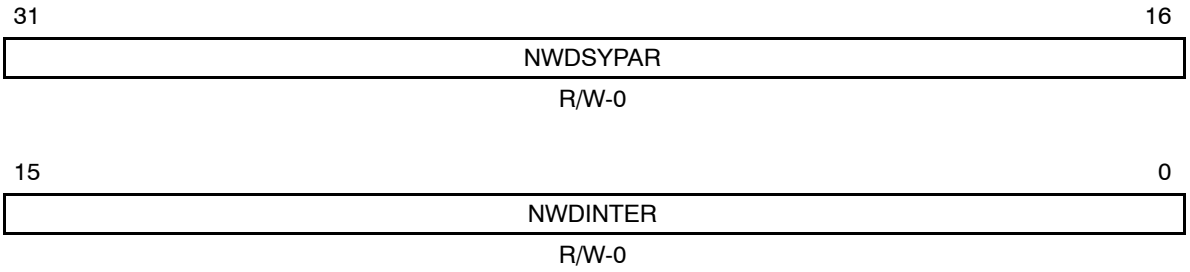
Bit	*field*†	*symval*†	Value	Description
31–24	SNR	OF(*value*)	0–100	SNR threshold (from 0 to 100) (SA mode only; don't care in SP mode).
		–	0	Disables stopping criteria.
23–21	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
20–16	MAXIT	OF(*value*)	0–31	Maximum number of iterations (from 0 to 32) (SA mode only; don't care in SP mode).
		–	0	Sets MAXIT to 32.
15–12	LASTNSB	OF(*value*)	0–15	Number of subblocks in the last subframe (SP mode only; don't care in SA mode).
11–8	NSB	OF(*value*)	0–15	Number of subblocks.
7–6	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
5–0	P	OF(*value*)	24–48	Prolog size (from 24 to 48). Number of symbols for the prolog. TCP forces value to 24, if the size is smaller than 24. In SP mode, P must be a multiple of 8 for rate 1/2 and 1/3, and a multiple of 16 for rate 1/4.

† For CSL implementation, use the notation TCP_IC2_*field_symval*

6.4 TCP Input Configuration Register 3 (TCPIC3)

The TCP input configuration register 3 (TCPIC3) is shown in Figure 9 and described in Table 17. TCPIC3 is used to inform the TCP on the EDMA data flow segmentation.

Figure 9. TCP Input Configuration Register 3 (TCPIC3)



Legend: R/W = Read/Write; -*n* = value after reset

Table 17. TCP Input Configuration Register 3 (TCPIC3) Field Descriptions

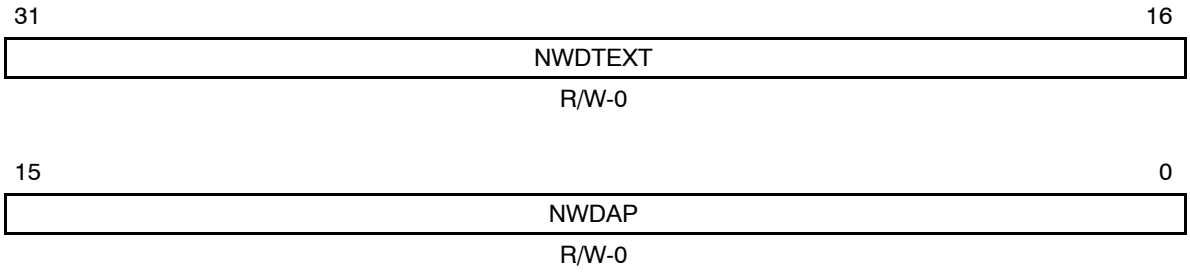
Bit	***field***†	***symval***†	Value	Description
31–16	NWDSYPAR	OF(*value*)	0–FFFFh	Number of systematic and parity words per XEVT.
15–0	NWDINTER	OF(*value*)	0–FFFFh	Number of interleaver words per XEVT (SA mode only; don't care in SP mode).

† For CSL implementation, use the notation TCP_IC3_*field_symval*

6.5 TCP Input Configuration Register 4 (TCPIC4)

The TCP input configuration register 4 (TCPIC4) is shown in Figure 10 and described in Table 18. TCPIC4 is used to inform the TCP on the EDMA data flow segmentation.

Figure 10. TCP Input Configuration Register 4 (TCPIC4)



Legend: R/W = Read/Write; *-n* = value after reset

Table 18. TCP Input Configuration Register 4 (TCPIC4) Field Descriptions

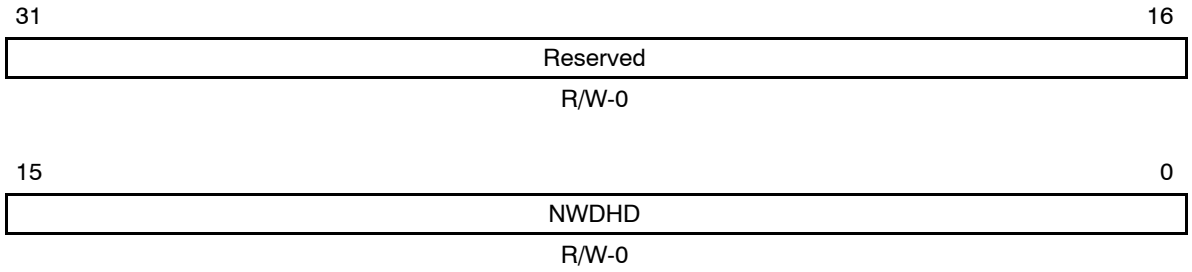
Bit	*field*†	*symval*†	Value	Description
31–16	NWDTEXT	OF(*value*)	0–FFFFh	Number of extrinsic words per REVT (SP mode only; don't care in SA mode).
15–0	NWDAP	OF(*value*)	0–FFFFh	Number of a priori words per XEVT (SP mode only; don't care in SA mode).

† For CSL implementation, use the notation TCP_IC4_*field_symval*

6.6 TCP Input Configuration Register 5 (TCPIC5)

The TCP input configuration register 5 (TCPIC5) is shown in Figure 11 and described in Table 19. TCPIC5 is used to inform the TCP on the EDMA data flow segmentation.

Figure 11. TCP Input Configuration Register 5 (TCPIC5)



Legend: R/W = Read/Write; -*n* = value after reset

Table 19. TCP Input Configuration Register 5 (TCPIC5) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–16	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
15–0	NWDHD	OF(*value*)	0–FFFFh	Number of hard decisions words per REVT (SA mode only; don't care in SP mode).

† For CSL implementation, use the notation TCP_IC5_*field_symval*

6.7 TCP Input Configuration Register 6 (TCPIC6)

The TCP input configuration register 6 (TCPIC6) is shown in Figure 12 and described in Table 20. TCPIC6 is used to set the tail bits used by the TCP.

Tail bits value must be set as following:

- SA mode and SP mode MAP1:
  - For IS2000 rate 1/2 and 3GPP rate 1/3:

31–24	23–16	15–8	7–0
0	X_{F+2}	X_{F+1}	X_F

  - For IS2000 rate 1/3 and 1/4: the systematics are repeated twice at the transmitter but are not necessarily the same at the receiver. You can program the first (X^1_{F+2}) or the second (X^2_{F+2}) received systematic tail symbol, or the addition and saturation of the two (*).

31–24	23–16	15–8	7–0
0	$(X^1_{F+2} + X^2_{F+2})$* or X^1_{F+2} or X^2_{F+2}	$(X^1_{F+1} + X^2_{F+1})$* or X^1_{F+1} or X^2_{F+1}	$(X^1_F + X^2_F)$* or X^1_F or X^2_F

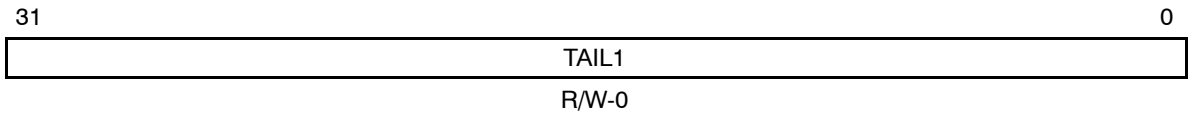
- SP mode MAP2:
  - For IS2000 rate 1/2 and 3GPP rate 1/3:

31–24	23–16	15–8	7–0
0	X'_{F+2}	X'_{F+1}	X'_F

  - For IS2000 rate 1/3 and 1/4: the systematics are repeated twice at the transmitter but are not necessarily the same at the receiver. You can program the first (X'^1_{F+2}) or the second (X'^2_{F+2}) received systematic tail symbol, or the addition and saturation of the two (*).

31–24	23–16	15–8	7–0
0	$(X'^1_{F+2} + X'^2_{F+2})$* or X'^1_{F+2} or X'^2_{F+2}	$(X'^1_{F+1} + X'^2_{F+1})$* or X'^1_{F+1} or X'^2_{F+1}	$(X'^1_F + X'^2_F)$* or X'^1_F or X'^2_F

Figure 12. TCP Input Configuration Register 6 (TCPIC6)



Legend: R/W = Read/Write; *-n* = value after reset

Table 20. TCP Input Configuration Register 6 (TCPIC6) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–0	TAIL1	OF(*value*)	0–FFFF FFFFh	Tail bits.

† For CSL implementation, use the notation TCP_IC6_*field_symval*

6.8 TCP Input Configuration Register 7 (TCPIC7)

The TCP input configuration register 7 (TCPIC7) is shown in Figure 13 and described in Table 21. TCPIC7 is used to set the tail bits used by the TCP.

Tail bits value must be set as following:

- SA mode and SP mode MAP1 all rates:

31–24	23–16	15–8	7–0
0	A_{t+2}	A_{t+1}	A_t

- SP mode MAP2 rate 1/2 and rate 1/3:

31–24	23–16	15–8	7–0
0	A'_{t+2}	A'_{t+1}	A'_t

- SP mode MAP2 rate 1/4:

31–24	23–16	15–8	7–0
0	B'_{t+2}	B'_{t+1}	B'_t

Figure 13. TCP Input Configuration Register 7 (TCPIC7)

31	0
TAIL2	
R/W-0	

Legend: R/W = Read/Write; *-n* = value after reset

Table 21. TCP Input Configuration Register 7 (TCPIC7) Field Descriptions

Bit	***field***†	***symval***†	Value	Description
31–0	TAIL2	OF(*value*)	0–FFFF FFFFh	Tail bits.

† For CSL implementation, use the notation TCP_IC7_*field_symval*

6.9 TCP Input Configuration Register 8 (TCPIC8)

The TCP input configuration register 8 (TCPIC8) is shown in Figure 14 and described in Table 22. TCPIC8 is used to set the tail bits used by the TCP.

Tail bits value must be set as following:

- SA mode and SP mode MAP1:
  - For rate 1/2 and 1/3:

31–24	23–16	15–8	7–0
0	0	0	0

  - For rate 1/4:

31–24	23–16	15–8	7–0
0	B_{t+2}	B_{t+1}	B_t

- SP mode MAP2:
  - For rate 1/2 and 1/3:

31–24	23–16	15–8	7–0
0	0	0	0

  - For rate 1/4:

31–24	23–16	15–8	7–0
0	A'_{t+2}	A'_{t+1}	A'_t

Figure 14. TCP Input Configuration Register 8 (TCPIC8)

31	0
TAIL3	
R/W-0	

Legend: R/W = Read/Write; *-n* = value after reset

Table 22. TCP Input Configuration Register 8 (TCPIC8) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–0	TAIL3	OF(*value*)	0–FFFF FFFFh	Tail bits.

† For CSL implementation, use the notation TCP_IC8_*field_symval*

6.10 TCP Input Configuration Register 9 (TCPIC9)

The TCP input configuration register 9 (TCPIC9) is shown in Figure 15 and described in Table 23. TCPIC9 is used to set the tail bits used by the TCP.

Tail bits value must be set as following:

- For IS2000 rate 1/2 and 3GPP rate 1/3:

31–24	23–16	15–8	7–0
0	X'_{F+2}	X'_{F+1}	X'_{F}

- For IS2000 rate 1/3 and 1/4: the systematics are repeated twice at the transmitter but are not necessarily the same at the receiver. You can program the first (X'^{1}_{F+2}) or the second (X'^{2}_{F+2}) or the average of the two.

31–24	23–16	15–8	7–0
0	$(X'^{1}_{F+2} + X'^{2}_{F+2})/2$ or X'^{1}_{F+2} or X'^{2}_{F+2}	$(X'^{1}_{F+1} + X'^{2}_{F+1})/2$ or X'^{1}_{F+1} or X'^{2}_{F+1}	$(X'^{1}_{F} + X'^{2}_{F})/2$ or X'^{1}_{F} or X'^{2}_{F}

Figure 15. TCP Input Configuration Register 9 (TCPIC9)

31	0
TAIL4	
R/W-0	

Legend: R/W = Read/Write; *-n* = value after reset

Table 23. TCP Input Configuration Register 9 (TCPIC9) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–0	TAIL4	OF(*value*)	0–FFFF FFFFh	Tail bits. (SA mode only; don't care in SP mode.)

† For CSL implementation, use the notation TCP_IC9_*field_symval*

6.11 TCP Input Configuration Register 10 (TCPIC10)

The TCP input configuration register 10 (TCPIC10) is shown in Figure 16 and described in Table 24. TCPIC10 is used to set the tail bits used by the TCP.

Tail bits value must be set as following:

31−24	23−16	15−8	7−0
0	A'_{t+2}	A'_{t+1}	A'_{t}

Figure 16. TCP Input Configuration Register 10 (TCPIC10)

31		0
TAIL5		
R/W-0		

Legend: R/W = Read/Write; *-n* = value after reset

Table 24. TCP Input Configuration Register 10 (TCPIC10) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31−0	TAIL5	OF(*value*)	0−FFFF FFFFh	Tail bits. (SA mode only; don't care in SP mode.)

† For CSL implementation, use the notation TCP_IC10_*field_symval*

6.12 TCP Input Configuration Register 11 (TCPIC11)

The TCP input configuration register 11 (TCPIC11) is shown in Figure 17 and described in Table 25. TCPIC11 is used to set the tail bits used by the TCP.

Tail bits value must be set as following:

- For rate 1/4:

31–24	23–16	15–8	7–0
0	B'_{t+2}	B'_{t+1}	B'_{t}

- For rate 1/2 and 1/3:

31–24	23–16	15–8	7–0
0	0	0	0

Figure 17. TCP Input Configuration Register 11 (TCPIC11)

31	0
TAIL6	
R/W-0	

Legend: R/W = Read/Write; *-n* = value after reset

Table 25. TCP Input Configuration Register 11 (TCPIC11) Field Descriptions

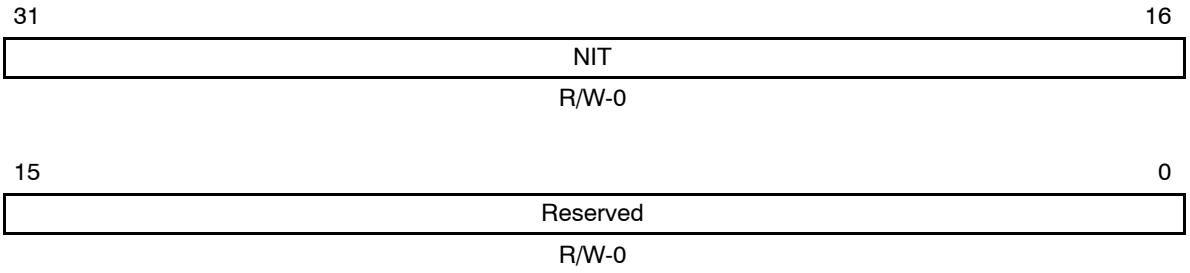
Bit	***field***†	***symval***†	Value	Description
31–0	TAIL6	OF(*value*)	0–FFFF FFFFh	Tail bits. (SA mode only; don't care in SP mode.)

† For CSL implementation, use the notation TCP_IC11_*field_symval*

6.13 TCP Output Parameter Register (TCPOUT)

The TCP output parameter register (TCPOUT) is shown in Figure 18 and described in Table 26.

Figure 18. TCP Output Parameter Register (TCPOUT)



Legend: R/W = Read/Write; *-n* = value after reset

Table 26. TCP Output Parameter Register (TCPOUT) Field Descriptions

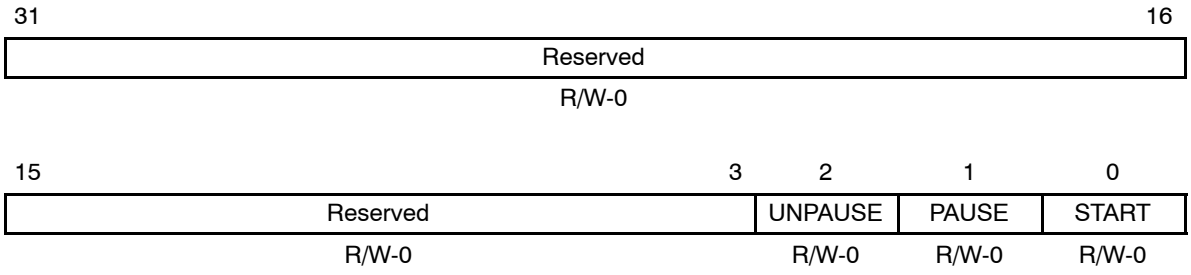
Bit	*field*†	*symval*†	Value	Description
31–16	NIT	OF(*value*)	0–FFFFh	Indicates the number of executed iterations – 1 and has no meaning in SP mode.
15–0	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

† For CSL implementation, use the notation TCP_OUT_*field_symval*

6.14 TCP Execution Register (TCPEXE)

The TCP execution register (TCPEXE) is shown in Figure 19 and described in Table 27.

Figure 19. TCP Execution Register (TCPEXE)



Legend: R/W = Read/Write; -*n* = value after reset

Table 27. TCP Execution Register (TCPEXE) Field Descriptions

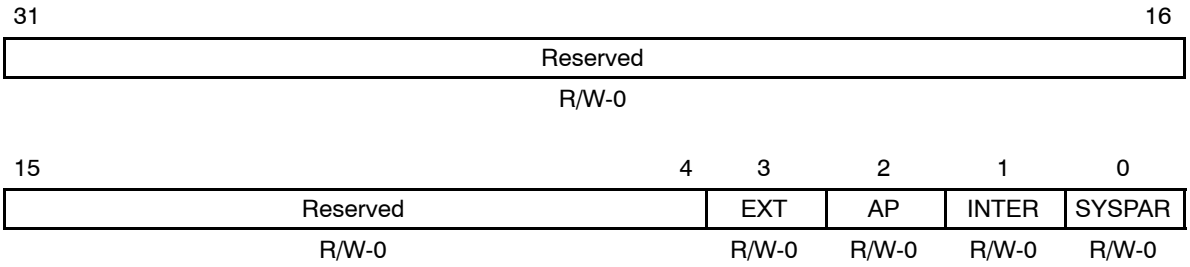
Bit	*field*†	*symval*†	Value	Description
31–3	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
2	UNPAUSE	OF(*value*)		Used to unpause the TCP.
		–	0	No effect.
		UNPAUSE	1	Unpause TCP.
1	PAUSE	OF(*value*)		Used to pause the TCP.
		–	0	No effect.
		PAUSE	1	Pause TCP.
0	START	OF(*value*)		Used to start the TCP.
		–	0	No effect.
		START	1	Start TCP.

† For CSL implementation, use the notation TCP_EXE_*field_symval*

6.15 TCP Endian Register (TCPEND)

The TCP endian register (TCPEND) is shown in Figure 20 and described in Table 28. TCPEND should only be used when the DSP is set to *big-endian mode*.

Figure 20. TCP Endian Register (TCPEND)



Legend: R/W = Read/Write; *-n* = value after reset

Table 28. TCP Endian Register (TCPEND) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–4	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3	EXT	OF(*value*)		Extrinsics memory format.
		32BIT	0	32-bit word packed
		NATIVE	1	Native format (7 bits logically right aligned on 8 bits)
2	AP	OF(*value*)		A prioris memory format.
		32BIT	0	32-bit word packed
		NATIVE	1	Native format (7 bits logically right aligned on 8 bits)
1	INTER	OF(*value*)		Interleaver indexes memory format.
		32BIT	0	32-bit word packed
		NATIVE	1	Native format (16 bits)
0	SYSPAR	OF(*value*)		Systematics and parities memory format.
		32BIT	0	32-bit word packed
		NATIVE	1	Native format (8 bits)

† For CSL implementation, use the notation TCP_END_*field_symval*

6.16 TCP Error Register (TCPERR)

The TCP error register (TCPERR) is shown in Figure 21 and described in Table 29.

Figure 21. TCP Error Register (TCPERR)

31	16
Reserved	
R/W-0	

15 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ACC	OP	INT	LR	R	MODE	Rsvd	SF	RATE	P	F	ERR
R/W-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 29. TCP Error Register (TCPERR) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–12	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
11	ACC	OF(*value*)		Memory access error bit.
		NO	0	No error.
		YES	1	TCP memories access not allowed in current state.
10	OP	OF(*value*)		Output parameters load error bit.
		NO	0	No error.
		YES	1	Output parameters load bit set to 1 in SP mode.
9	INT	OF(*value*)		Interleaver table load error bit.
		NO	0	No error.
		YES	1	Interleaver load bit set to 1 in SP mode.
8	LR	OF(*value*)		Last subframe reliability length error bit.
		NO	0	No error.
		YES	1	Last subframe reliability length < 40

† For CSL implementation, use the notation TCP_ERR_*field_symval*

Table 29. TCP Error Register (TCPERR) Field Descriptions (Continued)

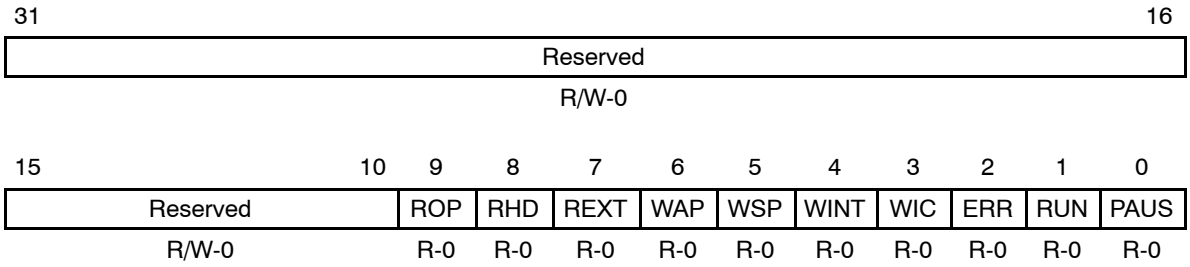
Bit	*field*†	*symval*†	Value	Description
7	R	OF(*value*)		Reliability length error bit.
		NO	0	No error.
		YES	1	Reliability length < 40
6	MODE	OF(*value*)		Operational mode error bit.
		NO	0	No error.
		YES	1	Operational mode is different from 4, 5, and 7.
5	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
4	SF	OF(*value*)		Subframe length error bit.
		NO	0	No error.
		YES	1	Subframe length > 5114.
3	RATE	OF(*value*)		Rate error bit.
		NO	0	No error.
		YES	1	Rate different from 1/2, 1/3, and 1/4.
2	P	OF(*value*)		Prolog length error bit.
		NO	0	No error.
		YES	1	Prolog length > 48.
1	F	OF(*value*)		Frame length error bit.
		NO	0	No error.
		YES	1	In SA mode frame length > 5114 or frame length < 40. In SP mode, frame length >20730 or frame length < 40.
0	ERR	OF(*value*)		Error bit.
		NO	0	No error.
		YES	1	Error has occurred.

† For CSL implementation, use the notation TCP_ERR_*field_symval*

6.17 TCP Status Register (TCPSTAT)

The TCP status register (TCPSTAT) is shown in Figure 22 and described in Table 30.

Figure 22. TCP Status Register (TCPSTAT)



Legend: R = Read only; R/W = Read/Write; -*n* = value after reset

Table 30. TCP Status Register (TCPSTAT) Field Descriptions

Bit	*field*†	*symval*†	Value	Description
31–10	Reserved	–	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
9	ROP	OF(*value*)		Defines if the TCP is waiting for output parameter data to be read.
		NREADY	0	Not waiting
		READY	1	Waiting
8	RHD	OF(*value*)		Defines if the TCP is waiting for hard decision data to be read.
		NREADY	0	Not waiting
		READY	1	Waiting
7	REXT	OF(*value*)		Defines if the TCP is waiting for extrinsic data to be read.
		NREADY	0	Not waiting
		READY	1	Waiting
6	WAP	OF(*value*)		Defines if the TCP is waiting for a priori data to be written.
		NREADY	0	Not waiting
		READY	1	Waiting

† For CSL implementation, use the notation TCP_STAT_*field_symval*

Table 30. TCP Status Register (TCPSTAT) Field Descriptions (Continued)

Bit	*field*†	*symval*†	Value	Description
5	WSP	OF(*value*)		Defines if the TCP is waiting for systematic and parity data to be written.
		NREADY	0	Not waiting
		READY	1	Waiting
4	WINT	OF(*value*)		Defines if the TCP is waiting for interleaver indexes to be written.
		NREADY	0	Not waiting
		READY	1	Waiting
3	WIC	OF(*value*)		Defines if the TCP is waiting for input control words to be written.
		NREADY	0	Not waiting
		READY	1	Waiting
2	ERR	OF(*value*)		Defines if the TCP has encountered an error.
		NO	0	No error.
		YES	1	Error
1	RUN	OF(*value*)		Defines if the TCP is running.
		NO	0	Not running.
		YES	1	Running.
0	PAUS	OF(*value*)		Defines if the TCP is paused.
		NO	0	No activity – waiting for start instruction
		YES	1	Paused

† For CSL implementation, use the notation TCP_STAT_*field_symval*

7 Endianness Issues

In big-endian mode only, the TCP endian register (TCPEND) should be programmed as shown in Table 31. This allows flexibility for you to save the data in the memory in its native format or as 32-bit words.

Table 31. TCP Endian Register Programming

Data	Native Format	Saved in DSP Memory as …	TCPEND	Section
Systematic and parity	8 bits	8 bits	SYSPAR = 1	7.1
		Packed on 32 bits	SYSPAR = 0	
Interleaver indexes	16 bits (13 bits right justified)	16 bits	INTER = 1	7.2
		Packed on 32 bits	INTER = 0	
A priori	8 bits (7 bits logically right justified)	8 bits	AP = 1	7.3
		Packed on 32 bits	AP = 0	
Extrinsic	8 bits (7 bits logically right justified)	8 bits	EXT = 1	7.4
		Packed on 32 bits	EXT = 0	

7.1 Systematic and Parity Data

When the data are saved in their 8-bit native format (SYSPAR = 1), they must be organized in the DSP memory as described in Table 32. When the data are packed on 32-bit words (SYSPAR = 0), they must be organized in the DSP memory as described in Table 33.

*Table 32. Systematic and Parity Data (SP*n*) When SYSPAR = 1*

Address (hex bytes)	Data
Base	SP0
Base + 1	SP1
Base + 2	SP2
Base + 3	SP3
Base + 4	SP4
Base + 5	SP5
Base + 6	SP6
Base + 7	SP7

*Table 33. Systematic and Parity Data (SP*n*) When SYSPAR = 0*

Address (hex bytes)	Data
Base	SP3
Base + 1	SP2
Base + 2	SP1
Base + 3	SP0
Base + 4	SP7
Base + 5	SP6
Base + 6	SP5
Base + 7	SP4

7.2 Interleaver Indexes

When the data are saved in their 16-bit native format (INTER = 1), they must be organized in the DSP memory as described in Table 34. When the data are packed on 32-bit words (INTER = 0), they must be organized in the DSP memory as described in Table 35.

*Table 34. Interleaver Indexes (INTER*n*) When INTER = 1*

Address (hex bytes)	Data
Base	INTER0
Base + 2	INTER1
Base + 4	INTER2
Base + 6	INTER3

*Table 35. Interleaver Indexes (INTER*n*) When INTER = 0*

Address (hex bytes)	Data
Base	INTER1
Base + 2	INTER0
Base + 4	INTER3
Base + 6	INTER2

7.3 A priori Data

When the data are saved in their 8-bit native format (AP = 1), they must be organized in the DSP memory as described in Table 36. When the data are packed on 32-bit words (AP = 0), they must be organized in the DSP memory as described in Table 37.

*Table 36. A priori Data (AP*n*) When AP = 1*

Address (hex bytes)	Data
Base	AP0
Base + 1	AP1
Base + 2	AP2
Base + 3	AP3
Base + 4	AP4
Base + 5	AP5
Base + 6	AP6
Base + 7	AP7

*Table 37. A priori Data (AP*n*) When AP = 0*

Address (hex bytes)	Data
Base	AP3
Base + 1	AP2
Base + 2	AP1
Base + 3	AP0
Base + 4	AP7
Base + 5	AP6
Base + 6	AP5
Base + 7	AP4

7.4 Extrinsic Data

When the data are saved in their 8-bit native format (EXT = 1), they must be organized in the DSP memory as described in Table 38. When the data are packed on 32-bit words (EXT = 0), they must be organized in the DSP memory as described in Table 39.

Table 38. Extrinsic Data (EXTn) When EXT = 1

Address (hex bytes)	Data
Base	EXT0
Base + 1	EXT1
Base + 2	EXT2
Base + 3	EXT3
Base + 4	EXT4
Base + 5	EXT5
Base + 6	EXT6
Base + 7	EXT7

Table 39. Extrinsic Data (EXTn) When EXT = 0

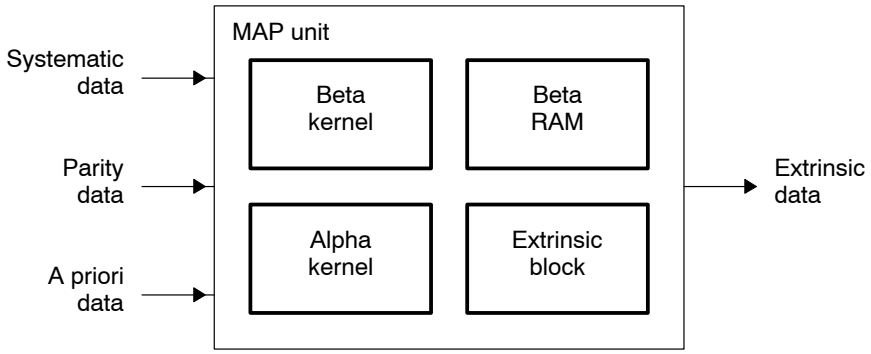
Address (hex bytes)	Data
Base	EXT3
Base + 1	EXT2
Base + 2	EXT1
Base + 3	EXT0
Base + 4	EXT7
Base + 5	EXT6
Base + 6	EXT5
Base + 7	EXT4

8 Architecture

The TCP processing unit (MAP unit) is shown in Figure 23. The beta kernel performs the MAP algorithm backward recursion and produces beta probabilities stored in the beta RAM. The alpha kernel performs the forward recursion producing alpha probabilities, and the extrinsic block computes the extrinsics to be stored in the extrinsics memory.

The processing unit implements the MAX* LOG-MAP algorithm using a sliding window technique that allows parallel processing and reduces dramatic memory requirements.

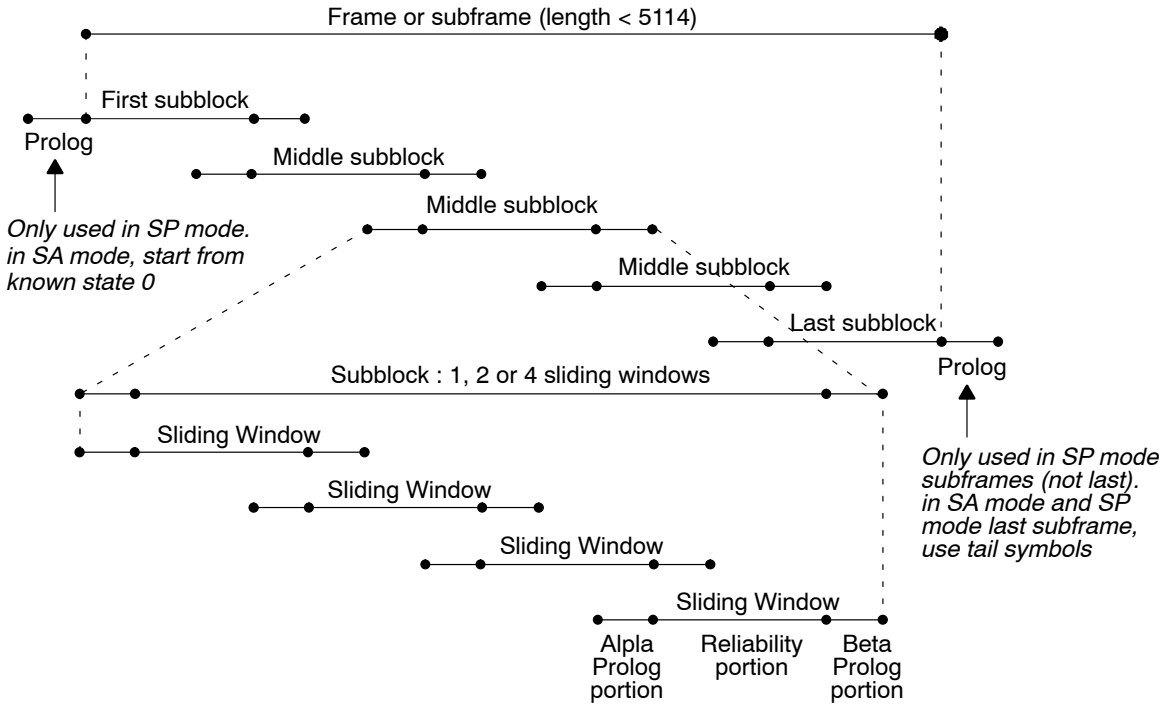
Figure 23. MAP Unit Block Diagram



8.1 Subblock and Sliding Window Segmentation

The MAP unit splits the input frame (standalone frame or shared processing subframe) into subblocks and sliding windows. More details are given in Figure 24.

Figure 24. Sliding Windows and Subblocks Segmentation (Example with 5 Subblocks)



The alpha and beta prolog portions are of equal length, referred to as the prolog length. The reliability portion length is user-defined as well as the the prolog length. The frame length, subframe length, and number of subblocks are also user-defined. The number of sliding windows is decided by the MAP unit based on the frame or subframe length (see Table 40).

Table 40. Number of Sliding Windows per Subblock (Nsw)

Frame/Subframe Length	Number of Sliding Windows per Subblock
From 40 to 128	1
From 129 to 256	2
From 257 to 5114	4

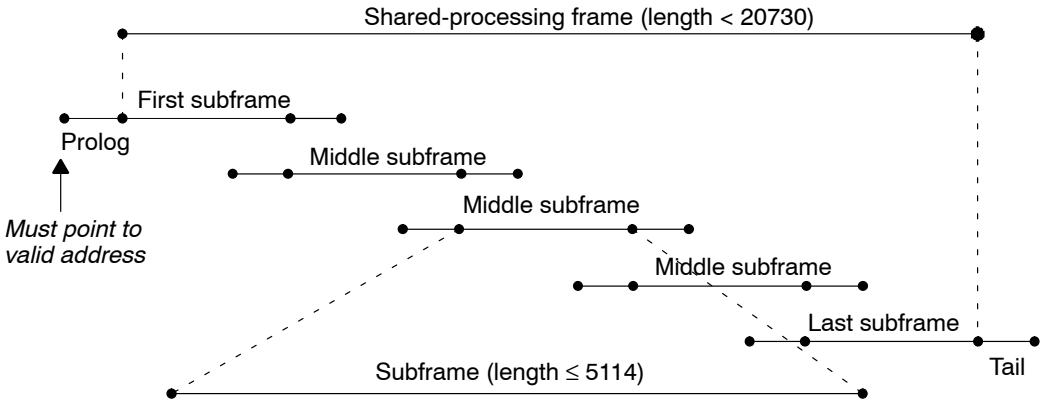
The MAP unit supports reliability sizes ranging from 40 to 128, the subblocks sizes range from 40 to 512 as there is a maximum of 4 sliding windows per subblock. Prolog sizes ranges from 24 to 48.

8.2 Subframe Segmentation (SP mode only)

A frame needs to be segmented into subframes when working in shared processing (SP) mode; therefore, the minimum number of subframes is 2. The subframe length should be chosen so that its reliable portion is less than 5114. The segmentation is shown in Figure 25. Each subframe is then further divided into subblocks and sliding windows as described in section 8.1.

All subframes except the last one must have the same length.

Figure 25. Shared Processing Subframe Segmentation (Example with 5 Subframes)



8.3 Reliability and Prolog Length Calculation

F: frame length (number of bits in a frame prior to turbo-encoding)

R: inverse code rate (there are $F \times R$ bits after turbo-encoding not including the tail bits)

P: prolog length (number of symbols to be used in the prolog not taking into account the rate)

Nsb: number of subblocks

Nsw: number of sliding windows per subblock

The prolog size is chosen depending on whether the code is punctured or nonpunctured. For nonpunctured codes, the prolog size should range between 24 to 32. This is equal to 3 to 4 times the number of states of the trellis. For punctured codes, the prolog should be larger with a maximum size of 48.

The reliability length should be chosen to optimally fill the pipeline in order to optimize the processing time. It can be computed from the frame length and the prolog length and must fill the following properties in order to get a correct decoding:

- Reliability length is from 40 to 128
- The last sliding window reliability can be smaller than the others (last beta prolog is not processed, tails are used to initialize beta states).
- The prolog of the sliding window before the last must point before the end of the frame, that is, $F = (Nsb \times Nsw - 1) \times R + P + r$ with $r>0$

A formula meeting the above properties is:

$$R = ceil\left(\frac{ceil(F/Nsw)}{Nsb}\right) \text{ with } Nsb = ceil\left(\frac{ceil(F/Nsw)}{128}\right)$$

9 Programming

The TCP requires setting up the following context per user-channel:

- standalone (SA) mode
  - 3 to 5 EDMA parameters (see Table 41)
  - the input configurations parameters
- shared-processing (SP) mode
  - 3 to 4 EDMA parameters (see Table 42)
  - the input configurations parameters

Note that several user channels can be programmed prior to starting the TCP.

Table 41. EDMA Parameters in Standalone (SA) Mode

Direction Transmit (DSP→TCP) Receive (TCP→DSP)	Data	Usage	Required/Optional
Transmit	Input configuration parameters	Send the input configuration parameters	Required
Transmit	Systematics and parities	Send the systematics and parities	Required
Transmit	Interleaver indexes	Send the interleaver table	Optional INTER bit
Receive	Hard decisions	Read the hard decisions	Required
Receive	Output parameters	Read the output parameters	Optional OUTF bit

Table 42. EDMA Parameters in Shared Processing (SP) Mode

Direction Transmit (DSP→TCP) Receive (TCP→DSP)	Data	Usage	Required/Optional
Transmit	Input configuration parameters	Send the input configuration parameters	Required
Transmit	Systematics and parities	Send the systematics and parities	Required
Transmit	A prioris	Send the a prioris	Required except for first iteration MAP1
Receive	Extrinsics	Read the extrinsics	Required

9.1 EDMA Resources

9.1.1 TCP Dedicated EDMA Resources

Within the available 64 EDMA channels event sources, two are assigned to the TCP: event 30 and event 31.

- Event 30 is associated to the TCP receive event (TCPREVT) and is used as the synchronization event for the EDMA transfer from the TCP to the DSP (receive).
- Event 31 is associated to the TCP transmit event (TCPXEVT) and is used as the synchronization event for the EDMA transfer from the DSP to the TCP (transmit).

9.1.2 Special TCP EDMA Programming Considerations

The EDMA parameters consists of six words as shown in Figure 26. All EDMA transfers, in the context of the TCP, must be done using 32-bit word elements, must contain an even number of words, and have source and destination addresses double-word aligned.

All EDMA transfers must be double-word aligned and the element count for the TCP EDMA transfer must be a multiple of 2. Single-word transfers that are not double-word aligned cause errors in TCP/VCP memory.

For more information, see the *TMS320C6000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide* (SPRU234).

Figure 26. EDMA Parameters Structure

(a) EDMA Registers

	31	0	EDMA parameter
Word 0	EDMA Channel Options Parameter (OPT)		OPT
Word 1	EDMA Channel Source Address (SRC)		SRC
Word 2	Array/frame count (FRMCNT)	Element count (ELECNT)	CNT
Word 3	EDMA Channel Destination Address (DST)		DST
Word 4	Array/frame index (FRMIDX)	Element index (ELEIDX)	IDX
Word 5	Element count reload (ELERLD)	Link address (LINK)	RLD

(b) EDMA Channel Options Parameter (OPT)

31–29	28–27	26	25–24	23	22–21	20	19–16
PRI	ESIZE	2DS	SUM	2DD	DUM	TCINT	TCC

15	14–13	12	11	10–5	4	3	2	1	0
—	TCCM	ATCINT	—	ATCC	—	PDTS	PDTD	LINK	FS

9.2 Programming Standalone (SA) Mode

Table 41 highlights the required EDMA resources to perform a standalone (SA) mode decoding. Each set of EDMA parameters uses the EDMA linking capabilities. Section 9.2.1 details the EDMA transfers programming and section 9.2.2 details the input parameters programming.

Any notification mechanism to flag that a user-channel has just been decoded is left to you. Suggested implementation is to use the EDMA interrupt generation capabilities (see the *TMS320C6000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide*, SPRU234) and program the EDMA to generate an interrupt after the user-channel's last TCPREVT synchronized EDMA transfer has completed.

9.2.1 EDMA Programming

9.2.1.1 Input Configuration Parameters Transfer

This EDMA transfer to the input configuration parameters is a 12-word TCPXEVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 01 (autoincrement)
  - DUM = 01 (autoincrement)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: user input configuration parameters start address (must be double-word aligned)
- ELEMENT COUNT: 000Ch
- ARRAY/FRAME COUNT: 0000h
- DESTINATION ADDRESS: TCPIC0 (5800 0000h)
- ELEMENT INDEX: don't care
- ARRAY/FRAME INDEX: don't care
- LINK ADDRESS: address in the EDMA PaRAM of the EDMA parameters associated with the systematics and parities
- ELEMENT COUNT RELOAD: don't care

Upon completion, this EDMA transfer is linked to the EDMA for systematics and parities transfer parameters.

9.2.1.2 Systematics and Parities Transfer

This EDMA transfer to the systematics and parities memory is a TCPXEVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 01 (autoincrement)
  - DUM = 00 (fixed)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: systematics and parities start address (must be double-word aligned)
- ELEMENT COUNT: $2 \times ceil\left(\frac{F \times R}{8 \times (FrameCount + 1)}\right)$
- ARRAY/FRAME COUNT: $ceil\left(\frac{F \times R}{(NWDSYPAR \times 4)}\right) - 1$
- DESTINATION ADDRESS: TCPSP (5802 0000h)
- ELEMENT INDEX: don't care
- ARRAY/FRAME INDEX: don't care
- LINK ADDRESS: see cases 1, 2, and 3 below
- ELEMENT COUNT RELOAD: don't care

Upon completion, this EDMA transfer is linked to one of the following:

1) The EDMA interleaver table transfer parameters, if there is a new one to be loaded in the TCP (INTER bit is set to 1)
2) The EDMA input configuration parameters transfer parameters of the next user-channel, if there is one ready to be decoded and no interleaver table to be loaded in the TCP (INTER bit is cleared to 0)
3) Null EDMA transfer parameters (with all zeros), if there is no more user-channel ready to be decoded and no interleaver table to be loaded in the TCP (INTER bit is cleared to 0)

9.2.1.3 Interleaver Indexes Transfer

This EDMA transfer to the interleaver memory is a TCPXEVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 01 (autoincrement)
  - DUM = 00 (fixed)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: interleaver table start address (must be double-word aligned)
- ELEMENT COUNT: $2 \times ceil\left(\frac{F}{4 \times (FrameCount + 1)}\right)$
- ARRAY/FRAME COUNT: $ceil\left(\frac{F}{(NWDINTER \times 2)}\right) - 1$
- DESTINATION ADDRESS: TCPINTER (5808 0000h)
- ELEMENT INDEX: don't care
- ARRAY/FRAME INDEX: don't care
- LINK ADDRESS: see cases 1 and 2 below
- ELEMENT COUNT RELOAD: don't care

Upon completion this EDMA transfer is linked to one of the following:

1) The EDMA input configuration parameters transfer parameters of the next user-channel, if there is one ready to be decoded
2) Null EDMA transfer parameters (with all zeros), if there is no more user-channel ready to be decoded

9.2.1.4 Hard-decisions Transfer

This EDMA transfer to the hard decisions buffer is a TCPREVT frame-synchronized transfer. The parameters should be set as:

- ❏ OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 00 (fixed)
  - DUM = 01 (autoincrement)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- ❏ SOURCE ADDRESS: TCPHD (580A 0000h)
- ❏ ELEMENT COUNT: $2 \times ceil\left(\frac{F}{64 \times (FrameCount + 1)}\right)$
- ❏ ARRAY/FRAME COUNT: $ceil\left(\frac{F}{(NWDHD \times 32)}\right) - 1$
- ❏ DESTINATION ADDRESS: user hard decisions start address (must be double-word aligned)
- ❏ ELEMENT INDEX: don't care
- ❏ ARRAY/FRAME INDEX: don't care
- ❏ LINK ADDRESS: see cases 1, 2, and 3 below
- ❏ ELEMENT COUNT RELOAD: don't care

Upon completion this EDMA transfer is linked to one of the following:

1) The EDMA hard decisions transfer parameters of the next user-channel, if there is one ready to be decoded and the OUTF bit is cleared to 0
2) The EDMA output parameters transfer parameters, if the OUTF bit is set to 1
3) Null EDMA transfer parameters (with all zeros), if there is no more user-channel ready to be decoded and the OUTF bit is cleared to 0

9.2.1.5 Output Parameters Transfer

This EDMA transfer is optional and depends on the OUTF bit in the TCP input configuration register 0 (TCPIC0). This EDMA transfer is a TCPREVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 01 (autoincrement)
  - DUM = 01 (autoincrement)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: TCPOUT (5800 0030h)
- ELEMENT COUNT: 0002h
- ARRAY/FRAME COUNT: 0000h
- DESTINATION ADDRESS: user output parameters start address (must be double-word aligned)
- ELEMENT INDEX: don't care
- ARRAY/FRAME INDEX: don't care
- LINK ADDRESS: see cases 1 and 2 below
- ELEMENT COUNT RELOAD: don't care

Upon completion, this EDMA transfer is linked to one of the following:

1) The hard decisions EDMA transfer parameters of the next user-channel, if there is one ready to be decoded
2) Null EDMA transfer parameters (with all zeros), if there is no more user-channel ready to be decoded

9.2.2 Input Configurations Parameters Programming

The frame length (FL bits in TCPIC0) should be set to the number of information bits (prior to turbo-encoding and not including the rate or the tail bits).

The number of subblocks (NSB bits in TCPIC2), the reliability length (R bits in TCPIC1), and the prolog size (P bits in TCPIC2) should be set as described in section 8.3. It should be noted that a 1 must be subtracted from the calculated R value prior to writing to TCPIC1.

The maximum number of iterations (MAXIT bits in TCPIC2) should be selected as a function of the overall system performance (typical values are 6 to 8). A value 0 sets the maximum number of iterations to its maximum (32).

The SNR threshold ratio (SNR bits in TCPIC2) should be selected as a function of the overall system performance. A value 0 disables the stopping criteria algorithm.

The TAIL1, TAIL2, TAIL3, TAIL4, TAIL5, TAIL6 bits should be programmed as described in sections 6.7 through 6.12, respectively

The remaining bit fields should be set as described in Table 43 (a bit not mentioned is a don't care).

Table 43. Input Configuration Parameters Settings in Standalone (SA) Mode

Bit field	Register	Value
OPMOD	TCPIC0	OPMOD=0
INTER	TCPIC0	INTER=0, if no new interleaver table is needed; otherwise, INTER=1
OUTF	TCPIC0	OUTF=1, if TCPREVT is to be generated for the output parameters load; otherwise, OUTF=0
NWDSYPAR	TCPIC3	Number of words per TCPXEVT event for the systematics and parities EDMA transfer
NWDINTER	TCPIC3	Number of words per TCPXEVT event for the interleaver indexes EDMA transfer
NWDHD	TCPIC5	Number of words per TCPREVT event for the hard decisions EDMA transfer

9.3 Programming Shared-Processing (SP) Mode

Shared-processing (SP) mode requires more CPU intervention as stated in section 5. The input frame must be split into several subframes as described in section 8.2. To decode the whole frame, follow these steps:

1) The CPU sets up the context for MAP1 (OPMOD bits in TCPIC0 set to 4h) for iteration 1.
2) The CPU starts the TCP that will output noninterleaved extrinsic data.
3) The CPU interleaves the generated extrinsic data.
4) The CPU sets up the context for MAP2 (OPMOD bits in TCPIC0 set to 7h) for iteration 1.
5) The CPU starts the TCP that will output interleaved extrinsic data.
6) The CPU deinterleaves the generated extrinsic data.
7) The CPU sets up the context for MAP1 (OPMOD bits in TCPIC0 set to 5h) for iteration 2.
8) The CPU starts the TCP that will output noninterleaved extrinsic data.
9) The CPU interleaves the generated extrinsic data.
10) The CPU sets up the context for MAP2 (OPMOD bits in TCPIC0 set to 7h) for iteration 2.
11) The CPU starts the TCP that will output interleaved extrinsic data.
12) The CPU deinterleaves the generated extrinsic data.
13) Repeat steps 7 to 12 until the desired number of iterations has been reached.
14) The CPU calculates the hard decision data.

Table 42 highlights the required EDMA resources to perform a shared-processing (SP) mode decoding. As in standalone (SA) mode decoding, each set of EDMA parameters uses the EDMA linking capabilities. In addition, the a priori data transfer is done using the EDMA alternate transfer chaining capabilities. Section 9.3.1 details the EDMA transfers programming and section 9.3.2 details the input parameters programming. It should be noted that any stopping criteria algorithm has to be implemented by the CPU.

Any notification mechanism to flag that a user-channel has just been decoded is left to you. Suggested implementation is to use the EDMA interrupt generation capabilities (see the *TMS320C6000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide*, SPRU234) and program the EDMA to generate an interrupt after the user-channel's last TCPREVT synchronized EDMA transfer has completed.

9.3.1 EDMA Programming

9.3.1.1 Input Configuration Parameters Transfer

This EDMA transfer to the input configuration parameters is a 12-word TCPXEVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 01 (autoincrement)
  - DUM = 01 (autoincrement)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: user input configuration parameters start address (must be double-word aligned)
- ELEMENT COUNT: 000Ch
- ARRAY/FRAME COUNT: 0000h
- DESTINATION ADDRESS: TCPIC0 (5800 0000h)
- ELEMENT INDEX: don’t care
- ARRAY/FRAME INDEX: don’t care
- LINK ADDRESS: address in the EDMA PaRAM of the EDMA parameters associated with the systematics and parities
- ELEMENT COUNT RELOAD: don’t care

Upon completion, this EDMA transfer is linked to the EDMA for systematics and parities transfer parameters.

9.3.1.2 Systematics and Parities Transfer

This EDMA transfer to the systematics and parities memory is a TCPXEVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 11 (indexed)
  - DUM = 00 (fixed)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)

- If the OPMOD bits in TCPIC0 are set to 5h or 7h, chaining needs to be enabled:
  - ATCINT = 1 (enabled)
  - ATCC = EDMA channel ID for a priori data transfer
  - CCER[ATCC] bit set to 1

- SOURCE ADDRESS: systematics and parities start address (for MAP1 or MAP2, must be double-word aligned) minus offset bytes (note that the start source address must then point to a valid memory location), with offset defined as (P is defined in section 8.3):
  - $P \times 8/4$ for rate 1/2 and 1/3
  - $P \times 20/8$ for rate 1/4
- ELEMENT COUNT (SFL is defined in section 6.2):
  - $2 \times ceil\left(\frac{SFL}{4}\right)$ for rate 1/2 and 1/3
  - $2 \times ceil\left(\frac{SFL \times 5}{16}\right)$ for rate 1/4
- ARRAY/FRAME COUNT: number of subframes – 1
- DESTINATION ADDRESS: TCPSP (5802 0000h)
- ELEMENT INDEX: 0004h
- ARRAY/FRAME INDEX (SFL is defined in section 6.2 and P is defined in section 8.3):
  - $(SFL - 2 \times P) \times 8/4$ for rate 1/2 and 1/3
  - $(SFL - 2 \times P) \times 20/8$ for rate 1/4
- LINK ADDRESS: see cases 1, 2, and 3 below
- ELEMENT COUNT RELOAD: don't care

Upon completion, this EDMA transfer is linked to one of the following:

1) The EDMA input configuration parameters transfer parameters of the next user-channel, if there is one ready to be decoded and the current decoding is a MAP1 from the 1st iteration
2) Null EDMA transfer parameters (with all zeros), if there is no more user-channel ready to be decoded

9.3.1.3 A priori Transfer

This EDMA transfer to the a priori memory is a TCPXEVT chained and frame-synchronized transfer. This EDMA transfer is chained from the systematic and parity data transfer and occurs only when executing any MAP but the MAP1 of the first iteration, that is, the OPMOD bits in TCPIC0 must be set to 5h or 7h. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 11 (indexed)
  - DUM = 00 (fixed)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: a priori start address (for MAP1 or MAP2, must be double-word aligned) minus prolog length bytes
- ELEMENT COUNT (SFL is defined in section 6.2):

  $$2 \times ceil\left(\frac{SFL}{8}\right)$$

- ARRAY/FRAME COUNT: number of subframes − 1
- DESTINATION ADDRESS: TCPAP (5806 0000h)
- ELEMENT INDEX: 0004h
- ARRAY/FRAME INDEX (SFL is defined in section 6.2 and P is defined in section 8.3):

  $$SFL - 2 \times P$$

- LINK ADDRESS: see cases 1 and 2 below
- ELEMENT COUNT RELOAD: don’t care

Upon completion, this EDMA transfer is linked to one of the following:

1) The EDMA input configuration parameters transfer parameters of the next user-channel MAP, if there is one ready to be decoded
2) Null EDMA transfer parameters (with all zeros), if there is no more user-channel LOG-MAP ready to be decoded

9.3.1.4 Extrinsics Transfer

This EDMA transfer to the extrinsics buffer is a TCPREVT frame-synchronized transfer. The parameters should be set as:

- OPTIONS:
  - ESIZE = 00 (element size is 32 bits)
  - 2DS = 2DD = 0 (1 dimensional)
  - SUM = 00 (fixed)
  - DUM = 01 (autoincrement)
  - LINK = 1 (linking of event parameters is enabled)
  - FS = 1 (channel is frame synchronized)
- SOURCE ADDRESS: TCPEXT (5804 0000h)
- ELEMENT COUNT (SFL is defined in section 6.2 and P is defined in section 8.3):

$$2 \times ceil\left(\frac{(SFL - 2 \times P)}{8}\right)$$

- ARRAY/FRAME COUNT: number of subframes – 1
- DESTINATION ADDRESS: user extrinsics start address (for data from MAP1 or MAP2, must be double-word aligned)
- ELEMENT INDEX: don't care
- ARRAY/FRAME INDEX: don't care
- LINK ADDRESS: see cases 1 and 2 below
- ELEMENT COUNT RELOAD: don't care

Upon completion, this EDMA transfer is linked to one of the following:

1) The EDMA extrinsics transfer parameters of the next user-channel, if there is one ready to be decoded
2) Null EDMA transfer parameters (all with all zeros), if there is no more user-channel ready to be decoded

9.3.2 Input Configurations Parameters Programming

The frame length (FL bits in TCPIC0) should be set to the total shared-processing frame length (prior to turbo-encoding and not including any tail information).

The actual subframe length (SFL) bits programmed in TCPIC1 are the number of information bits (prior to turbo-encoding) in a subframe (not the last) augmented by 2 times the prolog length. Defining SF_uL as being the number of useful symbols within any subframe but the last (this value corresponds to the number of extrinsics that are issued after decoding the subframe), SF_uL is given by $SF_uL = SFL - 2 \times P$. It should be noted that SF_uL must be a multiple of 8 for rate 1/2 and 1/3, and a multiple of 16 for rate 1/4.

The number of subblocks (NSB bits in TCPIC2), the reliability length (R bits in TCPIC1), and the prolog size (P bits in TCPIC2) should be set as described in section 8.3, taking SF_uL as a frame length. It should be noted that a 1 must be subtracted from the calculated R value prior to writing to TCPIC1. P must be a multiple of 8 for rate 1/2 and 1/3, and a multiple of 16 for rate 1/4.

The last subframe reliability length LSF_uL depends on the number of subframe and is defined as $LSF_uL = FL - (n - 1) \times SF_uL$, where n is the number of subframes.

The number of subblocks (LASTNSB bits in TCPIC3) and the reliability length (LASTR bits in TCPIC2) for the last subframe should be set as described in section 8.3, taking LSF_uL as a frame length. It should be noted that a 1 must be subtracted from the calculated LASTR value prior to writing to TCPIC2.

The remaining bit fields should be set as described in Table 44 (a bit not mentioned is a don't care).

Table 44. Input Configuration Parameters Settings in Shared-Processing (SP) Mode

Bit field	Register	Value
OPMOD	TCPIC0	OPMOD = 4 for the first LOG-MAP1 of the first iteration OPMOD = 5 for any other LOG-MAP1 OPMOD = 7 for any LOG-MAP2
INTER	TCPIC0	INTER = 0
OUTF	TCPIC0	OUTF = 0
NWDSYPAR	TCPIC3	Only one event allowed per subframe. Should be set to: ❑ *ceil* (*SFL*/2) for rate 1/2 and 1/3 ❑ *ceil* (*SFL*×5/8) for rate 1/4
NWDTEXT	TCPIC3	Only one event allowed per subframe. Should be set to *ceil* ((*SFL* – 2 × *P*)/4)
NWDAP	TCPIC5	Only one event allowed per subframe. Should be set to *ceil* (*SFL*/4)

10 Output Parameters

The NIT bits in the TCP output parameter register (TCPOUT) indicate the actual number of iterations run, providing useful information when the stopping criteria is enabled.

11 Events Generation

A TCP transmit event (TCPXEVT) and TCP receive event (TCPREVT) is generated as shown in Figure 27 for standalone (SA) mode and as shown in Figure 28 for shared-processing (SP) mode (example of 2 subframes).

Figure 27. TCP Events Generation in Standalone (SA) Mode

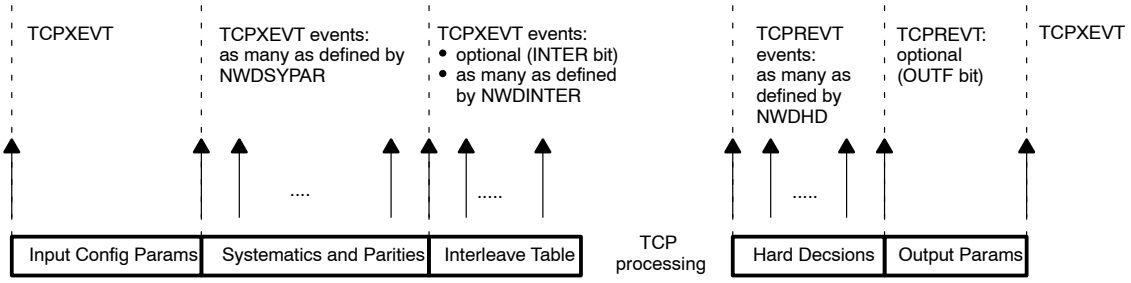
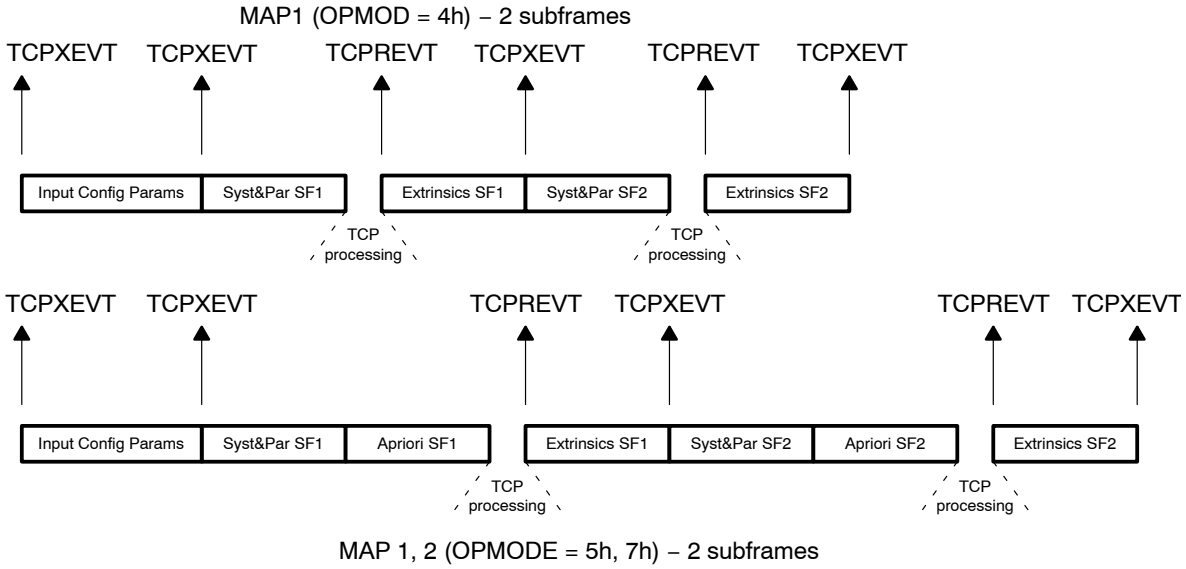


Figure 28. TCP Events Generation in Shared-Processing (SP) Mode (Example with 2 Subframes)



12 Starting, Pausing, and Resuming the TCP

To start the TCP, the START command must be written in the TCP execution register (TCPEXE). Writing a START stops any ongoing activity, generates a TCPXEVT, and the TCP then waits for input control parameters.

The pause feature allows the application to control the EDMA transfers associated with the TCP; it can be used when the EDMA resources are getting low. The system software can then choose to allocate EDMA bandwidth to higher priority tasks.

Pausing is performed by writing a 1 in the PAUSE bit of TCPEXE; unpausing is performed by writing a 1 in the UNPAUSE bit of TCPEXE.

The pause command acknowledgment is done by setting the PAUS bit in the TCP status register (TCPSTAT) to 1 (in case the PAUSE command occurs when sending the input configuration parameters, the PAUS bit is set to 1 only after the transfer is finished).

When paused during an EDMA transfer, this EDMA transfer runs to completion but no subsequent TCP event is generated, the TCPSTAT reflects the next state and waits to be unpaused. When paused during the MAP processing, the TCP stops within 4 cycles.

Unpausing the TCP causes any due event to be generated or the MAP processing to be resumed.

13 Errors and Status

13.1 Errors

The TCP error register (TCPERR) flags any errors that occurred in the TCP, the TCP stops, and the TCPINT interrupt is generated. TCPINT has an interrupt selector value of 31. See the *TMS320C6000 DSP Interrupt Selector Reference Guide* (SPRU646) for details on how to setup interrupts.

Reading TCPERR resets both TCPERR and the TCP status register (TCPSTAT) to their default values, that is, the TCP waits for a new START command.

13.1.1 Error Status: ERR

The ERR bit is set to 1 in case of error.

13.1.2 Unexpected Frame Length: F

The F bit is set to 1 if the programmed frame length is strictly smaller than 40 or is strictly greater than 20730.

13.1.3 Unexpected Prolog Length: P

The P bit is set to 1 if the specified prolog length is strictly greater than 48. Values smaller than 24 are ignored by the hardware and 24 is used.

13.1.4 Unexpected Code Rate: RATE

The RATE bit is set to 1 if the code rate differs from values 1/2, 1/3, and 1/4.

13.1.5 Unexpected Subframe Length: SF

The SF bit is set to 1 if the specified subframe length is strictly greater than 5114.

13.1.6 Unexpected Operational Mode: MODE

The MODE bit is set to 1 if the specified operational mode differs from values 4, 5, and 7.

13.1.7 Unexpected Reliability Length: R

The R bit is set to 1 if the specified reliability length minus 1 is strictly smaller than 40.

13.1.8 Unexpected Last Subframe Reliability Length: LR

The LR bit is set to 1 if the specified last subframe reliability length minus 1 is strictly smaller than 40.

13.1.9 Unexpected Interleaver Table Load: INT

The INT bit is set to 1 if loading an interleaver table has been requested in SP mode.

13.1.10 Unexpected Output Parameters Load: OP

The OP bit is set to 1 if loading the output parameters has been requested in SP mode.

13.1.11 Unexpected Memory Access: ACC

The ACC bit is set to 1 when an unexpected memory access occurs. This can be used to spot any EDMA programming issues. This can occur when:

- TCP in waiting for input configuration parameters state and memory access to any TCP memory but the interleaver memory is performed
- TCP in waiting for systematics and parities state and memory access to any TCP memory other than the TCPINTER memory
- TCP in waiting for a prioris state and memory access to any TCP memory other than the TCPAP memory
- TCP in waiting for extrinsics state and memory access to any TCP memory other than the TCPEXT memory
- TCP in waiting for hard decisions state and memory access to any TCP memory other than the TCPHD memory
- TCP in waiting for output parameters state and memory access to any TCP memory

13.2 Status

The TCP status register (TCPSTAT) reflects the state of the TCP. The ERR, RUN, and PAUS bits are exclusive.

13.2.1 TCP Paused: PAUS

The PAUS bit is set to 1 if the TCP has been paused. To clear the PAUS bit, an UNPAUSE command must be written in the TCP execution register (TCPEXE).

13.2.2 MAP Decoding Running: RUN

The RUN bit is set to 1 when the MAP has started executing. The RUN bit is reset when the programmed decoding is finished.

13.2.3 TCP Stopped Due to Error: ERR

The ERR bit is set to 1 if the TCP has encountered an error. The ERR bit is reset by writing a new START command in the TCP execution register (TCPEXE).

13.2.4 TCP Waiting for Input Control Parameters Write: WIC

The WIC bit is set to 1 when the TCP is waiting for the input configurations parameters to be written. After the very first decoding is finished, an XEVT is generated to allow any ready user-channel to be decoded and the WIC bit is set to 1.

13.2.5 TCP Waiting for Interleaver Table Write: WINT

The WINT bit is set to 1 when the TCP is waiting for the interleaver table to be written.

13.2.6 TCP Waiting for Systematics and Parities Write: WSP

The WSP bit is set to 1 when the TCP is waiting for the systematics and parities to be written.

13.2.7 TCP Waiting for A prioris Write: WAP

The WAP bit is set to 1 when the TCP is waiting for the a prioris to be written.

13.2.8 TCP Waiting for Extrinsics Read: REXT

The REXT bit is set to 1 when the TCP is waiting for the extrinsics to be read.

13.2.9 TCP Waiting for Hard-Decisions Read: RHD

The RHD bit is set to 1 when the TCP is waiting for the hard decisions to be read.

13.2.10 TCP Waiting for Output Parameters Read: ROP

The ROP bit is set to 1 when the TCP is waiting for the output parameters to be read.

14 Performance

14.1 Processing Delay

The processing blocks shown in Figure 23, page 49, (beta, alpha, and extrinsic blocks) have a latency of 4 TCP clock cycles (the TCP is running at a frequency of CPU clock divided by 2). Because each cycle is independent, this allows each block to process 4 sliding windows in parallel.

The sliding windows processing is being pipelined in each of the blocks (example of beta block is given in Table 45), so the performance of the MAP unit depends on the number of sliding windows per subblock:

- 4 sliding windows per subblock ⇒ 4 TCP cycles are required to process 4 symbols.
- 2 sliding windows per subblock ⇒ 4 TCP cycles are required to process 2 symbols.
- 1 sliding windows per subblock ⇒ 4 TCP cycles are required to process 1 symbols.

Table 45. Cycle Number versus Sliding Window Processing for the Beta Block

Cycle Number	Step 1	Step 2	Step3	Step 4
0	sw1(n−1)†			
1	sw2(n−1)	sw1(n−1)		
2	sw3(n−1)	sw2(n−1)	sw1(n−1)	
3	sw4(n−1)	sw3(n−1)	sw2(n−1)	sw1(n−1)
4	sw1(n−2)	sw4(n−1)	sw3(n−1)	sw2(n−1)
5	sw2(n−2)	sw2(n−2)	sw4(n−1)	sw3(n−1)
6	sw3(n−2)	sw3(n−2)	sw2(n−2)	sw4(n−1)
...	...	...	...	...
4(n−1)−3	sw1(0)			
4(n−1)−2	sw2(0)	sw1(0)		
4(n−1)−1	sw3(0)	sw2(0)	sw1(0)	
4(n−1)	sw4(0)	sw3(0)	sw2(0)	sw1(0)
4n−3		sw4(0)	sw3(0)	sw2(0)
4n−2			sw4(0)	sw3(0)
4n−1				sw4(0)

† sw1 (n−1) is the first sliding window of the subblock n−1.

The processing sequence is the following:

1) Prolog Beta (subblock n) – *not storing beta during this step, just initialize states for the reliability portion*

2) Beta (subblock n)

3) Prolog Alpha (subblock n) || Prolog Beta (subblock n + 1)

4) Alpha + Extrinsic processing (subblock n) || Beta (subblock n + 1)

Repeat steps 3 and 4 for all subblocks.

Given P (prolog length), R (reliability length), Nsb (number of subblocks), the cycle count (in TCP clock cycles) per step are:

1) $4 \times P + ControlLogicCycles \approx 6 \times P$

2) $4 \times R + ControlLogicCycles \approx 4 \times R + 2 \times P$

3) $(4 \times P + ControlLogicCycles) \times Nsb \approx 6 \times P \times Nsb$

4) $(4 \times R + ControlLogicCycles) \times Nsb \approx (4 \times R + 2 \times P) \times Nsb$

The total cycle count is estimated to: $\approx (4 \times R + 8 \times P) \times (Nsb + 1)$ for a single MAP decode on $Nsb \times Nsw \times R$ symbols.

Figure 29 shows the number of TCP cycles per symbol in the best case scenario (P = 24) and Figure 30 the worst case scenario (P = 48). The number of subblocks were calculated as described in section 8.3.

EDMA transfer cycles should also be added to the total number of processing cycles for an estimate of the overall processing delay, they depend greatly on system partitioning and loading. The EDMA transfers are at a maximum rate of 64 bits at a frequency of CPU clock divided by 4. It should be noted that the TCP starts processing as soon as all systematics and parities have been transferred.

Figure 29. Processing Unit Performance for P = 24

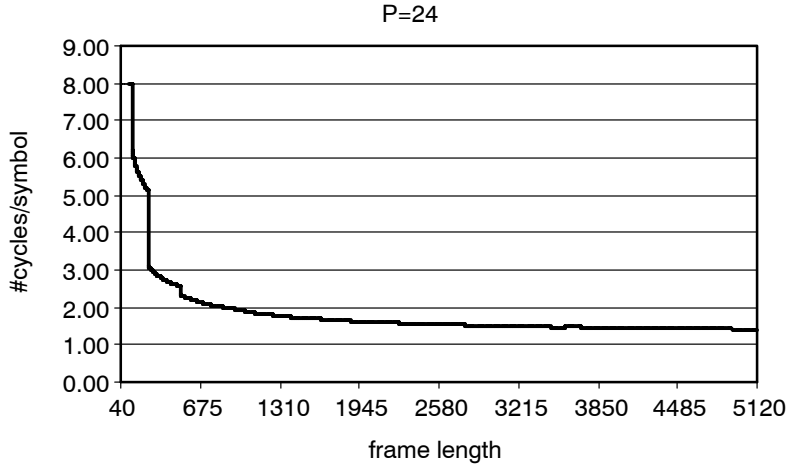
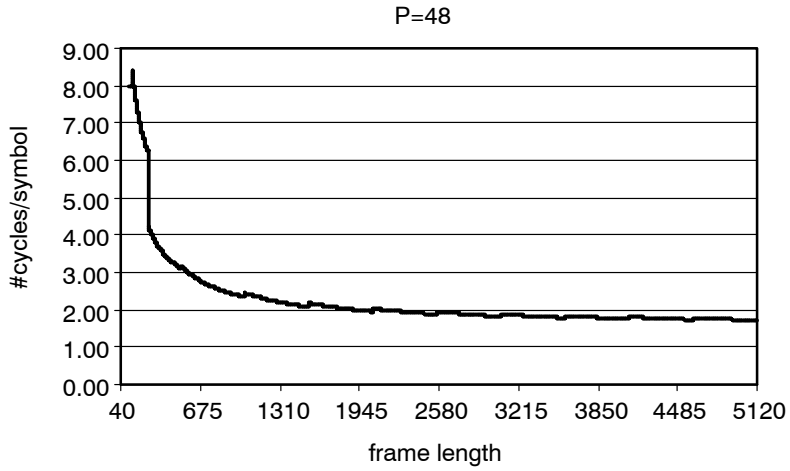


Figure 30. Processing Unit Performance for P = 48



14.2 Bit Error Rate

All curves were generated using AWGN channel data.

Figure 31 shows the BER performance of the TCP after 8 iterations for various frame sizes. Figure 32 shows the influence of the prolog size on the BER performance. Figure 33 and Figure 34 show the influence of the stopping criteria algorithm with different SNR threshold on the BER performance and processing delay.

Figure 31. BER Performance of TCP After 8 Iterations

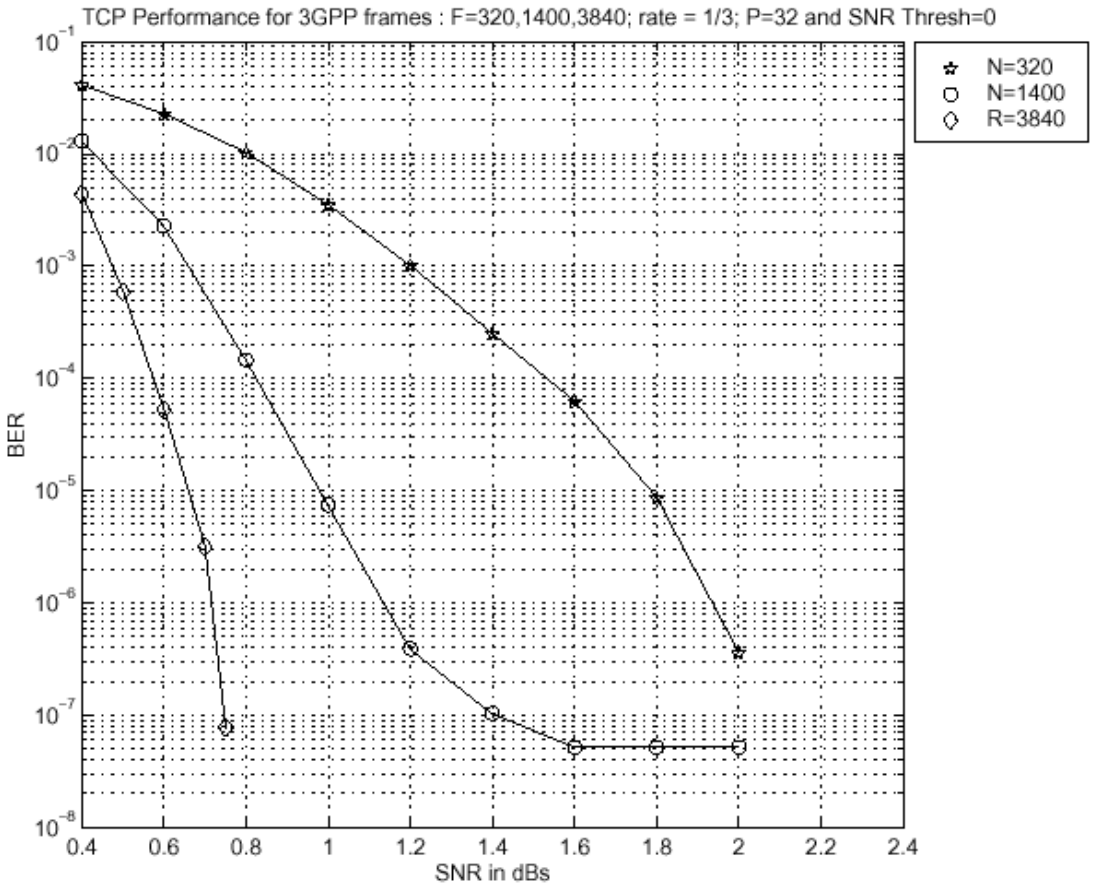


Figure 32. Prolog Size Influence on BER Performance

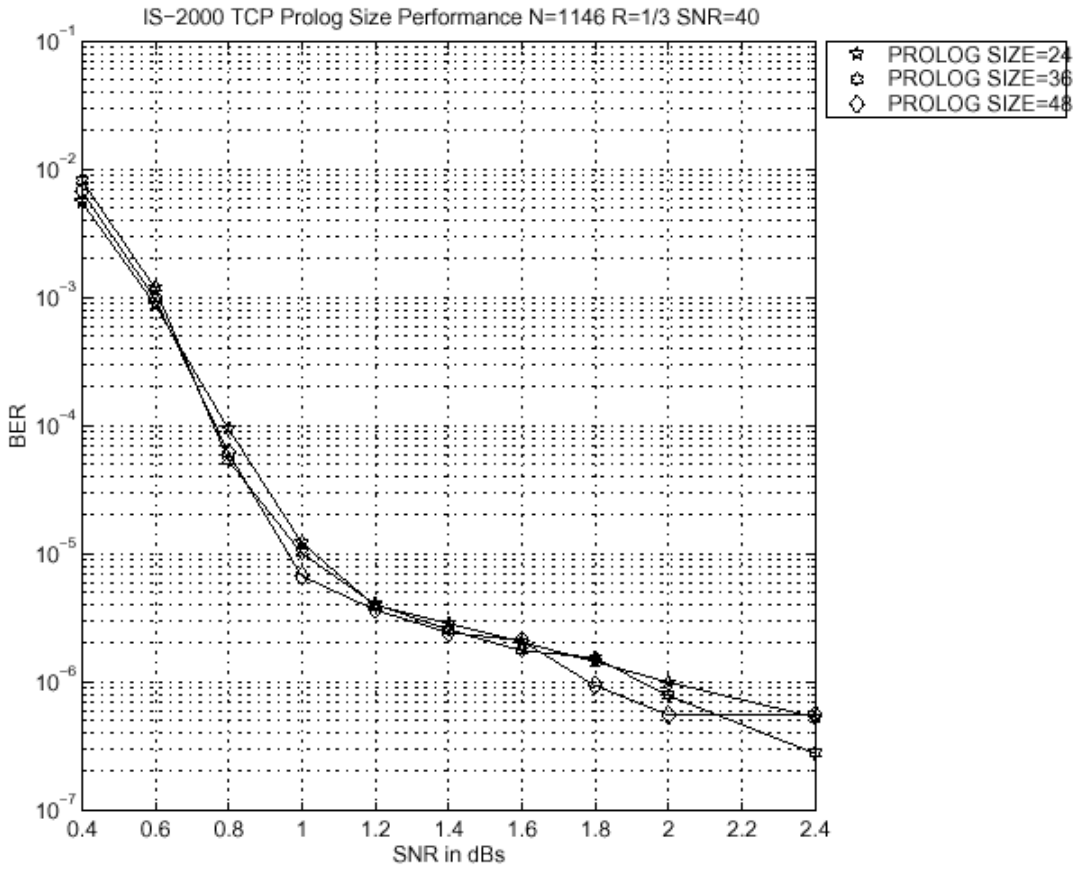


Figure 33. SNR Threshold Influence on BER Performance

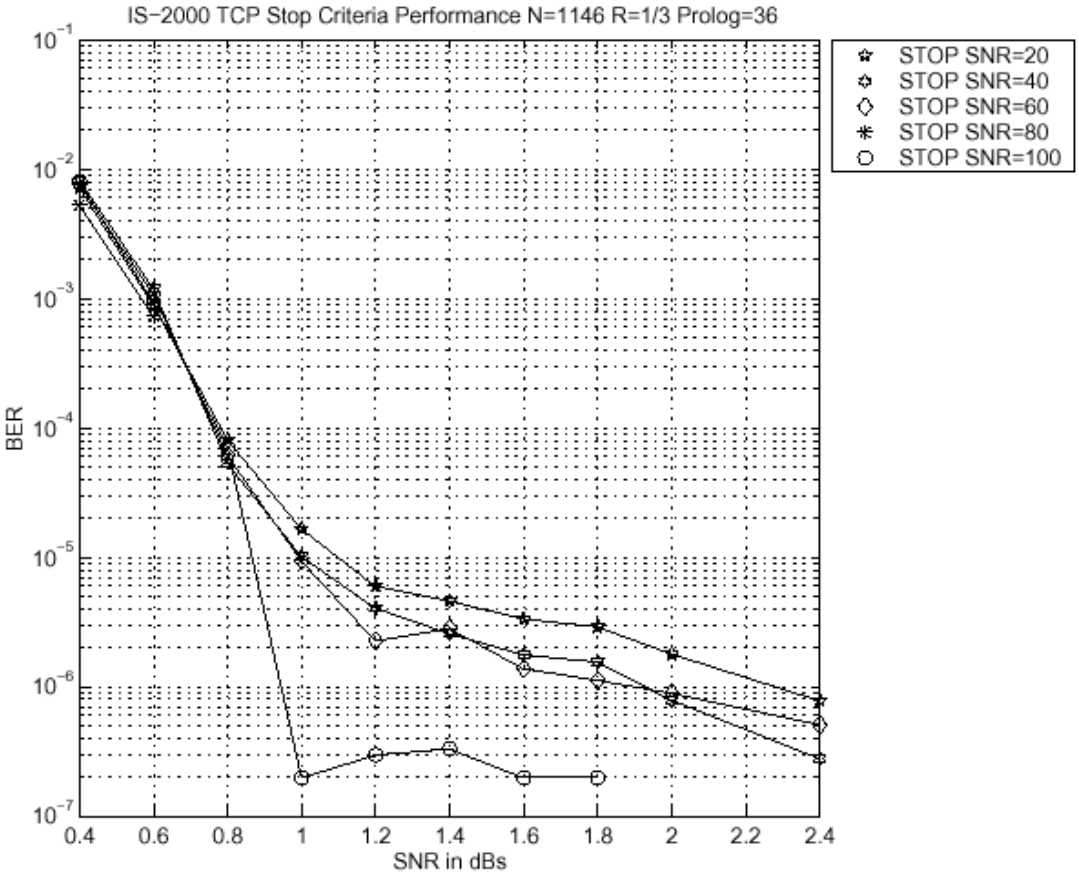
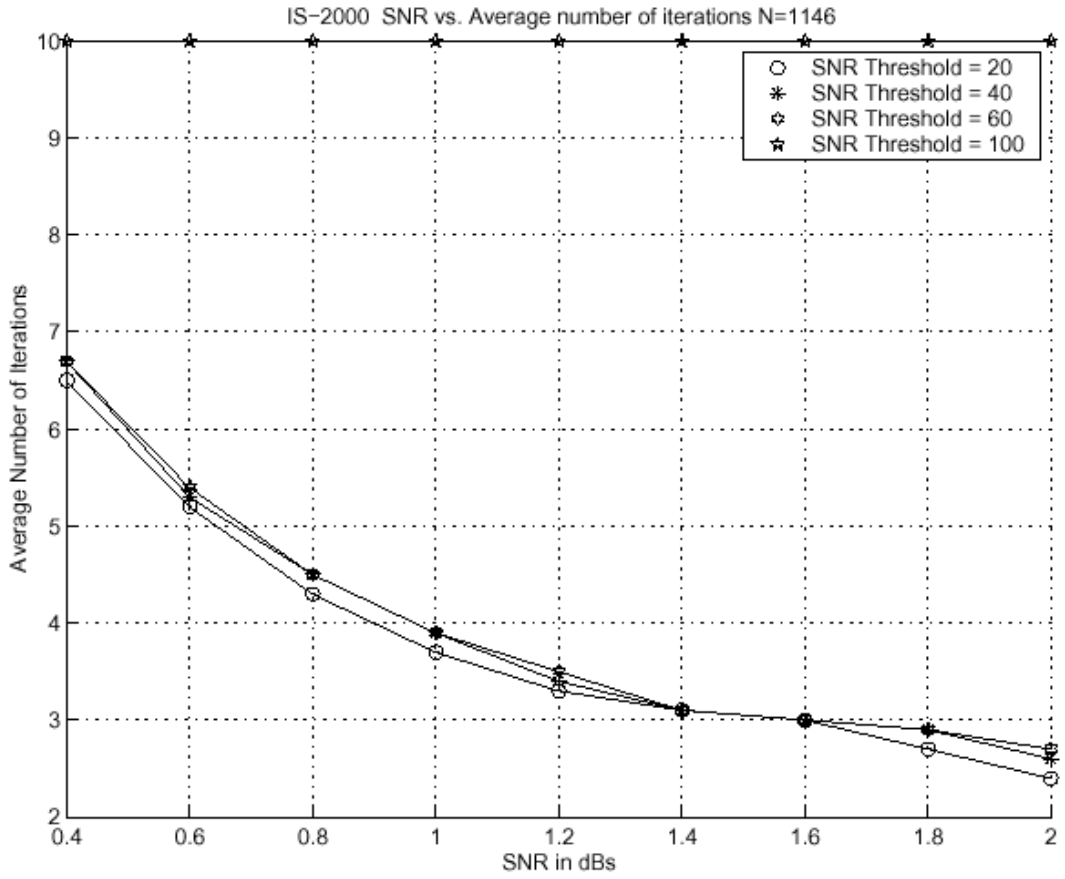


Figure 34. SNR Threshold Influence on Processing Delay



This page is intentionally left blank.

Revision History

Table 46 lists the changes made since the previous version of this document.

Table 46. Document Revision History

Page	Additions/Modifications/Deletions
24-43	Updated symbolic values (*symval*) of the bits in Table 14 through Table 30.

This page is intentionally left blank.

Index

N

O

P

R

S

T

U

W