CDMA Reference Blockset

For Use with Simulink®

- Modeling
- Simulation
- Implementation

User's Guide

Version 1

How to Contact The MathWorks:

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup

`support@mathworks.com`	Technical support
`suggest@mathworks.com`	Product enhancement suggestions
`bugs@mathworks.com`	Bug reports
`doc@mathworks.com`	Documentation error reports
`service@mathworks.com`	Order status, license renewals, passcodes
`info@mathworks.com`	Sales, pricing, and general information

508-647-7000 Phone

508-647-7001 Fax

The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098 Mail

For contact information about worldwide offices, see the MathWorks Web site.

CDMA Reference Blockset User's Guide

© COPYRIGHT 2000-2004 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and TargetBox is a trademark of The MathWorks, Inc.

Other product or brand names are trademarks or registered trademarks of their respective holders.

Printing History:	January 2000	Online only	New for Version 1.0.1 (Release 11.1)
	September 2000	First printing	Revised for Version 1.0.2 (Release 12)
	June 2001	Online only	Revised for Version 1.1 (Release 12.1)
	June 2004	Online only	Revised for Version 1.1 (Release 14)
	October 2004	Online only	Revised for Version 1.1 (Release 14SP1)

Contents

1 Introduction

2 Tutorial

1

Introduction

What Is the CDMA Reference Blockset?

The CDMA Reference Blockset is a collection of Simulink® blocks designed to help you develop and simulate CDMA wireless communication systems, based on the current North American IS-95A CDMA (code division multiple access) standard.

With the CDMA Reference Blockset, you can construct block diagram models of wireless systems quickly and easily using click-and-drag mouse operations. You can then run simulations on those models.You can also change parameters. The blocks in the CDMA Reference Blockset encompass the complete functionality required by the IS-95A standard.

Note The CDMA Reference Blockset is not designed to work with Real-Time Workshop®.

Using the CDMA Reference Blockset

You can use the CDMA Reference Blockset to develop end-to-end (transmitter-to-receiver) simulation models for forward and reverse communication links. Furthermore, you can reach the simulation stage quickly and focus on the specific components of greatest interest in the design.

The blocks support various modes, rates and types of channels. Some of the important features of the blocks are:

- **Multirate functionality:** The blocks support operations at all the rates supported by the standard. In operations requiring variable rates (that is, when the data rate may change on the fly), a block operates on an input frame according to the frame's rate input.
- **Rate set support:** The blocks support both the Rate Set I and Rate Set II specified in the J-STD-008 standard. This gives the blockset the capability to model the operation for some of the IS-95B capabilities.
- **Sync, Paging, and Traffic channel support for the forward link:** Each forward link block supports operation in the modes corresponding to these three channel types.

- **Access and Traffic channel support for the reverse link:** Each reverse link block can be used to simulate the Access channel, as well as the reverse Traffic channel.

You can also use components of the CDMA Reference Blockset as an architectural basis to create sophisticated subsystems. The Blockset provides a simulation platform on which you can develop further customized designs or proprietary algorithms for optimizing receiver performance. You can modify existing blocks or develop custom-designed blocks to design models and carry out simulations according to your own needs.

The CDMA Reference Blockset also includes several examples of models of wireless systems, which will help you understand how to use the Blockset. After studying these examples, you will be able to construct your own models, based on the IS-95A standard, using any configurations you require.

Related Products

The CDMA Reference Blockset requires MATLAB®, as well as the additional MathWorks products listed in the table below.

For more information about any of these products, see either:

- The online documentation for that product, if it is installed or if you are reading the documentation from the CD
- The MathWorks Web site, at `http://www.mathworks.com`. See the "products" section.

Note The toolboxes listed below include functions that extend MATLAB's capabilities. The blocksets include blocks that extend Simulink's capabilities.

Product	Description
Communications Blockset	Simulink block libraries for modeling the physical layer of communications systems
Communications Toolbox	MATLAB functions for modeling the physical layer of communications systems
Signal Processing Blockset	Simulink block libraries for the design, simulation, and prototyping of digital signal processing systems
Signal Processing Toolbox	Tool for algorithm development, signal and linear system analysis, and time-series data modeling
Simulink	Interactive, graphical environment for modeling, simulating, and prototyping dynamic systems

Using This Guide

This guide is intended to help you get started with the CDMA Reference Blockset and to provide reference information after you have begun to build your own simulations.

Expected Background

The CDMA Reference Blockset is intended for users interested in developing and simulating models and designs for the CDMA system, and in experimenting with signal processing and receiver algorithms for research and development activities. This guide assumes that you have a background in communications engineering. It also assumes that either you are familiar with the CDMA IS-95A standard or you can refer to other sources to learn about the IS-95A standard while using this document to learn about the CDMA Reference Blockset.

Organization of the Document

Chapter 1, “Tutorial” acquaints you with the CDMA Reference Blockset. It includes:

- A brief overview of wireless technology in general, and CDMA systems in particular
- The contents of the CDMA Reference Blockset
- Descriptions of important conventions the blockset uses
- Demo models as a way of helping you learn how to use the blockset to simulate CDMA systems
- Lists of reference sources that provide background information

Chapter 2, “Block Reference” contains block summary tables listed by library, and descriptive reference pages for the blocks in the CDMA Reference Blockset. Each reference page describes the inputs, outputs, parameters, and behavior of the given block. The reference pages are organized alphabetically by block name.

Online Help

You can access HTML reference documentation for blocks in the CDMA Reference Blockset in several ways:

- Press the **Help** button in the dialog box for any block.
- On Windows platforms, right-click on a block name in the **Simulink Library Browser** window and select the **Get help on...** option.
- Select **Help Desk** from the MATLAB **Help** menu, or type `helpdesk` or `doc` at the command line, to access the Help Desk facility.

Technical Conventions

This guide, especially the section “Demo Models” in Chapter 1, distinguishes between blocks that are part of the CDMA Reference Blockset and other types of blocks. Other types of blocks include built-in Simulink blocks, blocks from other blocksets, and custom subsystems that exist in the demo model, but not in any library.

The term *library block* refers to a block that is part of the CDMA Reference Blockset. The term *subsystem* refers to a block that represents a collection of connected blocks. If a subsystem has no mask, then double-clicking on the icon opens the subsystem. If a subsystem is a *masked subsystem*, then double-clicking on the icon opens the dialog box for the subsystem. To open a masked subsystem to see the blocks that it contains, select the icon and select **Look under mask** from the model window’s **Edit** menu.

Typographical Conventions

This manual uses some or all of these conventions.

Item	Convention Used	Example
Example code	`Monospace` font	To assign the value 5 to `A`, enter `A = 5`
Function names/syntax	`Monospace` font	The `cos` function finds the cosine of each array element. Syntax line example is `MLGetVar ML_var_name`
Keys	**Boldface** with an initial capital letter	Press the **Return** key.
Literal strings (in syntax descriptions in reference chapters)	**`Monospace bold`** for literals	`f = freqspace(n,'whole')`
Mathematical expressions	*Italics* for variables Standard text font for functions, operators, and constants	This vector represents the polynomial $p = x^2 + 2x + 3$

MATLAB output	`Monospace` font	MATLAB responds with
		`A =`
		`5`

Menu titles, menu items, dialog boxes, and controls	**Boldface** with an initial capital letter	Choose the **File** menu.
New terms	*Italics*	An *array* is an ordered collection of information.

Item	Convention Used	Example
Omitted input arguments	(...) ellipsis denotes all of the input/output arguments from preceding syntaxes.	`[c,ia,ib] = union(...)`
String variables (from a finite list)	*`Monospace italics`*	`sysc = d2c(sysd,'`*`method`*`')`

1

Tutorial

Wireless Communication and CDMA

This section provides some background information on wireless technology, wireless personal communications, and the basics of CDMA technology. If you are familiar with CDMA technology and the IS-95A standard, you can skip to these later sections:

- "Main Library Structure" on page 1-12, which describes the components of the CDMA Reference Blockset
- "Hints for Using the CDMA Reference Blockset" on page 1-19, which describes some important conventions in the CDMA Reference Blockset that you should keep in mind while studying or creating simulations
- "Demo Models" on page 1-22, which uses demonstration models to illustrate how to use the blockset for simulating CDMA techniques.

Growth of Wireless Communications Services

The wireless communications industry has grown extremely rapidly over the past decade, with markets doubling roughly every 2 years. Much of this growth has been due to the public's increasing demand for mobile telephones, and more recently, wireless data systems. The increasing number of wireless users has spurred communications engineers to improve the quality of service and to use the available spectrum resources more efficiently.

Types of Wireless Communications Services

Consumers today have access to many different kinds of wireless products and services, having widely varying range of coverage. Short-range products include cordless telephones and wireless local area networks (WLANs). Long-range products include satellite-based wireless units. At various intermediate ranges are products such as cellular telephones, wide-area wireless data and radio paging services, specialized satellite-based message services in the freight industry, and internet access from hand-held units. Clearly, wireless communications today encompasses many technologies, systems and services, aimed at many different applications.

Wireless technologies and systems can be divided into six distinct groups:

- Cellular mobile radio systems
- Cordless telephones

- Wide-area wireless data systems
- High-speed WLANs
- Paging/messaging systems
- Satellite-based mobile systems

Of these, the most widely used are cellular mobile radio systems

Cellular Mobile Radio Systems

Cellular mobile radio systems aim to provide high-mobility, wide-ranging, two-way wireless voice communications. These systems accomplish their task by integrating wireless access with large-scale networks, capable of managing mobile users. Cellular radio technology generally uses transmitter power at a level around 100 times that used by a cordless telephone (approximately 2 W for cellular).

Standards

Cellular radio has evolved into digital radio technologies, using the systems standards of GSM (at 900 and 1800 MHz) in Europe, PDC in Japan, and IS-136A and IS-95A in the United States. Third-generation systems, such as wideband code division multiple access (WCDMA) and cdma2000, are currently under development.

Design Considerations

One of the most significant consideration in designing digital systems is the high cost of cell sites. This has motivated system designers to try to maximize the number of users per megahertz, and users per cell site.

Another important consideration is maintaining adequate coverage in areas of varying terrain and population density. For example, in order to cover sparsely populated regions, system designers have retained the high-power transmission requirement to provide maximum range from antenna locations.

Communications engineers have also been developing very small coverage areas, or microcells. Microcells provide increased capacity in areas of high user density, as well as improved coverage of shadowed areas. Some microcell base stations are installed in places of high user concentrations, such as conference center lobbies.

Overview of CDMA

Code division multiple access (CDMA) is a modulation and multiple-access scheme based on spread-spectrum communication. In this scheme, multiple users share the same frequency band at the same time, by spreading the spectrum of their transmitted signals, so that each user's signal is pseudo-orthogonal to the signals of the other users.

CDMA Signals

In a CDMA system, each signal consists of a different pseudorandom binary sequence (called the spreading code) that modulates a carrier, spreading the spectrum of the waveform. A large number of CDMA signals share the same frequency spectrum. If CDMA is viewed in either the frequency or time domain, the multiple access signals overlap with each other. However, the use of statistically orthogonal spreading codes separates the various signals in the code space.

CDMA Receivers

A CDMA receiver separates the signals by means of a correlator that uses the particular binary sequence to despread the signal and collect the energy of the desired signal. Other users' signals, whose spreading codes do not match this sequence, are not despread in bandwidth and, as a result, contribute only to the noise. These signals represent a self-interference generated by the system. The output of the correlator is sent to a narrow-bandwidth filter. The filter allows all of the desired signal's energy to pass through, but reduces the interfering signal's energy by the ratio of the bandwidth before the correlator to the bandwidth after the correlator. This reduction greatly improves the signal-to-interference ratio of the desired signal. This ratio is also known as the processing gain. The signal-to-noise ratio is determined by the ratio of the desired signal power to the sum of all of the other signal powers. It is enhanced by the processing gain or the ratio of spread bandwidth to baseband data rate.

CDMA Channel Assignments

A CDMA digital cellular waveform design uses a pseudorandom noise (PN) sequence to spread the spectrum. The sample rate of the spreading sequence (called the chip rate) is chosen so that the bandwidth of the filtered signal is several times the bandwidth of the original signal.

A typical system might use multiple PN sequences. In addition, it might use repeated spreading codes of known lengths to ensure orthogonality between

signals intended for different users. The channel assignment is essentially determined by the set of codes that are used for that particular link. Thus, the signal transmitted at any time in a logical channel is determined by:

- The frequency of operation for the base station
- The current symbol
- The specific orthogonal spreading code assigned for the logical channel
- The PN spreading code

CDMA Signal Processing

In the demodulation of CDMA signals, the different paths may be independently received, which greatly reduces the severity of the multipath fading. However, multipath fading is not completely eliminated because occasionally there may be multiple paths that cannot be independently processed by the demodulator.

Different users in CDMA employ signals that have very small cross-correlation. Thus, correlators can extract individual signals from a mixture of signals even though they are transmitted simultaneously in the same frequency band. CDMA systems employ wideband signals with good cross-correlation properties, which means the output of a filter matched to one user's signal is small when it receives a different user's signal as input.

In direct-sequence spread-spectrum systems, a high-rate antipodal pseudorandom spreading sequence modulates the transmitted signal so that the bandwidth of the resulting signal is roughly equal to the rate of the spreading sequence. The cross-correlation of the signals is then largely determined by the cross-correlation properties of the spreading signals. Although CDMA signals overlap in both time and frequency domains, they can be separated, based on their spreading waveforms.

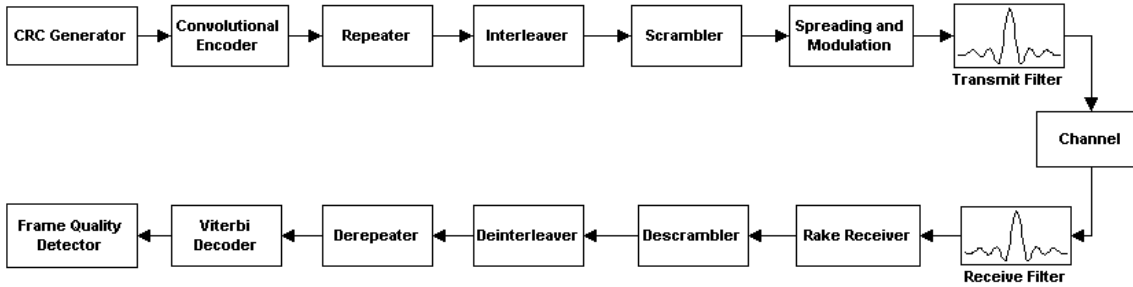
Spreading rates can be chosen to exceed the coherence bandwidth so that the channel becomes frequency selective. For instance, different spectral components are affected unequally by the channel, and only parts of the signal are affected by fades. Expressing the same observation in time domain terms, multipath components are resolvable at a resolution equal to the chip period and can be combined coherently, for example, by means of a rake receiver. Coherent combination of multipath components requires an estimate of the channel impulse response. Such an estimate can be calculated from a training sequence or by means of a pilot signal.

Fundamentals of the IS-95A CDMA System

This section describes the important features of the IS-95A CDMA system specifications. The CDMA Reference Blockset is designed to help you develop models to simulate different components of the IS-95A CDMA system.

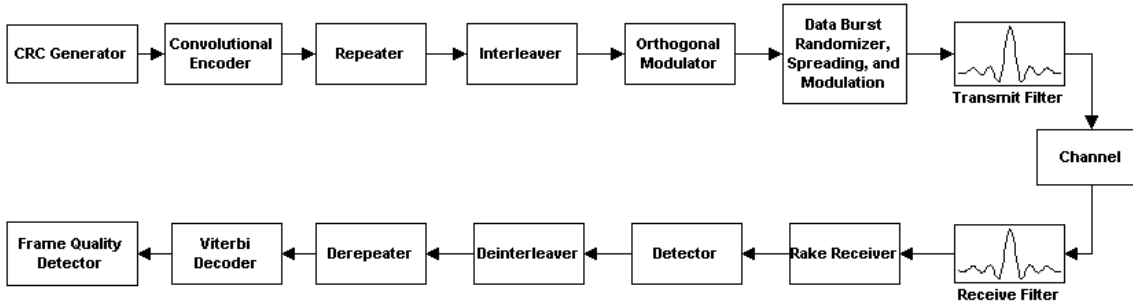
Channel Schematics

The following figure illustrates an IS-95A forward channel. The transmitter section includes channel coding, modulation and spreading, and filtering. The receiver section includes filtering, despreading and demodulation, and channel decoding.



IS-95A Forward Channel Diagram

The following figure illustrates an IS-95A reverse channel. It includes many of the same operations that are in the forward channel, but the functionalities of the blocks correspond to the reverse channel specifications.



IS-95A Reverse Channel Diagram

Forward and Reverse Channels

The primary features of the forward and reverse channels are described below in these sections:

- “Channel Assignment in IS-95A”
- “Forward Channel Coding”
- “Base Station Modulation and Spreading”
- “Base Station Transmitter Interface”
- “Coherent Rake Receiver”
- “Reverse Channel Coding”
- “Mobile Station Walsh Modulation and Spreading”
- “Noncoherent Rake Receiver”

Channel Assignment in IS-95A

The CDMA system requires spreading of the spectrum using a PN sequence. In IS-95A, the rate of this PN sequence (called the chip rate) is 1.2288 Mchips/s. This causes the resulting bandwidth of the spread signals to be about 1.25MHz, which is about one-tenth of the total bandwidth allocated to one cellular service carrier.

The IS-95A system uses two PN codes:

- The short PN code is a pair of periodic binary PN sequences with a period of 2^{15}. These sequences are used for spreading and despreading signals into in-phase and quadrature components. The same short PN code is used by multiple base stations using the same frequency band by using different timing offsets in the code cycle.
- The long PN code is a sequence with a period of 2^{42}-1, and this is used for spreading on the reverse link, as well as for data scrambling and power control burst randomization.

In addition to the PN codes, there is a set of length-64 mutually orthogonal codes called the Walsh code, which is used for ensuring orthogonality between the signals for different users receiving from the same base station. The Walsh code is also used for modulation for the reverse channel of IS-95A. Thus, the logical channel on the forward link is determined by the short PN code offset, the Walsh code assigned, and the assigned frequency of operation. On the

reverse link, the logical channel is determined by the short PN code offset, the long code offset, and the assigned frequency of operation.

The IS-95A forward link uses several logical channels:

- The Pilot channel modulates a constant symbol and is used for channel estimation, which allows for coherent demodulation of the other channels that carry information bits.
- The Sync channel is used for providing synchronization and configuration information to the mobile stations.
- The Paging channels are used for control information and sending paging messages to the mobile station.
- The Traffic channel carries the speech or data.

Similarly, the reverse link has the Traffic and Access logical channels:

- The Access channel is meant for control information, and is used for originating requests, responding to paging and other messages, or providing other data to the base station.
- The Traffic channel carries the speech or data. The Traffic channel supports variable data rate operation. According to the J-STD-008 requirement, there are two sets of Traffic channel data rates. Rate Set I has a maximum data rate of 9.6 kbps, whereas Rate Set II has a maximum data rate of 14.4 kbps. Both rate sets support full, half, quarter, and eighth rates with respect to the maximum data rate.

Forward Channel Coding

The IS-95A system uses the forward channels Sync, Paging, and Traffic to carry information from the base station to the mobile units. The channel coding operations in the forward CDMA operations use 20 ms frames for all channels, except for the Sync channel, which is coded using 26.666 ms frames. For error protection, a 1/2-rate convolutional code is used for all information channels. The receiver can use the Viterbi algorithm for optimal decoding of the encoded data.

Because the system uses variable data rates, the number of bits generated by the vocoder in one frame changes depending on the voice activity. To ensure that the symbol rate at the modulation stage is kept constant, the symbols are repeated for the lower data rate frames. For protection against bursts of errors,

the frame data is interleaved prior to modulation. These error protection measures improve the overall bit error rate (BER) on the link.

Base Station Modulation and Spreading

The IS-95A forward CDMA channel consists of the Pilot channel for coherent demodulation and the information channels. These channels are spread orthogonally by using a set of codes of length 64 called the Walsh codes. The combined signal is spread in quadrature by a pair of PN sequences with a fixed spreading rate.

The base station transmitter performs the encoding, the repetition, the interleaving, and the scrambling prior to spreading. The generated modulation symbols also need to carry power control commands to correct the mobile station transmit power. For this purpose, some of the symbols are replaced with command bits known as power control bits. The locations of these bits are randomized using a scheme based on a decimated long PN code. The details of these are in the IS-95A standard. The spread signal is filtered at baseband before transmission.

Base Station Transmitter Interface

The base station transmitter interface combines the various channels in the CDMA forward channel. The forward channel contains the Pilot channel, the Sync channel, the Paging channels, and multiple Traffic channels. Each Traffic channel has a unique Walsh code assigned to it. These Walsh codes are orthogonal to each other, and the different symbol streams are spread by their respective Walsh code (in bipolar form) and added together in a manner such that the weight given to each corresponds to the intended power in that channel. The Traffic channel has a variable data rate. In the case of lower data rate frames, the symbols are repeated and the Traffic channel power is reduced by the same repetition factor. This reduction ensures that the power transmitted for each information data bit (before coding and repetition) is the same.

Coherent Rake Receiver

The coherent rake receiver demodulates the desired channel in the input signal by despreading it with the corresponding Walsh code and the short PN code. The mobile user receives the signal transmitted from the serving CDMA base station through several paths with different propagation delays. The received signal, in addition to being corrupted by noise, is also distorted by the channel

fading. For a basic receiver design, the delay-spread results in a loss of performance.

The rake receiver, on the other hand, uses the direct-sequence spreading of the coded signal to separate the components of the received signal corresponding to different propagation-delay paths. You can almost say that the rake receiver derives diversity gain from a potentially poor channel. After rake receiver despreading, a demodulation routine detects the transmitted data from each delayed-path component and combines the results.

Reverse Channel Coding

On the reverse CDMA channel in IS-95A, all the transmission uses 20 ms frames. The channels used in the reverse link are:

- The Traffic Channel and Access Channel. The Traffic channel carries data and speech. The data rate on the Traffic channel is variable; hence each frame may have a different rate.
- The Access channel carries control messages and requests to the base station.

The mobile station convolutionally encodes the data transmitted on the reverse channel. The station uses a 1/3-rate code when the Traffic channel uses Rate Set I, and a 1/2-rate code when the Traffic channel uses Rate Set II. The Access channel uses the 1/3-rate code. As in the case of the forward channel, the Viterbi algorithm provides optimal decoding at the receiver. Also, prior to modulation, symbols are repeated in the case of lower data rates, and the frame data is interleaved for protection against burst errors.

Mobile Station Walsh Modulation and Spreading

The reverse CDMA channel is composed of the Access and Traffic channels. These channels share the same assigned CDMA frequency using direct-sequence CDMA techniques. Each of these channels is assigned a distinct user long PN code sequence, which is used for spreading the spectrum of the signal.

All transmission on the reverse CDMA channel in IS-95A uses 20-ms frames. The data frames are convolutionally encoded and interleaved for error protection. The modulation used is a 64-ary orthogonal modulation. Direct sequence spreading and filtering are used to obtain the modulation signal for transmission. Because the reverse traffic channel has a variable rate,

randomized gating is used in the case of lower data rate frames to control the transmit power. Because the power control is done by gating, the power level of the transmitted symbols is kept constant, unlike the forward channel. The details of these transmitter tasks are in the IS-95A standard.

Noncoherent Rake Receiver

There is no pilot available for the reverse link transmission, as there is for the forward link. A pilot is valuable for obtaining good carrier channel estimation, making it possible to perform coherent detection and combining of multipath components. The absence of carrier phase and amplitude estimation necessitates either noncoherent or differential coherent detection. Timing of all paths must also be acquired and tracked. This discussion assumes that timing is available, but phase and amplitude estimates are not.

The Walsh modulator collects $\log_2 64 = 6$ data bits and transmits one of 64 orthogonal Walsh functions. The demodulator tracks L independent paths. Assume that each path has a separate demodulator, but that their outputs are noncoherently combined. The optimum noncoherent demodulator of Walsh modulation is a bank of 64 orthogonal noncoherent correlators. Each of the 64 noncoherent correlators squares each of the two quadrature components and adds the results. For a single path, this demodulator makes its decision by selecting the largest of the 64 output magnitudes. In the case of L paths, the demodulator adds the individual noncoherent correlator outputs for each of the L independent paths before making a decision.

In practice, it is possible to make a decision for each bit separately, rather than making decisions for all six bits at once. This may be done by finding the difference between the highest correlation values that correspond to the two possible values (0 and 1) of the bit of interest. Such an approach may be applied bit by bit and used to arrive at *soft decisions* (decisions whose difference value serves as a metric for decision reliability). Soft decisions can be used in the Viterbi decoder for the convolutional code to improve performance.

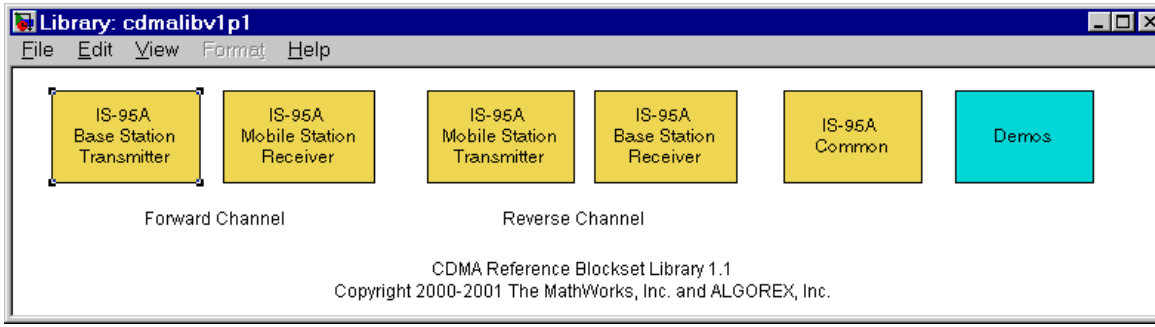
Main Library Structure

This section describes the structure of the CDMA Reference Blockset and summarizes the functionality of each library. It also explains referencing between libraries and how to open the main library.

Main Library Structure Overview

The main library of the CDMA Reference Blockset, shown below, is organized into these sublibraries:

- IS-95A Base Station Transmitter Library
- IS-95A Mobile Station Receiver Library
- IS-95A Mobile Station Transmitter Library
- IS-95A Base Station Receiver Library
- IS-95A Common Library



Functionality of Libraries

Blocks used in the Base Station Transmitter, the Base Station Receiver, the Mobile Station Transmitter, and the Mobile Station Receiver make up four libraries. Blocks that relate to more than one of these functional groups are collected in a fifth library, called Common. Blocks in the Common library include those that generate the different PN codes or determine frame quality.

The transmitter functions are divided between the Base Station Transmitter and Mobile Station Transmitter libraries, depending on their role in either the

forward link or reverse link, respectively. The corresponding receiver functions are not described in the standard, but are derived from the corresponding transmitter functions. The receiver functions are divided between the Base Station Receiver and Mobile Station Receiver libraries, depending on their role in either the reverse link or forward link, respectively.

The main library also displays the “Demos” icon. Use this icon to access the demo models, which are described in “Demo Models” on page 1-22

Reference Between Libraries

The libraries in the CDMA Reference Blockset are true libraries. This means that when you copy a block from a CDMA Reference Blockset library to the model window, Simulink creates a reference to the source block (in the library). Updates made to the source block affect every model block that refers to it. See *Using Simulink* for complete information about true libraries.

Opening the Main Library

To open the CDMA Reference Blockset, type

```
cdmalibv1p1
```

at the MATLAB command line. Double-click on a library icon in the main library window to open that library, and double-click on any icon in a library to open the block’s parameter dialog box.

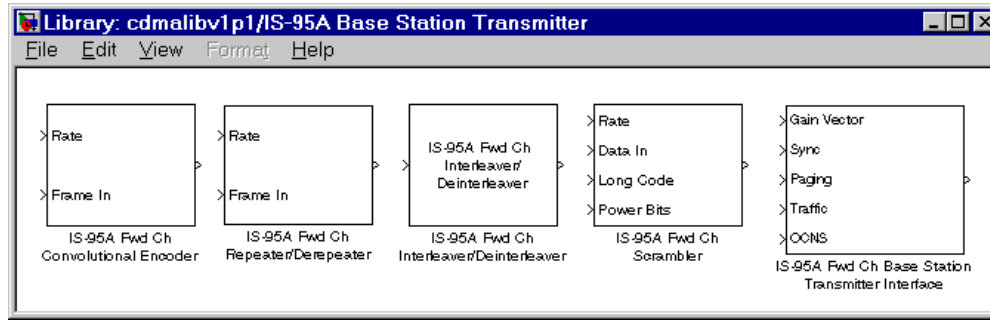
Library Summaries

This section surveys the contents of the blockset, considering each of the libraries in turn.

IS-95A Base Station Transmitter Library

The Base Station Transmitter Library provides the blocks required to perform the CDMA system base station or forward link transmitter functions.

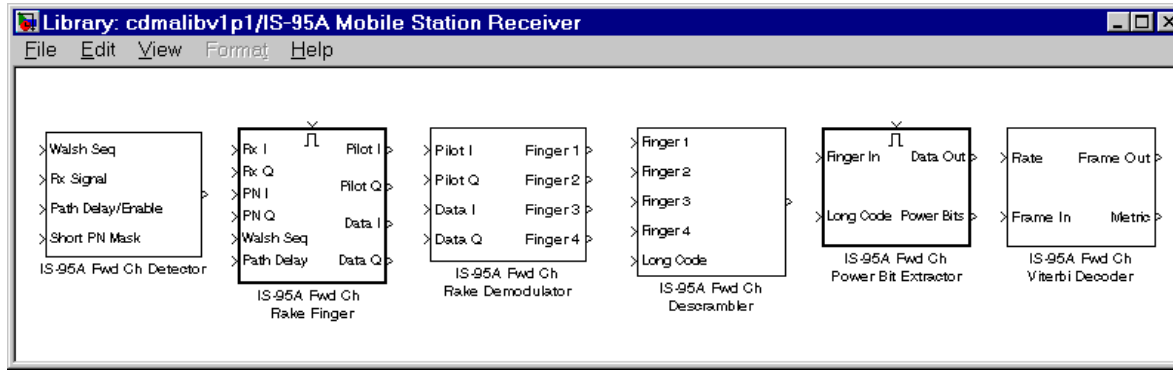
Function	Block(s)
Channel coding for the transmit data at the base station	IS-95A Fwd Ch Convolutional Encoder IS-95A Fwd Ch Repeater/Derepeater IS-95A Fwd Ch Interleaver/Deinterleaver
Pseudorandom scrambling of the transmitted data based on the user-specific long code	IS-95A Fwd Ch Scrambler
Spreading of the data sources corresponding to the different channels	IS-95A Fwd Ch Base Station Transmitter Interface



IS-95A Mobile Station Receiver Library

The Mobile Station Receiver Library provides the blocks required to perform the CDMA system mobile station or forward link receiver functions.

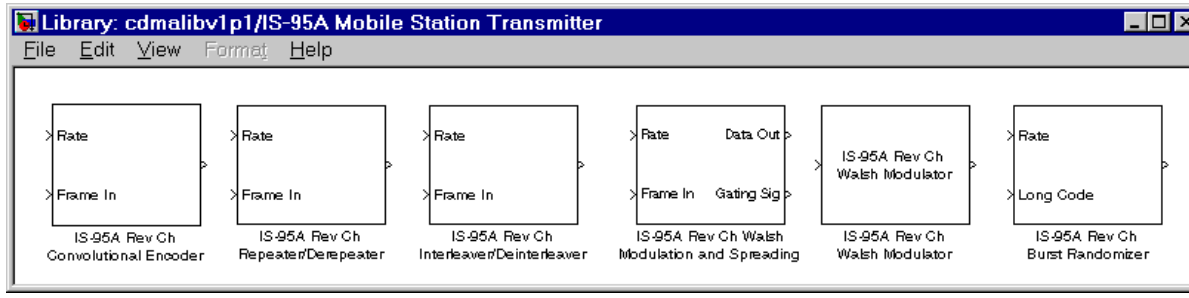
Function	Block(s)
Rake correlator for despreading of the received Pilot and data signals for demodulation	IS-95A Fwd Ch Rake Finger
Rake-based demodulation of the correlator data and the recovery of raw data symbols	IS-95A Fwd Ch Detector (includes IS-95A Fwd Ch Rake Finger, IS-95A Fwd Ch Rake Demodulator, and IS-95A Fwd Ch Descrambler) IS-95A Fwd Ch Rake Demodulator
Descrambling to undo the effect of data scrambling on the transmitter data and the removal of power bits to prepare the received data sequences for decoding.	IS-95A Fwd Ch Descrambler (includes IS-95A Fwd Ch Power Bit Extractor) IS-95A Fwd Ch Power Bit Extractor
Channel decoding of the raw data to recover the speech or data bits	IS-95A Fwd Ch Viterbi Decoder



IS-95A Mobile Station Transmitter Library

The Mobile Station Transmitter Library provides the blocks required to perform the CDMA system mobile station or reverse link transmitter functions. The functions and their corresponding blocks are shown below.

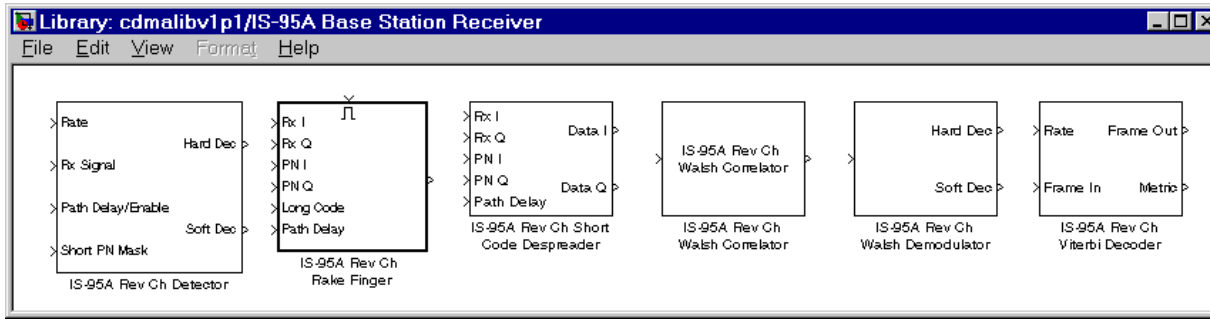
Function	Block(s)
Channel coding and interleaving of the information bits for error protection	IS-95A Rev Ch Convolutional Encoder IS-95A Rev Ch Repeater/Derepeater IS-95A Rev Ch Interleaver/Deinterleaver
Modulation of the signal to spread its spectrum	IS-95A Rev Ch Walsh Modulation and Spreading (includes IS-95A Rev Ch Walsh Modulator and IS-95A Rev Ch Burst Randomizer)
Walsh modulation, which involves the conversion of the channel symbol bits to one of several orthogonal Walsh codewords	IS-95A Rev Ch Walsh Modulator
Burst randomization, which consists of the gating of the transmitted signal using the long code	IS-95A Rev Ch Burst Randomizer



IS-95A Base Station Receiver Library

The Base Station Receiver Library provides the blocks required to perform the CDMA system base station or reverse link receiver functions.

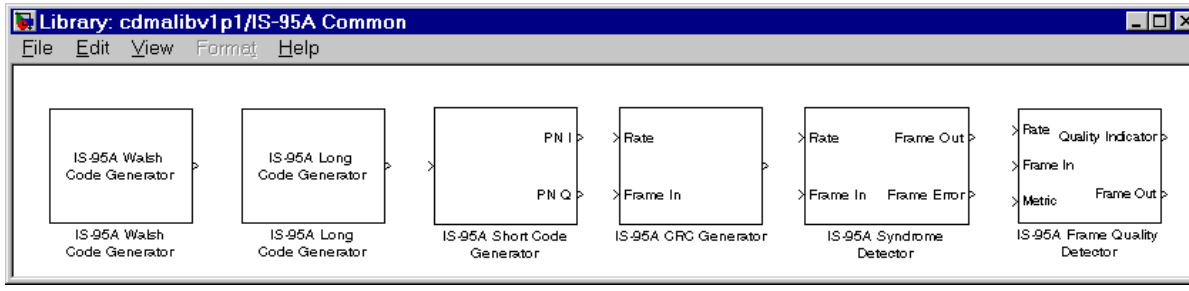
Function	Block(s)
Combining of the different rake fingers and the subsequent Walsh demodulation of the combined signal, while also accounting for the randomized transmit bursts	IS-95A Rev Ch Detector (includes IS-95A Rev Ch Rake Finger and IS-95A Rev Ch Walsh Demodulator) IS-95A Rev Ch Walsh Demodulator
Noncoherent rake correlator that despreads the signal by correlating with the short and long PN codes and the Walsh sequences	IS-95A Rev Ch Rake Finger (includes IS-95A Rev Ch Short Code Despreader and IS-95A Rev Ch Walsh Correlator) IS-95A Rev Ch Short Code Despreader
Correlation of the received and despread sequences with the Walsh sequences	IS-95A Rev Ch Walsh Correlator
Channel decoding of the raw data to recover the speech or user data	IS-95A Rev Ch Viterbi Decoder



IS-95A Common Library

The Common Library provides the blocks that are used in the transmitter as well as the receiver, or in the forward as well as the reverse link.

Function	Block
Walsh code generation for forward channel orthogonal spreading and despreading, and reverse channel modulation	IS-95A Walsh Code Generator
Long code generation for forward link scrambling and power-bit insertion, and for reverse link spreading and data burst randomization	IS-95A Long Code Generator
Short code generation for spreading and despreading of both the forward and reverse link signals in quadrature	IS-95A Short Code Generator
CRC generation, which provides redundant bits for error detection	IS-95A CRC Generator
CRC syndrome computation for detecting errors in received data	IS-95A Syndrome Detector
Frame quality detection using a combination of CRC and Viterbi decoder metrics	IS-95A Frame Quality Detector



Hints for Using the CDMA Reference Blockset

This section describes some conventions that the CDMA Reference Blockset uses. Understanding these conventions helps you interpret the demo models, construct your own models, and troubleshoot models.

Understanding Rates and Rate Sets

The IS-95A CDMA standard allows for data transmission at four different rates. The four rates together form a *rate set*. The CDMA Reference Blockset supports two rate sets. The first is based on the IS-95A standard, while the second is based on the J-STD-008 requirement. Both rate sets allow for varying-rate speech coders.

Some blocks in this blockset, such as the IS-95A Fwd Ch Repeater/Derepeater, behave differently depending on the data transmission rate. In such blocks, you specify the rate set as a dialog box parameter (called **Rate set** in the block dialog box) and you select the data transmission rate using an input signal (marked Rate near the input port of the block icon). After a simulation begins, you can change the data rate by changing the Rate input, but you cannot change the **Rate set** parameter.

The following table shows how different values for the Rate input determine the data transmission rate, for each of the two **Rate set** parameter values.

How Data Rate Input Value Determines Data Rate

Rate Input Value	Rate Set I Data Transmission Rate (bps)	Rate Set II Data Transmission Rate (bps)
0	9600 (Full rate)	14400 (Full rate)
1	4800 (Half rate)	7200 (Half rate)
2	2400 (Quarter rate)	3600 (Quarter rate)
3	1200 (Eighth rate)	1800 (Eighth rate)

Because the Rate value is an input to a block rather than a dialog box parameter, you can easily change the data rate during a simulation. However, the vector lengths of the input and output signals are fixed throughout the simulation at the *maximum* possible size for any data rate. This means that if,

at a given time in the simulation, the data rate is smaller than the maximum, then the input and output signals consist partially of relevant information and partially of placeholder bits that the block does not process. To find out how many bits are relevant for a particular block with a particular data rate, see the block's online reference page by pressing the block's **Help** button.

Note To ensure compatibility of blocks within your model, use the same **Rate set** parameter and Rate input for all blocks that process a given frame of data.

The modulation bit rate is the same for all rate sets and data rates, as a result of different convolutional coding and code repetition procedures. The symbol rate after interleaving is 28.8 ksps for the reverse channel and 19.2 ksps for the forward channel. However, for Rate Set II, data puncturing is performed before interleaving to bring the symbol rate from 28.8 to 19.2 ksps. This means that two of every six symbols are deleted. Therefore, the symbol rate of the forward link after the interleaver is 19.2 ksps for all rates and rate sets.

Understanding Frame Contents and Padding

For some blocks in this blockset, the size of an input or output signal does not necessarily indicate the number of elements that the block processes while performing its communication task. This means that studying or troubleshooting a model requires a clear understanding of how to interpret the contents of a frame. Examining signal sizes alone is not sufficient.

The Problem. Some blocks need to process different amounts of information in each frame, depending on their Rate input value, **Rate set** parameter, and/or **Channel type** parameter. As an example, the IS-95A Fwd Ch Convolutional Encoder library block encodes 192 bits when it simulates the encoder in a full-rate Paging channel, 96 bits when it simulates the encoder in a half-rate Paging channel, and 32 bits when it simulates the encoder in an eighth-rate Sync channel.

Furthermore, as the section "Understanding Rates and Rate Sets" on page 1-19 indicates, some blocks (including the IS-95A Fwd Ch Convolutional Encoder library block) allow you to change the data transmission rate during a simulation by changing the value of the Rate input signal. However, Simulink does not permit a signal to change its vector size during a simulation. This means that a block cannot accept an input data frame of 192 bits when its Rate

input indicates a full-rate channel, and 96 bits when its Rate input indicates a half-rate channel.

The Solution. To accept or produce the correct amount of data while also keeping its input and output vector sizes constant throughout the simulation, some blocks simply accept or produce padded data. For example, the IS-95A Fwd Ch Convolutional Encoder always accepts an input of exactly 288 bits. This number is the frame size that the convolutional encoder for a full-rate Rate Set II Traffic channel uses. If you use the block to simulate the encoder for a full-rate Traffic channel, then your input signal is naturally a vector of size 288. If you use the block to simulate the encoder for a full-rate Paging channel, then you must pad the 192 bits that you want to encode with 96 zeros, so that the input vector to the block has a vector size of 288. Similarly, the block always produces an output of exactly 576 bits. It automatically pads its output vector with zeros if the coded data frame consists of fewer than 576 bits.

If a block uses this padded signal scheme to accommodate varying data amounts while keeping signal sizes constant, then its block reference page includes information such as:

- The vector size of input and output signals
- The number of elements of an input or output vector representing the actual data, as opposed to zeros used to pad the vector; columns of a table list the “Number of Relevant Input Bits” and/or “Number of Relevant Output Bits” for given channel types and rates.

You can access the block’s reference page by pressing the **Help** button in its dialog box.

Demo Models

This section describes several demonstration models to illustrate how to use the blockset to model and test various aspects of the communication system. You can use the demos to familiarize yourself with some common system configurations. You can also use the demos as a starting point and customize them to meet your specific needs.

Demos for the CDMA Reference Blockset are listed in the table below. To begin working with a demo, see "Using the Demos" on page 1-23.

List of Demos	Filename
IS-95A Forward Traffic Channel End-to-End Demo	`is95fwdchendtoend2`
IS-95A Forward Traffic Channel Detection Demo	`is95fwdchdetection2`
IS-95A Reverse Traffic Channel Detection Demo	`is95revchdetection2`
IS-95A Forward Traffic Channel Codec Demo	`is95fwdchcodec2`
IS-95A Reverse Traffic Channel Codec Demo	`is95revchcodec2`
IS-95A Reverse Traffic Channel Transmitter Demo	`is95revchtransmitter2`

The demos cover these areas:

- Transmitters
- Modulation and demodulation
- Channel coders and decoders

More specifically, the demos show the nature of signal encoding, channel error correction schemes, signal spreading, and modulation schemes used in the IS-95 forward link and reverse link transmission. The receivers in the demos illustrate the corresponding despreading and detection of an IS-95A spread-spectrum signal.

Using the Demos

This section describes how to open and run the demos, and view the results. More advanced users can also change parameters and customize the demo.

Opening a Demo

To open a demo, follow these steps:

1. Open the main CDMA Reference Blockset library by typing `cdmalibv1p1` at the MATLAB prompt.
2. Double-click on the Demos icon. This opens the MATLAB Demos window and selects the CDMA Reference Blockset in the left pane.
3. In the lower right pane of the MATLAB Demos window, double-click on the name of the demo that you want to open.

Alternatively, you can open a demo by typing the demo filename at the MATLAB prompt. For a list of filenames, see the "List of Demos" on page 1-22.

Exploring and Running a Demo

You can examine the demo before running it. Here are some suggestions to get you started:

- Study the block diagram, noticing which blocks are in the model and how they are connected.
- Double-click on any icon to open it (if it is a subsystem that has no mask) or to see its parameter dialog box. A parameter dialog box briefly describes the block, shows its parameter values, allows you to change any of the parameter values, and also includes a **Help** button. If the block is linked to a library instead of being a custom block for the demo, then the **Help** button links to online help for the block.
- Select any CDMA Reference Blockset block, and then select **Go to library link** from the model window's **Edit** menu, to open the blockset library in which the block resides. This shows you where to find the block if you later want to use it in your own model.

To run the demo after opening it, select **Start** from the model window's **Simulation** menu. To stop or pause the simulation while it is running, select **Stop** or **Pause** from the **Simulation** menu.

Viewing or Collecting Results from a Demo

Some of the demos use Display and Scope blocks, both from the Simulink Sinks library, to show performance results such as the bit error rate (BER). These blocks display data continuously during the simulation using curves or a numeric display of the values.

To help retrieve and further process the resulting data, you can also connect any number of To Workspace blocks, from the Simulink Sinks library, to parts of the model that interest you. The To Workspace block transfers the data from the simulation to the MATLAB workspace. Then you can plot and analyze the data in the MATLAB workspace.

To save data directly in MATLAB format, you can connect the To File block (from the Simulink Sinks library) to parts of the model that interest you. The writing of data on the disk may slow down the simulation. However, when logging a large data series, it is better to write to a file than to fill the system memory as the To Workspace block would do.

Changing a Demo's Parameters

To understand the function of a demo, you should first run it with the parameters provided. If you later want to modify the configuration to be tested or the conditions under which a demo is run, you can change the parameters in the blocks of the demo.

Note Before attempting to change a demo's parameters, you should have a good understanding of how the demo works.

To view a block's parameters, double-click on the block's icon. To change a parameter, enter a new value in the parameter dialog box and click the **OK** button, or the **Apply** button if there is one.

Note Some blocks or systems impose consistency constraints on various block parameters. For instance, to change the channel types from the default, you might need to change parameters in several blocks. Furthermore, you might need to change parameters such as sizes of data vectors from the model's sources appropriately.

Some typical properties of the demos that you can modify by changing parameters include:

- Simulation duration
- Data rate for the user channel
- Type of logical channel
- Signal-to-noise (SNR) ratio
- Multipath channel fade rate or Doppler frequency

Customizing a Demo

You can customize any demo by adding, removing, and modifying blocks from the CDMA Reference Blockset, Communications Toolbox, or Signal Processing Blockset. By customizing the demos, you can vary the aspects of the CDMA system that the demos simulate. Possible changes include:

- Different control channels
- Different traffic channel configurations
- Further simulation of system algorithms such as timing acquisition, gain control and frequency control

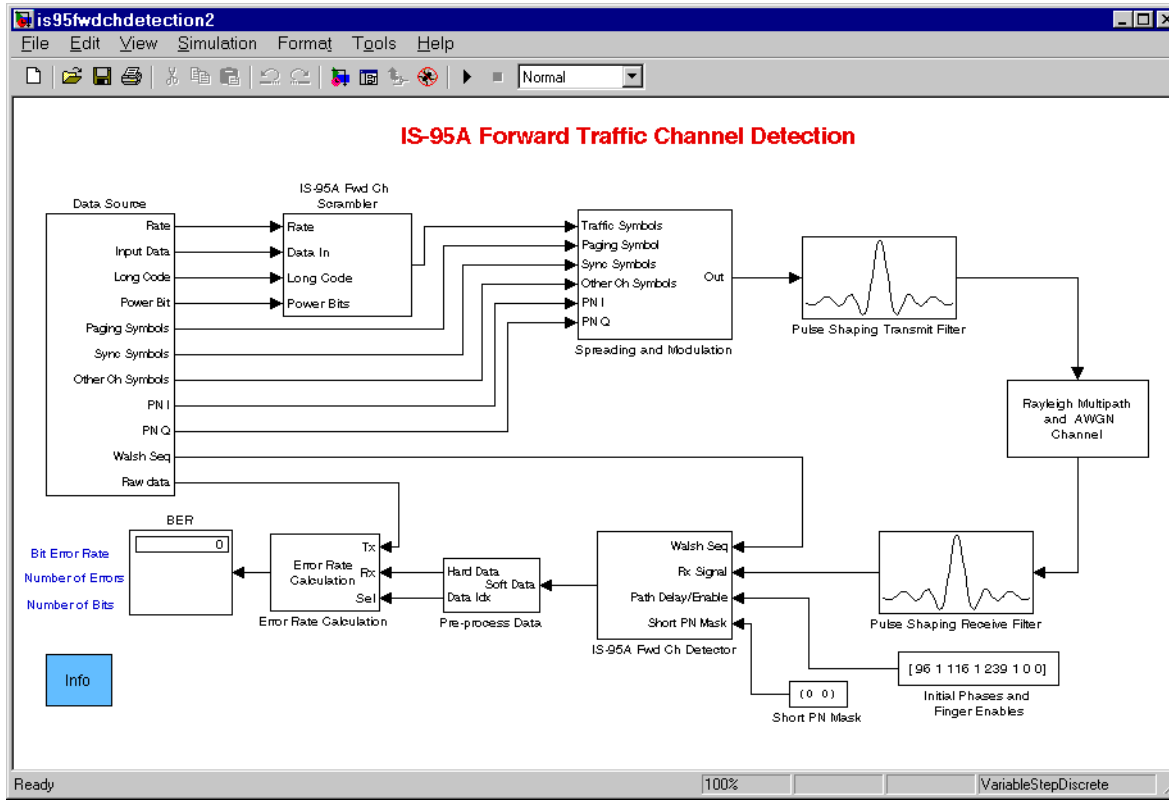
Note Before attempting to customize a demo, you should have a good understanding of how it works.

IS-95A Forward Traffic Channel Detection Demo

The IS-95A Forward Traffic Channel Detection demo shows the modulation and spreading of the IS-95A CDMA signal at the base station transmitter, as well as despreading and coherent demodulation of the signal at the mobile

station receiver. The basic components of this demo are the transmitter, the receiver, and the channel. The resulting raw (without channel coding) bit error rate (BER) is displayed in the simulation as a measure of performance. The demo essentially describes how the spreading and despreading of the data symbols are performed with the blockset and illustrates the usage of the system components involved.

You can open the model by following the instructions in “Opening a Demo” on page 1-23, or by typing `is95fwdchdetection2` at the MATLAB prompt.



Library Blocks in the Demo

IS-95A Forward Traffic Channel Detection demo uses these library blocks from the CDMA Reference Blockset:

- IS-95A Fwd Ch Base Station Transmitter Interface (inside Spreading and Modulation subsystem)
- IS-95A Fwd Ch Detector
- IS-95A Long Code Generator (inside Data Source subsystem)
- IS-95A Short Code Generator (inside Data Source subsystem)
- IS-95A Walsh Code Generator (inside Data Source subsystem)
- IS-95A Fwd Ch Scrambler

How the Demo Works

The role of the transmitter section is to generate the modulated waveform that contains the various forward link channel components including the Pilot, Sync, Paging, user Traffic, and other users' Traffic channels. The transmitter components are the data source, the scrambler, the spreading and modulation, and the transmit filter.

The Data Source subsystem produces many data elements in this simulation. It contains:

- Random number generators that generate the bipolar random data symbols for the Traffic, Sync and Paging channels, as well as the interfering Other Ch Symbols input
- A signal that represents the data rate
- The IS-95A Long Code Generator library block, which generates the long code used to scramble the data
- The IS-95A Walsh Code Generator and IS-95A Short Code Generator library blocks, which generate the Walsh and complex PN codes, respectively. These codes are used for orthogonal spreading of the modulation symbol data.

The IS-95A Fwd Ch Scrambler library block uses the decimated long code input to scramble the input traffic frame and insert power bits. The scrambled traffic channel data and the data symbols from the various other types of channel sources are input to the Spreading and Modulation subsystem. These inputs are orthogonally encoded by their respective Walsh codes, added, and spread with the in-phase and quadrature components of the short PN sequence. Part of this is accomplished by the IS-95A Fwd Ch Base Station Transmitter Interface library block, which is inside the Spreading and Modulation subsystem. The signal generated is processed by the pulse shaping Transmit Filter block, which generates the modulated I and Q waveforms.

The Rayleigh Multipath and AWGN Channel subsystem simulates the propagation through multiple paths of a Rayleigh fading channel. Complex white Gaussian noise is added to the channel output. This noise represents the interference generated by other base stations that are using the same frequency band.

The receiver section of the system is responsible for the recovery of the data symbols transmitted on the traffic channel. The operations performed in this section include the receive filtering, the rake correlator, the rake demodulator, and descrambling. The Receive Filter block performs FIR filtering on the I and Q sample streams with a filter that is matched to the transmit filter to maximize the in-band signal-to-noise ratio.

The IS-95A Fwd Ch Detector library block is a masked subsystem with several components inside. The rake receiver computes symbol duration correlations for the Traffic data and Pilot symbols. These correlation values are used by the IS-95A Fwd Ch Rake Demodulator to recover the Traffic channel symbols. The Traffic symbols are further processed by the IS-95A Fwd Ch Descrambler to obtain the decision values for the original transmitted data symbols.

This simulation uses the raw BER as the measure of the performance under the channel and noise conditions selected. After the Pre-process Data subsystem makes a hard decision on the decoded data, the raw BER is obtained by comparing the hard decisions with the transmitted bits.

Visible Results of the Demo

The demo shows the raw BER for the forward channel. The BER depends on the channel conditions and the number of rake receiver fingers active. As the signal-to-noise ratio or the channel conditions are changed, the effect of these on the raw BER can be seen by running the model for these different conditions.

Changing Demo Parameters

This section suggests some ways that you can alter the parameters in the demo after you understand the model.

Simulation Duration. The simulation duration in this demo is set to 2 seconds. Because the frame duration in IS-95A is 0.02 second, this model processes 100 frames. This gives sufficient time for the BER measurements to converge at reasonable SNR settings. To run the simulation for a longer or shorter time,

select **Parameters** from the **Simulation** menu, and change the Stop time to 0.02 times the number of frames you want to simulate.

Data Rate. To change the data rate, double-click on the Data Source icon. Then double-click on the Base Station Transmitter Data Rate icon, and change the **Data Rate** parameter. Choices are Full, Half, Quarter, and One-Eighth.

Rate Set. The **Rate set** parameter is Rate Set I by default in the simulation, but alternatively you can select Rate Set II. To change the **Rate set** parameter, double-click on IS-95A Fwd Ch Scrambler and the IS-95A Fwd Ch Detector icons, and select either Rate Set I or Rate Set II from the **Rate set** parameter menu.

Doppler Frequency in Channel. To change the Doppler frequency, double-click on the Rayleigh Multipath and AWGN Channel icon. Then double-click on the Multipath Rayleigh Fading icon, and change the **Doppler frequency** parameter. Reasonable Doppler frequencies representative of the cellular mobile environment are in the 0 to 200 Hz range.

Signal-to-Noise Ratio. To change the signal-to-noise ratio, double-click on the Rayleigh Multipath and AWGN Channel icon. Then double-click on the AWGN Channel icon, and change the **Es/No** parameter. The detection performance improves with an increase in the signal-to-noise ratio.

Random Seed. To run the simulation with different seeds, change the **Initial seed** parameters in one or more of these: Random Data Frame Generator, Random Power Bits, Multipath Rayleigh Fading, and AWGN Channel. The first two items are inside the Data Source subsystem and the last two items are inside the Rayleigh Multipath and AWGN Channel subsystem.

Channel Paths and Rake Fingers. Changes in the number or delay of fading channel paths and rake fingers must correspond to each other. To change the number or delay of fading channel paths, double-click on the Rayleigh Multipath and AWGN Channel icon. Then double-click on the Multipath Rayleigh Fading icon and change the **Delay vector** parameter. The vector length of this parameter is the number of paths and the vector elements measure the delay of each path in seconds.

To change the number or delay of rake fingers, double-click on the Initial Phases and Finger Enables icon and change the **Constant value** parameter. The four consecutive pairs of elements of this length-eight vector indicate the

delay and status of the four rake fingers. The delay is measured in samples (not in seconds). A status of zero disables the finger and a status of one enables the finger. By default, this demo enables three rake fingers, with delays of 96, 116, and 239 samples, respectively. These delays incorporate the filter delays of 96 samples as well as the three default channel path delays of 0, $2x10^{-6}$ seconds, and $14.5x10^{-6}$ seconds, respectively. (To convert a delay from seconds to samples, multiply by the chip rate of 1.2288 Mcps, and then by the oversampling rate of 8.)

Further Explorations

The demo serves as the starting point for building simulation models with different characteristics and conditions. Parameters in the demo's components may be changed for different configurations. For instance, you can alter the channel type in all components (on the main level or within subsystems) that include a **Channel type** parameter, to obtain a simulation for a different configuration.

Other possible changes that you can make include:

- Introducing components from the codec
- Replacing the transmit and receive filters by systems developed using the Signal Processing Blockset with different oversampling rates
- Introducing an analog-to-digital converter to observe the performance degradation at the receiver

---

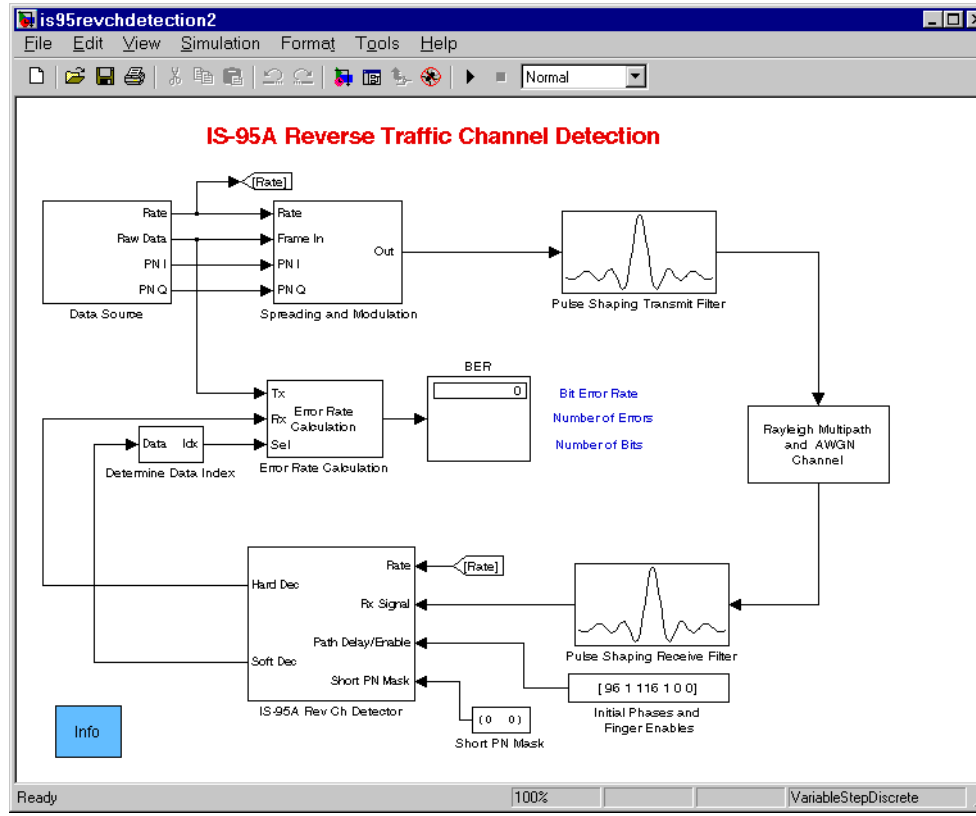
Note Some changes involve consistency considerations. For instance, a change in the oversampling rate requires corresponding changes in the channel, rake receiver, and the filters.

IS-95A Reverse Traffic Channel Detection Demo

The IS-95A Reverse Traffic Channel Detection demo simulates modulation and spreading of the data symbols at the mobile station transmitter, as well as despreading and noncoherent demodulation at the base station receiver. At the transmitter, the symbols are modulated by a Walsh modulator and spread, and a randomized gating is applied to the transmit bursts. At the receiver, the

noncoherent rake demodulator recovers the data. The bit error rate for the data is displayed in the simulation.

You can open the model by following the instructions in “Opening a Demo” on page 1-23, or by typing `is95revchdetection2` at the MATLAB prompt.



Library Blocks in the Demo

The IS-95A Reverse Traffic Channel Detection demo uses these library blocks from the CDMA Reference Blockset:

- IS-95A Rev Ch Detector

- IS-95A Rev Ch Walsh Modulation and Spreading, which is inside the Spreading and Modulation subsystem
- IS-95A Short Code Generator, which is inside the Data Source subsystem

How the Demo Works

Among the components used at the transmitter, the Data Source subsystem provides a source for random data bits, the data rate for the channel, and the short pseudonoise (PN) code used for the in-phase and quadrature spreading of the signal. In particular, the IS-95A Short Code Generator library block generates the short PN code.

The Spreading and Modulation subsystem contains several blocks that are responsible for the Walsh modulation, and the spreading with the long and short PN codes. The IS-95A Rev Ch Walsh Modulation and Spreading library block contains the IS-95A Rev Ch Burst Randomizer library block, which processes the long code and generates a gating signal based on the long code and the data rate of the input frame. The IS-95A Rev Ch Walsh Modulation and Spreading library block also groups the input data in 6-bit groups, maps each group of 6 bits to a 64-symbol Walsh code, upsamples these 64 symbols by a factor of 4 to bring the result to chip rate, and then spreads the upsampled symbols with the gated long code. This gating ensures that the transmission is only performed for a fraction of the frame duration (half the time for half rate, and so on). Finally, other portions of the Spreading and Modulation subsystem spreads the data in quadrature by the PN code.

The Transmit Filter block generates the I and Q waveforms. The Q waveform is delayed by a 1/2-chip duration relative to the I waveform.

The Rayleigh Multipath and AWGN Channel subsystem simulates the propagation through multiple paths of a Rayleigh fading channel. Complex white Gaussian noise is added to the channel output; this noise represents the interference generated by other base stations that are using the same frequency band.

In the receiver section, the incoming signal is first filtered by the Receive Filter block, which implements a filter matched to the transmit filter. The filters in this demo are designed to maximize the signal power within the desired frequency band.

Then the filtered signal is sent to the IS-95A Rev Ch Detector library block, which contains the reverse channel rake receiver. The rake receiver consists of

three rake fingers that are set to different delays to handle up to three multipaths. Each active rake finger performs the despreading of the input data with the short PN sequence, followed by despreading with the long code. This is followed by the correlation with the entire set of 64 Walsh codes. The energies in the I and Q components are added, and the results from the fingers are added together. This is processed by the Walsh demodulator, which generates decisions in groups of 6 bits, the size used for modulation. The Walsh demodulator outputs both soft decisions and bipolar-valued hard decisions. both are gated by the data burst randomizer signal.

The Error Rate Calculation block compares the transmitted bits and the received decisions, and produces the raw (that is, without channel coding) bit error rate. This rate is displayed in the model window during the simulation.

Visible Results of the Demo

The demo shows the raw bit error rate (BER) for the reverse channel. The BER depends on the channel conditions and the number of rake receiver fingers active. As the signal-to-noise ratio or the channel conditions are changed, the effect of these on the raw BER can be seen by running the model for these different conditions.

Changing Demo Parameters

This section suggests some ways that you can alter the parameters in the demo after you understand the model.

Simulation Duration. The simulation duration in this demo is set to 2 seconds. Because the frame duration in IS-95A is 0.02 second, this model processes 100 frames. This gives a sufficient time for the BER measurements to converge at reasonable SNR settings. To run the simulation for a longer or shorter time, select **Parameters** from the **Simulation** menu and change the Stop time to 0.02 times the number of frames you want to simulate.

Data Rate. To change the data rate, double-click on the Data Source icon. Then double-click on the Mobile Station Transmitter Data Rate icon and change the **Data Rate** parameter. Choices are Full, Half, Quarter, and One-Eighth rates.

Long Code Mask. To change the long code mask, double-click on the Spreading and Modulation icon. Then double-click on the IS-95A Rev Ch Walsh Modulation and Spreading icon, and enter a new value for the **Long code mask** parameter. This parameter can be any nonnegative integer less than 2^{42}-1.

The block uses the binary representation of this number to generate the code. The value for the **Long code mask** parameter is also required in the IS-95A Rev Ch Detector library block. Experiment with using the same or different values in the two blocks. This illustrates why the receiver does not demodulate the signal not intended for it, although they are transmitted in the same frequency band. For more information, see the IS-95A specification.

Doppler Frequency in Channel. To change the Doppler frequency, double-click on the Rayleigh Multipath and AWGN Channel icon. Then double-click on the Multipath Rayleigh Fading icon, and change the **Doppler frequency** parameter. Reasonable Doppler frequencies representative of the cellular mobile environment are in the 0 to 200 Hz range.

Signal-to-Noise Ratio. Change the **Es/No** parameter in the AWGN Channel block in the channel subsystem. The detection performance improves with an increase in the signal-to-noise ratio.

Random Seed. To run the simulation with different seeds, change the **Initial seed** parameters in one or more of these:

- Random Data Frame Generator
- Multipath Rayleigh Fading
- AWGN Channel

The first item is inside the Data Source subsystem and the last two blocks are inside the Rayleigh Multipath and AWGN Channel subsystem.

Channel Paths and Rake Fingers. Changes in the number or delay of fading channel paths and rake fingers must correspond to each other. To change the number or delay of fading channel paths, double-click on the Rayleigh Multipath and AWGN Channel icon. Then double-click on the Multipath Rayleigh Fading icon and change the **Delay vector** parameter. The vector length of this parameter is the number of paths and the vector elements measure the delay of each path in seconds.

To change the number or delay of rake fingers, double-click on the Initial Phases and Finger Enables icon and change the **Constant value** parameter. The three consecutive pairs of elements of this length-six vector indicate the delay and status of the three rake fingers. The delay is measured in samples, not in seconds. A status of zero disables the finger and a status of one enables the finger. By default, this demo enables two rake fingers, with delays of 96 and

116 samples, respectively. These delays incorporate the filter delays of 96 samples, as well as the two default channel path delays of 0 and $2x10^{-6}$ seconds. (To convert a delay from seconds to samples, multiply by the chip rate of 1.2288 Mcps and then by the oversampling rate of 8.)

Further Explorations

The demo serves as the starting point for building simulation models with different characteristics and conditions. Parameters in the demo's components may be changed for different configurations. For instance, you can alter the channel type in all components (on the main level or within subsystems) that include a **Channel type** parameter, to obtain a simulation for a different configuration.

Other possible changes that you can make include:

- Introducing components from the reverse channel codec
- Replacing the transmit and receive filters by systems developed using the Signal Processing Blockset with different oversampling rates, coefficients, or precision –for instance, to evaluate the effects of fixed-point implementation
- Introducing an analog-to-digital converter to observe the performance degradation at the receiver

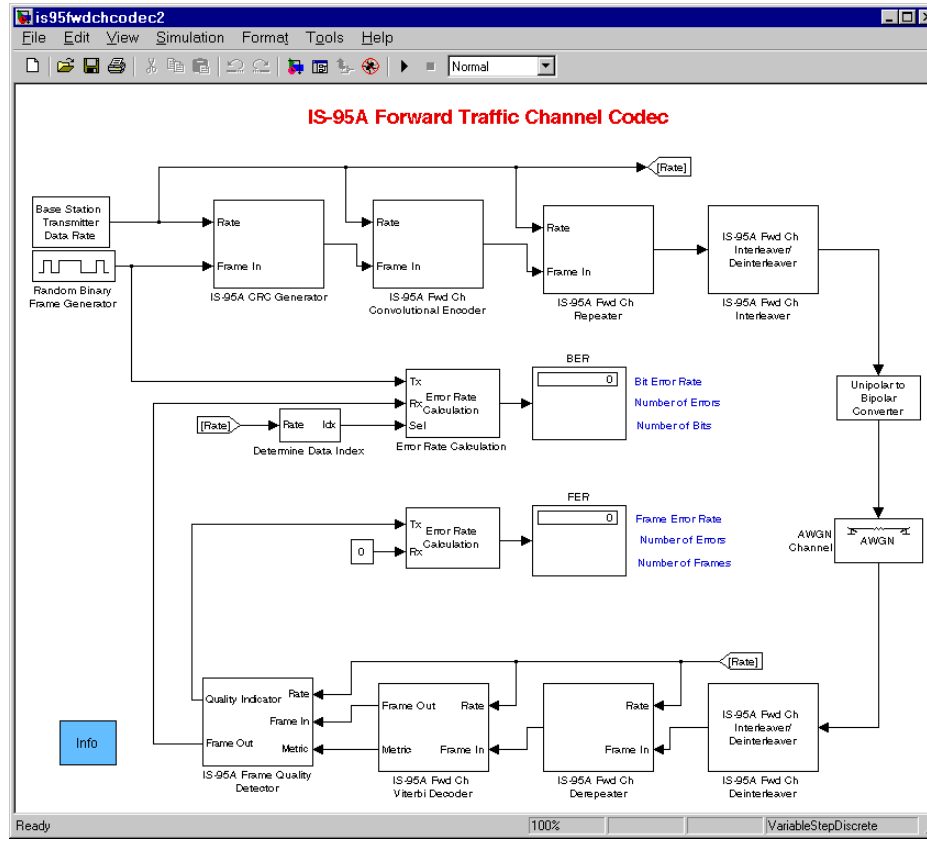
---

Note Some changes involve consistency considerations. For instance, a change in the oversampling rate requires corresponding changes in the channel, rake receiver, and the filters.

IS-95A Forward Traffic Channel Codec Demo

The IS-95A Forward Traffic Channel Codec demo shows the channel coding at the base station and the corresponding decoding at the mobile station receiver. The transmitter encodes the data, and the channel model adds noise to simulate errors in transmission. The receiver retrieves the information bits by performing the decoding tasks. The operations performed conform to the IS-95A channel coding requirements, and illustrate the usage of the corresponding elements of the blockset.

You can open the model by following the instructions in "Opening a Demo" on page 1-23, or by typing `is95fwdchcodec2` at the MATLAB prompt.



Library Blocks in the Demo

The IS-95A Forward Traffic Channel Codec demo uses these library blocks from the CDMA Reference Blockset:

- IS-95A CRC Generator
- IS-95A Frame Quality Detector
- IS-95A Fwd Ch Convolutional Encoder
- IS-95A Fwd Ch Interleaver/Deinterleaver
- IS-95A Fwd Ch Repeater/Derepeater
- IS-95A Fwd Ch Viterbi Decoder

How the Demo Works

The base station transmitter section performs the CRC (cyclic redundancy check) generation, convolutional encoding, symbol repetition, and interleaving. The Random Binary Frame Generator masked subsystem generates random data that act as information bits. The Base Station Transmitter Data Rate masked subsystem provides the selection of the data rate. The IS-95A CRC Generator library block appends the CRC bits to the information bits. These CRC bits are used to detect errors in the data frame at the receiver. The IS-95A Fwd Ch Convolutional Encoder library block convolutionally encodes the data using a 1/2-rate encoder for protection against channel errors. Because IS-95A supports variable data rate operation, the data frame at this stage can have a number of different sizes. Depending on the data rate, the IS-95A Fwd Ch Repeater/Derepeater library block (denoted IS-95A Fwd Ch Repeater in this demo) may repeat the bits it receives to create a data frame of 384 symbols. (If you change the simulation so that it uses Rate Set II, then the block also punctures the input frame; that is, it removes some of the input symbols to keep the output frame size constant.) Then the IS-95A Fwd Ch Interleaver/Deinterleaver library block (denoted IS-95A Fwd Ch Interleaver in this demo) interleaves the data frame for protection against the localized error bursts that can occur in fading channel conditions.

This interleaved data is converted to the bipolar form, and the AWGN Channel block adds white Gaussian noise to each symbol of the bipolar data. This noise represents the random error in the demodulation of the symbol.

The mobile side performs the reverse operations. The IS-95A Fwd Ch Interleaver/Deinterleaver library block (denoted IS-95A Fwd Ch Deinterleaver in this demo) deinterleaves the input data to restore the original symbol ordering. The IS-95A Fwd Ch Repeater/Derepeater library block (denoted IS-95A Fwd Ch Derepeater in this demo) derepeats the symbols depending on the symbol rate, which involves averaging the symbols that were repeated. (If you change the simulation so that it uses Rate Set II, then the block also depunctures the data by inserting zeros in the locations corresponding to the punctured symbols.) The resulting frame is then provided as input to the IS-95A Fwd Ch Viterbi Decoder library block, which retrieves the information that was previously encoded. The decoded information bits and the CRC bits are provided to the IS-95A Frame Quality Detector library block. The final metrics from the IS-95A Fwd Ch Viterbi Decoder block are also input to the IS-95A Frame Quality Detector block, which decides whether the frame was correctly received. The IS-95A Frame Quality Detector block outputs the

Quality Indicator signal, as well as the information bits without the CRC bits. One Error Rate Calculation block compares the information bits to the bits generated at the source, while another Error Rate Calculation block compares the Quality Indicator bits with zero. Finally, the resultant bit and frame error rates are displayed.

Visible Results of the Demo

The demo shows the BER and the FER for the forward channel coding and decoding operations. Both of these error rates show the protection that the coding offers against errors introduced in the channel. The error rates depend on the additive noise in the system, as well as the data rate in the simulation. Your can see the error rates change as you change the **SNR** parameter in the AWGN Channel block, or the **Data Rate** parameter inside the Base Station Transmitter Data Rate masked subsystem.

Changing Demo Parameters

This section suggests some ways that you can alter the parameters in the demo after you understand the model.

Simulation Duration. The simulation duration in this demo is set to 10 seconds. Because the frame duration in IS-95A is 0.02 second, this model processes 500 frames. This gives a sufficient time for the BER measurements to converge at reasonable SNR settings. For lower SNR settings the FER may take a longer time to converge. To run the simulation for a longer or shorter time, select **Parameters** from the **Simulation** menu, and change the Stop time to 0.02 times the number of frames you want to simulate.

Data Rate. To change the data rate, double-click on the Base Station Transmitter Data Rate icon and change the **Data Rate** parameter. Choices are Full, Half, Quarter, and One-Eighth rates.

Rate Set. The **Rate set** parameter is Rate Set I by default in the simulation, but alternatively you may select Rate Set II. The **Rate set** parameter should be changed in all of the blocks by double-clicking on their icons and selecting either Rate Set I or Rate Set II from the **Rate set** parameter menu. The blocks that need to be changed to the same **Rate set** parameter are the IS-95A CRC Generator, IS-95A Fwd Ch Convolutional Encoder, IS-95A Fwd Ch Repeater, IS-95A Fwd Ch Derepeater, IS-95A Fwd Ch Viterbi Decoder, and IS-95A Frame Quality Detector.

Signal-to-Noise Ratio. Change the **SNR** parameter in the AWGN Channel block. The detection performance improves with an increase in the signal-to-noise ratio.

Random Seed. To run the simulation with different seeds, change the **Initial seed** parameters in the Random Binary Frame Generator masked subsystem and/or the AWGN Channel block.

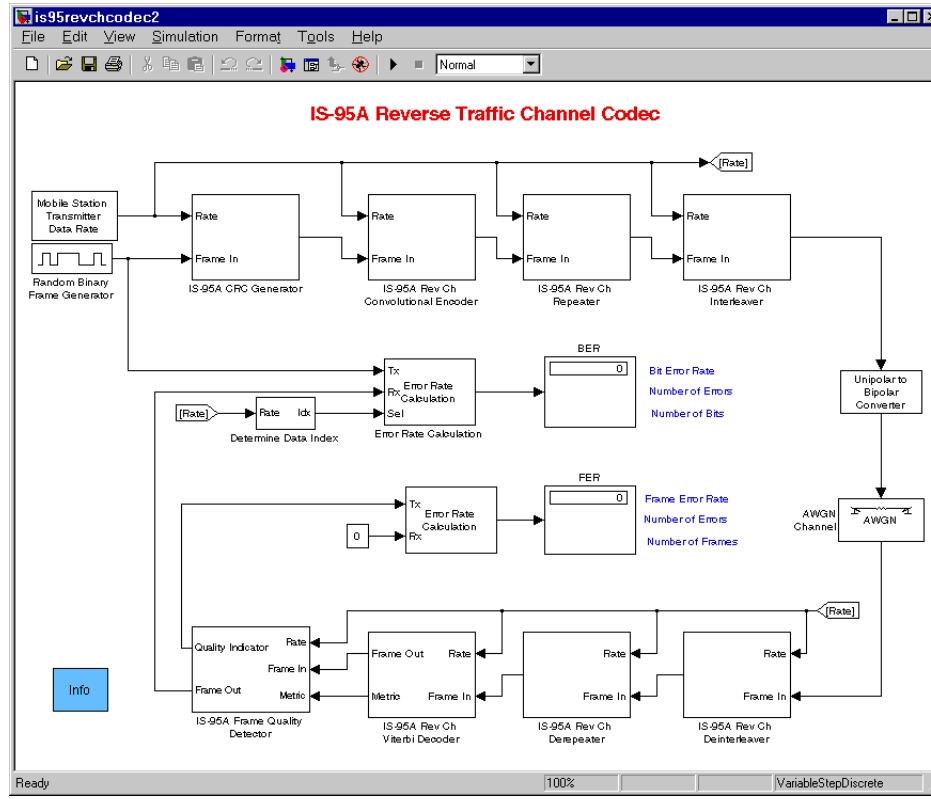
Further Explorations

The demo is the starting point for building simulation models for channel coding and decoding. It can be used for developing codec models for an other forward link channels such as the Sync and Paging channels. In addition, it is a good platform for experimenting with variations in channel decoding algorithms such as soft-decision quantization, and specialized implementations of the Viterbi decoder. Furthermore, by introducing the spreading and despreading components in the simulation, you can simulate the forward link in greater detail.

IS-95A Reverse Traffic Channel Codec Demo

The IS-95A Reverse Traffic Channel Codec demo shows the channel coding at the mobile station and the corresponding decoding at the base station receiver. The transmitter encodes the data, and the channel model adds noise to simulate errors in transmission. The receiver side retrieves the information bits by performing the decoding tasks. The operations performed conform to the IS-95A channel coding requirements, and illustrate the usage of the corresponding elements of the blockset.

You can open the model by following the instructions in “Opening a Demo” on page 1-23, or by typing `is95revchcodec2` at the MATLAB prompt.



Library Blocks in the Demo

The IS-95A Reverse Traffic Channel Codec demo uses these library blocks from the CDMA Reference Blockset:

- IS-95A CRC Generator
- IS-95A Frame Quality Detector
- IS-95A Rev Ch Convolutional Encoder
- IS-95A Rev Ch Interleaver/Deinterleaver (two instances)
- IS-95A Rev Ch Repeater/Derepeater (two instances)
- IS-95A Rev Ch Viterbi Decoder

How the Demo Works

The base station transmitter section performs the Cyclic Redundancy Check (CRC) generation, convolutional encoding, symbol repetition, and interleaving. The Random Binary Frame Generator masked subsystem generates random data that act as information bits. The Mobile Station Transmitter Data Rate masked subsystem provides the selection of the data rate. The IS-95A CRC Generator library block appends the CRC bits to the information bits. These CRC bits are used to detect errors in the data frame at the receiver. The IS-95A Rev Ch Convolutional Encoder library block convolutionally encodes the data using a 1/2-rate encoder for protection against channel errors.

Because IS-95A supports variable data rate operation, the data frame at this stage can have a number of different sizes. Depending on the data rate, the IS-95A Rev Ch Repeater/Derepeater library block (denoted IS-95A Rev Ch Repeater block in this demo) may repeat the bits it receives to create a data frame of 384 symbols. Note that if you change the simulation so that it uses Rate Set II, then the block also punctures the input frame; that is, the block removes some of the input symbols to keep the output frame size constant. Then the IS-95A Rev Ch Interleaver/Deinterleaver library block (denoted IS-95A Rev Ch Interleaver in this demo) interleaves the data frame for protection against the localized error bursts that can occur in fading channel conditions.

This interleaved data is converted to the bipolar form, and the AWGN Channel block adds white Gaussian noise to each symbol of the bipolar data. This noise represents the random error in the demodulation of the symbol.

The mobile side performs the reverse operations. The IS-95A Rev Ch Interleaver/Deinterleaver library block (denoted IS-95A Rev Ch Deinterleaver in this demo) deinterleaves the input data to restore the original symbol ordering. The IS-95A Rev Ch Repeater/Derepeater library block (denoted IS-95A Rev Ch Derepeater in this demo) derepeats the data depending on the symbol rate, which involves averaging the symbols that were repeated. Note that if you change the simulation so that it uses Rate Set II, then the block also depunctures the data by inserting zeros in the locations corresponding to the punctured symbols. The resulting frame is then provided as input to the IS-95A Rev Ch Viterbi Decoder library block, which retrieves the information that was previously encoded. The decoded information bits and the CRC bits are provided to the IS-95A Frame Quality Detector library block. The final metrics from the IS-95A Rev Ch Viterbi Decoder block are also input to the IS-95A Frame Quality Detector block, which decides whether the frame was correctly

received. The IS-95A Frame Quality Detector block outputs the Quality Indicator signal, as well as the information bits without the CRC bits. One Error Rate Calculation block compares the information bits to the bits generated at the source, while another Error Rate Calculation block compares the Quality Indicator bits with zero. Finally, the resultant bit and frame error rates are displayed.

Visible Results of the Demo

The demo displays the bit error rate (BER) and the frame error rate (FER) for the reverse channel coding and decoding operations. Both of these error rates show the protection that the codec offers against errors introduced in the channel. The error rates depend on the additive noise in the system as well as the data rate in the simulation. You can see the error rates change as you change the **Es/No** parameter in the AWGN Channel block, or the or the **Data Rate** parameter inside the Mobile Station Transmitter Data Rate masked subsystem.

Changing Demo Parameters

This section suggests some ways that you can alter the parameters in the demo after you understand the model.

Simulation Duration. The simulation duration in this demo is set to 10 seconds. Because the frame duration in IS-95A is 0.02 second, this model processes 500 frames. This gives a sufficient time for the BER measurements to converge at reasonable SNR settings. For lower SNR settings the FER may take a longer time to converge. To run the simulation for a longer or shorter time, select **Parameters** from the **Simulation** menu, and change the Stop time to 0.02 times the number of frames you want to simulate.

Data Rate. To change the data rate, double-click on the Mobile Station Transmitter Data Rate icon and change the **Data Rate** parameter. Choices are Full, Half, Quarter, and One-Eighth rates.

Rate Set. The **Rate set** parameter is Rate Set I by default in the simulation, but alternatively you may select Rate Set II. The **Rate set** parameter should be changed in all of the blocks by double-clicking on their icons and selecting either Rate Set I or Rate Set II from the **Rate set** parameter menu. The blocks that need to be changed to the same **Rate set** parameter are the IS-95A CRC

Generator, IS-95A Rev Ch Convolutional Encoder, IS-95A Rev Ch Viterbi Decoder, and IS-95A Frame Quality Detector.

Signal-to-Noise Ratio. Change the **Es/No** parameter in the AWGN Channel block. The detection performance improves with an increase in the signal-to-noise ratio.

Random Seed. To run the simulation with different seeds, change the **Initial seed** parameters in the Random Binary Frame Generator masked subsystem and/or the AWGN Channel block.

Further Explorations

The demo serves as the starting point for building simulation models for channel coding and decoding. It can be used for developing codec models for the Access channel. In addition, it is a good platform for experimenting with variations in channel decoding algorithms such as soft-decision quantization, and specialized implementations of the Viterbi decoder. In addition, you may combine this model with parts of the reverse channel detection model for a more elaborate simulation.

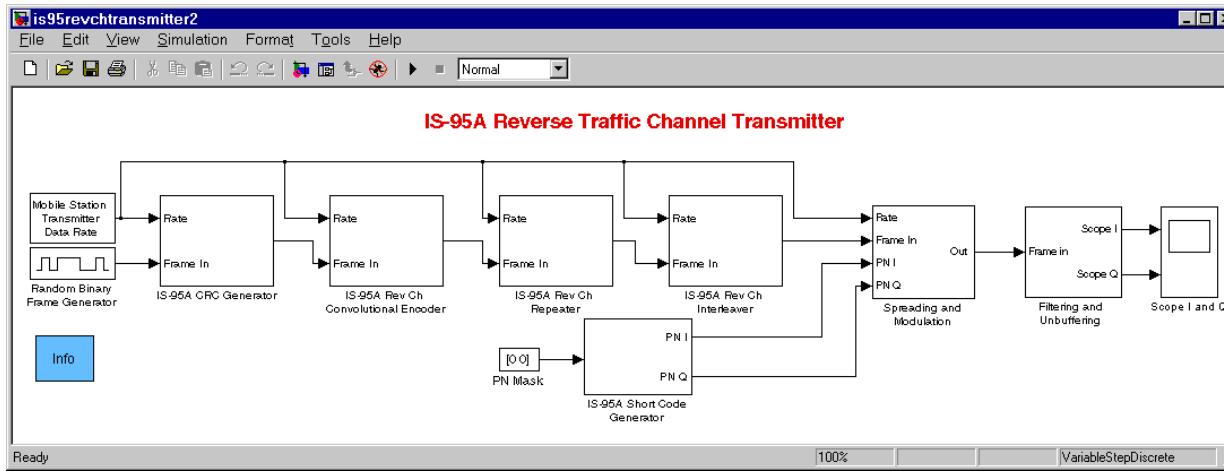
IS-95A Reverse Traffic Channel Transmitter Demo

The IS-95A Reverse Traffic Channel Transmitter demo simulates the mobile station transmitter, which is the transmitting section of the IS-95A reverse link. These are the main components of the demo:

- CRC generator
- Convolutional encoder
- Symbol repeater
- Interleaver
- Modulator and spreader
- Transmit filter

The mobile transmitter model is important when designing a receiver, because the ability to simulate the transmitted waveform and its characteristics, such as the peak-to-mean ratio, affects the power control components of the design.

You can open the model by following the instructions in “Opening a Demo” on page 1-23, or by typing `is95revchtransmitter2` at the MATLAB prompt.



Library Blocks in the Demo

The IS-95A Reverse Traffic Channel Transmitter demo uses these library blocks from the CDMA Reference Blockset:

- IS-95A CRC Generator
- IS-95A Rev Ch Convolutional Encoder
- IS-95A Rev Ch Interleaver/Deinterleaver
- IS-95A Rev Ch Repeater/Derepeater
- IS-95A Rev Ch Walsh Modulation and Spreading (inside Spreading and Modulation subsystem)
- IS-95A Short Code Generator

How the Demo Works

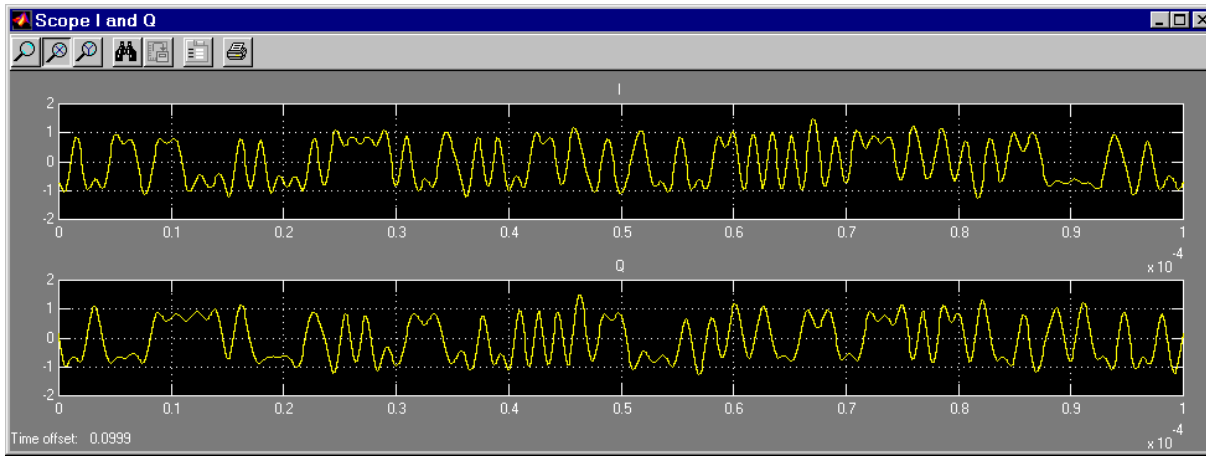
The Mobile Station Transmitter Data Rate masked subsystem provides the data rate, while the Random Binary Frame Generator masked subsystem generates random data bits for the system to transmit. The raw data corresponds to the speech frame that is generated at the vocoder. The IS-95A CRC Generator library block generates the CRC (cyclic redundancy check) bits and appends them to the frame data depending on the data rate. The result of the CRC addition is then convolutionally encoded for error protection by the

IS-95A Rev Ch Convolutional Encoder library block. Because the modulation symbol rate is constant regardless of the data rate, the IS-95A Rev Ch Repeater/Derepeater library block (denoted IS-95A Rev Ch Repeater in this demo) repeats the convolutionally encoded data by a factor depending on the data rate to bring the output symbols to a constant symbol rate. This data is then interleaved by the IS-95A Rev Ch Interleaver/Deinterleaver (denoted IS-95A Rev Ch Interleaver in this demo) for protection against error bursts. Meanwhile, the IS-95A Short Code Generator generates the short PN code, which is used for quadrature spreading.

The IS-95A Rev Ch Walsh Modulation and Spreading library block, which is inside the Spreading and Modulation subsystem, performs the Walsh modulation and spreading with the long code. The Spreading and Modulation subsystem also performs the short PN spreading. The Filtering and Unbuffering subsystem performs the pulse shaping to generate the I and Q waveforms that are displayed in the **Scope I and Q** window.

Visible Results of the Demo

The demo has a built-in Scope block (denoted Scope I and Q in this demo) to display the in-phase and quadrature components of the modulated waveform. The waveform quality may be examined using the display. Also when the rate is less than full rate, the displays show that the transmit waveform is gated off for part of the duration. Under the default parameter settings, the nature of the transmitter I and Q waveforms is shown below.



Changing Demo Parameters

This section suggests some ways that you can alter the parameters in the demo after you understand the model.

Simulation Duration. The simulation duration in this demo is set to 0.1 second. Because the frame duration in IS-95A is 0.02 seconds, this model processes five frames. To obtain long term statistical measurements of waveform properties, you need to increase the duration of the simulation. To do this, select **Parameters** from the **Simulation** menu, and change the Stop time to 0.02 times the number of frames you want to simulate.

Data Rate. To change the source data rate, double-click on the Mobile Station Transmitter Data Rate icon, and choose either Full, Half, Quarter, or One-Eighth rates from the menu.

Rate Set. Change the **Rate set** parameter in the IS-95A CRC Generator library block and the IS-95A Rev Ch Convolutional Encoder library block. To do this, double-click on these blocks' icons and choose either Rate Set I or Rate Set II from the **Rate set** menu.

Random Seed. The simulation uses a random seed to generate random data at the source. To run the simulation with a different random data sequence,

double-click on the Random Binary Frame Generator icon and change the **Initial seed** parameter.

Further Explorations

The demo serves as the starting point for building simulation models with different objectives relevant to a specific application. Parameters in the demo's components may be changed for different configurations. For instance, you can alter the channel type in all components (on the main level or within subsystems) that include a **Channel type** parameter, to obtain a simulation for a different configuration.

Other possible changes that you can make include:

- Adding components such as a specific digital-to-analog converter or power amplifiers, to evaluate their performance
- Evaluating waveform quality, for example, by measuring error vectors
- Replacing the transmit filters with blocks from the Signal Processing Blockset or adding extra filters to model specific analog filter components
- Introducing performance measurement blocks to obtain statistics for the waveform properties

In addition, this transmitter may be used for design and evaluation of base-station receiver algorithms for objectives such as acquisition.

Selected Bibliography

[1] *ANSI/J-STD-008: Personal Station Base-Station Compatibility Requirements for 1.9 to 2.0 GHz Code Division Multiple Access (CDMA) Personal Communications Systems.* New York: American National Standards Institute, 1998.

[2] Lee, Jhong Sam and Leonard E. Miller. *CDMA Systems Engineering Handbook*. Boston, Mass.: Artech House, 1998.

[3] *TIA/EIA Interim Standard 95-A: Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System*. Arlington, VA: Telecommunications Industry Association, 1995.

[4] Viterbi, Andrew J. *CDMA Principles of Spread Spectrum Communications*. Reading, Mass.: Addison-Wesley, 1995.

2

Block Reference

Blocks — Categorical List

The summary tables below list the blocks in the CDMA Reference Blockset by library and give a brief description of the purpose of each block. Following these are individual reference pages, in alphabetical order, providing detailed descriptions of each block:

- IS-95A Base Station Transmitter Library
- IS-95A Mobile Station Receiver Library
- IS-95A Mobile Station Transmitter Library
- IS-95A Base Station Receiver Library
- IS-95A Common Library
- Alphabetical List of Blocks

IS-95A Base Station Transmitter Library

Block	Purpose
IS-95A Fwd Ch Base Station Transmitter Interface	Combine various control and user data using different Walsh code sequences
IS-95A Fwd Ch Convolutional Encoder	Convolutionally encode the input data frame
IS-95A Fwd Ch Interleaver/ Deinterleaver	Interleave or deinterleave the symbols of the incoming frame
IS-95A Fwd Ch Repeater/ Derepeater	Repeat, puncturing if necessary, or derepeat the symbols of the encoded frame
IS-95A Fwd Ch Scrambler	Scramble the data on the Paging and forward Traffic channels by the decimated long code and insert power control bits into the Traffic channel

IS-95A Mobile Station Receiver Library

Block	Purpose
IS-95A Fwd Ch Descrambler	Perform power bit extraction, rake combining, descrambling, and frame buffering
IS-95A Fwd Ch Detector	Perform despreading, demodulation, and rake combining
IS-95A Fwd Ch Power Bit Extractor	Extract power bits from the received data
IS-95A Fwd Ch Rake Demodulator	Demodulate the data obtained from the rake receiver fingers
IS-95A Fwd Ch Rake Finger	Correlate the input signal over each Walsh code interval with the short PN code and Walsh code sequences
IS-95A Fwd Ch Viterbi Decoder	Decode a convolutionally encoded information sequence optimally

IS-95A Mobile Station Transmitter Library

Block	Purpose
IS-95A Rev Ch Burst Randomizer	Generate a pseudorandom masking pattern of zeros and ones that can be used to mask redundant power groups
IS-95A Rev Ch Convolutional Encoder	Convolutionally encode the input data frame
IS-95A Rev Ch Interleaver/ Deinterleaver	Interleave or deinterleave a data frame for the Access channel or the reverse Traffic channel
IS-95A Rev Ch Repeater/ Derepeater	Repeat or derepeat the symbols of the input frame for the reverse Access and Traffic channels based on the data rate
IS-95A Rev Ch Walsh Modulation and Spreading	Perform Walsh code modulation, spreading with the long code, and power group randomization
IS-95A Rev Ch Walsh Modulator	Modulate the input data sequence with the Walsh codes

IS-95A Base Station Receiver Library

Block	Purpose
IS-95A Rev Ch Detector	Perform noncoherent detection of the input data frame
IS-95A Rev Ch Rake Finger	Despread the received signal with the short PN and long codes, and then correlate the despread signal with all of the Walsh code sequences
IS-95A Rev Ch Short Code Despreader	Despread the input sequence with the I and Q short PN codes
IS-95A Rev Ch Viterbi Decoder	Decode a convolutionally encoded information sequence optimally for the Access and reverse Traffic channels
IS-95A Rev Ch Walsh Correlator	Correlate the input sequence with a bank of Walsh sequences for the reverse Access and Traffic channels
IS-95A Rev Ch Walsh Demodulator	Perform Walsh demodulation of the combined output of the rake fingers

IS-95A Common Library

Block	Purpose
IS-95A CRC Generator	Generate Cyclic Redundancy Check (CRC) bits and append them along with tail bits to an input data frame
IS-95A Frame Quality Detector	Determine the quality of a received frame of data and output the decoded data and a frame quality indicator
IS-95A Long Code Generator	Generate the long code sequence used for reverse channel spreading and scrambling and for forward channel scrambling
IS-95A Short Code Generator	Generate the I and Q short PN code sequences
IS-95A Syndrome Detector	Check the CRC bits to determine the frame quality, and output the decoded data and frame quality decision
IS-95A Walsh Code Generator	Generate the Walsh code sequences corresponding to the specified Walsh order and indices

Functions — Alphabetical List

IS-95A CRC Generator

Purpose

Generate Cyclic Redundancy Check (CRC) bits and append them along with tail bits to an input data frame

Library

IS-95A Common

Description

This block generates the Cyclic Redundancy Check (CRC) bits in accordance with the IS-95A standard. The block then appends the CRC bits and tail bits to the input information bits to generate the output frame. The generator polynomial used to generate the CRC bits is specified in the IS-95A standard.

The number of relevant information bits may vary from frame to frame, depending on the type of channel and the data rate; therefore, the size of the Frame In port has been set to the maximum size of 268 (Rate Set II, Traffic Full Rate). Also, the number of bits added depends on the channel, rate set, and rate. Hence, the output port size has been set to the maximum value of 288 (Rate Set II, Traffic Full Rate). The numbers for the relevant input and output bits are specified in the table below.

The CRC and tail bits immediately follow the relevant information bits. The tail bits are used in the decoding of the convolutionally coded data at the receiver. These are all zeros and are added only in the case of the Traffic and Access channels. The bits of the 288 that are output and that do not correspond to the relevant bits for the data rate in use, are cleared to zero.

Channel Type	Number of Relevant Input Bits	Number of CRC Bits	Number of Tail Bits	Number of Relevant Output Bits
Sync, Eighth Rate	32	0	0	32
Paging, Full Rate	192	0	0	192
Paging, Half Rate	96	0	0	96
Access	88	0	8	96
Traffic, Rate Set I, Full Rate	172	12	8	192
Traffic, Rate Set I, Half Rate	80	8	8	96

Channel Type	Number of Relevant Input Bits	Number of CRC Bits	Number of Tail Bits	Number of Relevant Output Bits
Traffic, Rate Set I, Quarter Rate	40	0	8	48
Traffic, Rate Set I, Eighth Rate	16	0	8	24
Traffic, Rate Set II, Full Rate	268	12	8	288
Traffic, Rate Set II, Half Rate	126	10	8	144
Traffic, Rate Set II, Quarter Rate	56	8	8	72
Traffic, Rate Set II, Eighth Rate	22	6	8	36

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1

Channel Type	Data Rate (bps)	Input Value
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

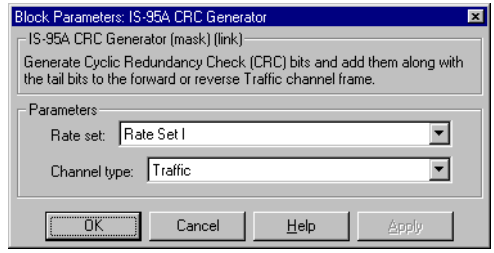
Frame In

Binary vector of size 268 that contains the input frame to which the block appends the CRC and tail bits.

Outputs

Binary vector of size 288. The relevant bits of the output include the input bits, the CRC bits, and the tail bits, in that order.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The forward or reverse channel type, either Sync, Paging, Access, or Traffic.

See Also

IS-95A Frame Quality Detector
IS-95A Syndrome Detector
IS-95A Reverse Traffic Channel Transmitter Demo

IS-95A Forward Traffic Channel Codec Demo
IS-95A Reverse Traffic Channel Codec Demo

Specification References

IS-95A 6.1.3.3.2, 7.1.3.5.2
J-STD-008 2.1.3.3.2.1, 3.1.3.5.2.1

IS-95A Frame Quality Detector

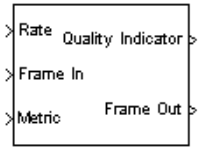
Purpose

Determine the quality of a received frame of data and output the decoded data and a frame quality indicator

Library

IS-95A Common

Description



This is a hierarchical block that determines if there are any frame errors using a syndrome detection procedure or via a metric comparison procedure for the Access and Traffic channels. In the case of the Sync and Paging channels, this block does not determine frame errors since the data is encoded over multiple frames.

The syndrome detection procedure uses the IS-95A Syndrome Detector block to compute a syndrome for the CRC encoded received data frame. A zero value for the syndrome indicates a correct frame has been received. The syndrome detection procedure is applicable to the Access channel and the Traffic channel full and half rates in Rate Set I and all rates in Rate Set II. When the block uses the syndrome detection procedure, it also removes the CRC and tail bits.

The metric comparison procedure is applicable to the quarter and eighth rate Traffic channel in Rate Set I where a CRC check is not provided in the frame. In this case, the metric detection procedure compares the soft-value metric at the output of the Viterbi decoder to a threshold. If the metric value equals or exceeds the threshold, it is assumed that the decoded frame is received correctly.

The Frame In port size is equal to the maximum possible size of 288 (Rate Set II, Traffic Full Rate). Similarly, the Frame Out port size is equal to the maximum possible size of 268 (Rate Set II, Traffic Full Rate). The following table gives the number of relevant output bits, in a frame, for all channel types. The block sets to zero any unused output bits. The CRC and tail bits immediately follow the relevant information bits.

Channel Type	Number of Relevant Inputs Bits	Number of CRC Bits	Number of Tail Bits	Number of Relevant Output Bits
Sync, Eighth Rate	32	0	0	32
Paging, Full Rate	192	0	0	192

Channel Type	Number of Relevant Inputs Bits	Number of CRC Bits	Number of Tail Bits	Number of Relevant Output Bits
Paging, Half Rate	96	0	0	96
Access	96	0	8	88
Traffic, Rate Set I, Full Rate	192	12	8	172
Traffic, Rate Set I, Half Rate	96	8	8	80
Traffic, Rate Set I, Quarter Rate	48	0	8	40
Traffic, Rate Set I, Eighth Rate	24	0	8	16
Traffic, Rate Set II, Full Rate	288	12	8	268
Traffic, Rate Set II, Half Rate	144	10	8	126
Traffic, Rate Set II, Quarter Rate	72	8	8	56
Traffic, Rate Set II, Eighth Rate	36	6	8	22

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0

Channel Type	Data Rate (bps)	Input Value
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Frame In

Real vector of size 288 containing the input frame with the CRC and tail bits.

Metric

Real scalar representing the final soft metric value computed in the IS-95A Fwd Ch Viterbi Decoder using the Viterbi algorithm.

Outputs

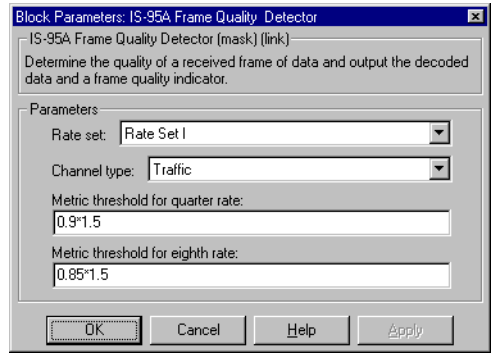
Quality Indicator

Integer scalar that indicates the decision as to whether the frame was in error. A value of 0 indicates no error and 1 indicates an error. In the cases of Sync, Paging, and Access channels, the value is set to -1.

Frame Out

Binary vector of size 268 representing the data frame after removing CRC and tail bits, if any.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Sync, Paging, Access, or Traffic.

Metric threshold for quarter rate

Real scalar that specifies the threshold value against which the metric from the IS-95A Fwd Ch Viterbi Decoder is compared for the Traffic channel quarter rate frame. This is only applicable in Rate Set I.

Metric threshold for eighth rate

Real scalar that specifies the threshold value against which the metric from the IS-95A Fwd Ch Viterbi Decoder is compared for the Traffic channel eighth rate frame. This is only applicable in Rate Set I.

See Also

IS-95A CRC Generator
IS-95A Syndrome Detector
IS-95A Forward Traffic Channel Codec Demo
IS-95A Reverse Traffic Channel Codec Demo

Specification References

IS-95A 6.1.3.3.2, 7.1.3.5.2,
J-STD-008 2.1.3.3.2.1, 3.1.3.5.2.1

IS-95A Fwd Ch Base Station Transmitter Interface

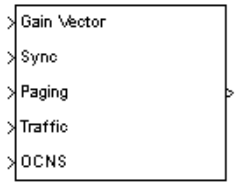
Purpose

Combine various control and user data using different Walsh code sequences

Library

IS-95A Base Station Transmitter

Description



This block encodes the Pilot, Sync, Paging, Traffic, and the Orthogonal Channel Noise Simulator (OCNS) symbols using the specified Walsh code sequences. The encoded symbols for each channel are then gain-adjusted and added before they are output. The Pilot symbol value is always equal to one, whereas the symbols for the other channels are provided as inputs to this block.

Inputs

Gain Vector

Real vector of size five representing gains to be applied to the Pilot, Sync, Paging, Traffic, and OCNS symbols.

Sync

Real scalar of bipolar data representing the Sync channel data symbols.

Paging

Real scalar of bipolar data representing the Paging channel data symbols.

Traffic

Real scalar of bipolar data representing the Traffic channel data symbols after scrambling.

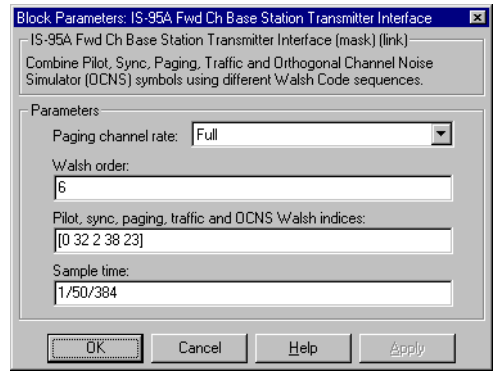
OCNS

Real scalar of bipolar data representing the OCNS data symbols.

Outputs

Real vector representing the combination of all channel data symbols encoded by various Walsh code sequences. If the **Walsh order** parameter is W, then the output vector size equals 2^W. For IS-95A, the Walsh Order is 6, and hence the size of this vector is 64.

Dialog Box



Parameters

Paging channel rate

The rate fraction for the Paging channel, either Half or Full.

Walsh order

Integer scalar that specifies the order of the Walsh code used to encode each data symbol. The default value, 6, corresponds to a Walsh code length of 64.

Pilot, sync, paging, traffic and OCNS Walsh indices

Integer vector of size five that specifies the Pilot, Sync, Paging, Traffic, and OCNS channels' Walsh indices.

Sample time

Real scalar that specifies the block sample time.

See Also

IS-95A Walsh Code Generator
IS-95A Fwd Ch Scrambler
IS-95A Forward Traffic Channel Detection Demo

IS-95A Fwd Ch Convolutional Encoder

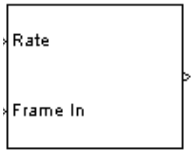
Purpose

Convolutionally encode the input data frame

Library

IS-95 Base Station Transmitter

Description



This block convolutionally encodes the input data frame. For the Sync and Paging channels, the data frame consists only of the information bits. For the Traffic channel, the data frame includes the information data, the CRC bits and the tail bits in accordance with the IS-95A specification. Note that as a result, the last 8 bits of each Traffic channel input frame must be 0.

The convolutional encoder is a rate 1/2 encoder having a constraint length of nine. The generator functions are 753 (octal) and 561 (octal). The state of the convolutional encoder, upon initialization, is the all-zeros state. For this block, the Frame In port size is equal to the maximum possible size of 288 (Rate Set II, Traffic Full Rate). Similarly, the output port size is equal to the maximum possible size of 576 (Rate Set II, Traffic Full Rate). The following table shows the number of relevant bits in the input and output, which depends on the channel type and the data rate. The block ignores input bits beyond the relevant bits, and sets output bits beyond the relevant bits to zero.

Channel Type	Number of Relevant Input Bits	Number of Relevant Output Bits
Sync, Eighth Rate	32	64
Paging, Full Rate	192	384
Paging, Half Rate	96	192
Traffic, Rate Set I, Full Rate	192	384
Traffic, Rate Set I, Half Rate	96	192
Traffic, Rate Set I, Quarter Rate	48	96
Traffic, Rate Set I, Eighth Rate	24	48

Channel Type	Number of Relevant Input Bits	Number of Relevant Output Bits
Traffic, Rate Set II, Full Rate	288	576
Traffic, Rate Set II, Half Rate	144	288
Traffic, Rate Set II, Quarter Rate	72	144
Traffic, Rate Set II, Eighth Rate	36	72

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Rate Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1

Channel Type	Data Rate (bps)	Rate Input Value
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

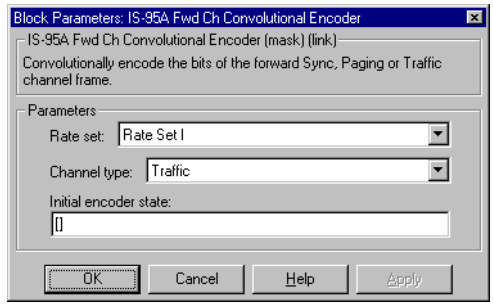
Frame In

Real binary vector of size 288 containing the input frame, which consists of information bits, CRC, tail bits, and zeros.

Outputs

Binary vector of size 576 containing the convolutionally encoded output frame.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Sync, Paging, or Traffic.

Initial encoder state

Integer scalar that specifies the convolutional encoder's initial state. The default value is between 0 and 511. Empty brackets indicate the default value of 0.

See Also

IS-95A Fwd Ch Viterbi Decoder
IS-95A Rev Ch Convolutional Encoder
IS-95A Rev Ch Viterbi Decoder
IS-95A Forward Traffic Channel Codec Demo

Specification References IS-95A 7.1.3.1.3
J-STD-008 3.1.3.1.3

IS-95A Fwd Ch Descrambler

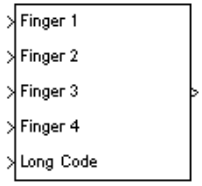
Purpose

Perform power bit extraction, rake combining, descrambling, and frame buffering

Library

IS-95 Mobile Station Receiver

Description



This is a hierarchical block performing power bit extraction, rake combining, descrambling and frame buffering. It extracts the power bit for each finger using the long code and combines the signals from all fingers. The extracted power bit is updated once every power group, that is, 16 times every 20 ms frame. There are two symbols carrying the power control bit in Rate Set I and only one symbol in Rate Set II.

After power bit extraction the block sets to zero the symbol(s) corresponding to the power control bit. The block subsequently descrambles the combined signal using the decimated long code sequence. It stores the descrambled data in a buffer to obtain one complete frame.

Inputs

Finger 1

Real scalar representing the demodulated symbol for finger 1.

Finger 2

Real scalar representing the demodulated symbol for finger 2.

Finger 3

Real scalar representing the demodulated symbol for finger 3.

Finger 4

Real scalar representing the demodulated symbol for finger 4.

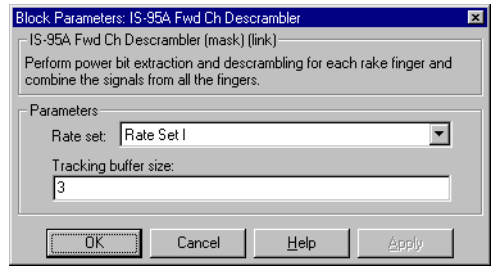
Long Code

Real scalar of binary data representing the decimated long code value for the symbols to be descrambled.

Outputs

Real vector of size 384 representing the combined and descrambled data frame.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Tracking buffer size

Integer scalar that specifies the length of the buffer (in number of symbols) needed in the rake fingers. The block uses this parameter to adjust its delay with respect to the frame boundary.

See Also

IS-95A Fwd Ch Scrambler
IS-95A Long Code Generator
IS-95A Fwd Ch Power Bit Extractor

IS-95A Fwd Ch Detector

Purpose

Perform despreading, demodulation, and rake combining

Library

IS-95A Mobile Station Receiver

Description

This is a hierarchical block consisting of four rake fingers, coherent rake demodulator, descrambler, and short and the long PN code generators. This block performs the front-end forward channel receiver functions to detect the received signal. For each finger, this block downsamples the received signal, despreads it with the Walsh sequence, correlates it with the short PN code, and estimates the pilot channel multipath strength at the phase assigned to the finger. Using the channel estimates for each finger, this block demodulates the received signal, extracts the power bits, and combines the signals from all of the rake fingers to represent the soft decision detected symbols whose value is scrambled with the decimated long code. Finally, this block buffers and outputs a frame of detected soft-decision symbols after descrambling them with the decimated long code.

Inputs

Walsh Seq

Real vector of bipolar data representing the Walsh code sequence. If the **Walsh order** parameter is W, then the Walsh Seq input vector size is 2^W.

Rx Signal

Complex vector representing the oversampled input signal. The vector size equals the Walsh Seq input vector size times the **Oversampling rate** parameter times the **Input frame size** parameter.

Path Delay/Enable

Integer vector of size eight containing a pair of elements for each rake finger. The first element in each pair represents the initial PN phase offset that is applied to each rake finger. The second element in each pair is an enable signal for the rake finger. A value of 0 disables the finger and a value of 1 enables the finger.

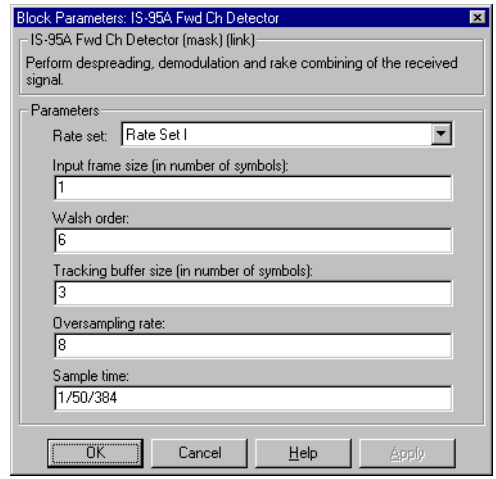
Short PN Mask

Integer vector of size two representing the masks for the in-phase and quadrature components of the short PN code. The masks instantaneously change the phase value of the short PN code generator output.

Outputs

Real vector of size 384 representing an output frame of detected symbols.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Walsh order

Integer scalar that specifies the order of the Walsh code used to encode each data symbol.

Input frame size (in number of symbols)

Integer scalar that specifies the number of symbols to be demodulated in each execution of the block.

Oversampling rate

Integer scalar that specifies the number of samples per chip. The default value, 8, corresponds to eight samples in a chip interval.

Tracking buffer size (in number of symbols)

Integer scalar that specifies the buffer length (in number of symbols) required to correct for differences in path delays of the received multipath rays. The length of this buffer sets a limit on the maximum phase difference between the received rays.

Sample time

Real scalar that specifies the block sample time.

Algorithm

The signal received by the rake receiver module of the mobile can be modeled by

$$y(t) = \sum_{l=1}^{L} \sum_{j=1}^{J} [\alpha(k,l)p(k)b(k,j)u(t - kT - \tau(l))] + z(t) + \varepsilon(t)$$

where:

- k is the time index
- t is between kT and $(k+1)T$
- T is the period of a chip (1/1228800 second)
- L is the number of propagation delay paths
- J is the number of users on the same CDMA channel
- $\alpha(k,l)$ is the equivalent complex gain of the l-th delay path of the propagation channel at sample k
- $\tau(l)$ is the delay of path l
- $\{b(k,j)\}$ is the jth user's Walsh-spread sequence
- $p(k)$ is the short (complex) PN sequence that identifies the base station

Furthermore, $z(t)$ models the interference generated by other CDMA base stations, while ε(t) models the thermal noise, as well as the existing narrow-band interference.

$b(k,j)$, in turn, can be expanded as

$$b(k,j) = s(n,j)w(m,j)$$

where:

- $m = k \pmod{64}$
- $n = (k-m)/64$
- $\{w(m,j)\}$ is the Walsh sequence
- $\{s(n,j)\}$ is the sequence of bipolar encoded data bits sent by the base station to the jth user/channel

Because different Walsh sequences are orthogonal and the short PN sequence has zero cross-correlation over its period, the optimum way to detect the transmitted data bits $\{s(n,j)\}$ sent over the propagation-delay path l, is to

multiply the received signal by $p^*(k)w(m,j)u(t\text{-}kT\text{-}\tau(l))$ (where p^* is the conjugate of p) and integrate over the Walsh sequence period, that is, 64 chips. This, in fact, is the principle behind the construction of the rake receiver. After synchronizing each finger to a different propagation-delay path, the output of lth finger for the jth user (or channel), is then given by

$$r(n,l,j) = \frac{1}{64}\sum_{k=64n}^{64(n+1)}[\alpha(k,l)]s(n,j) + \sum_{d\neq l} I(n,d) + Z(n) + E(n)$$

where $I(n,d)$, $Z(n)$, and $E(n)$ are respectively the residual terms due to transmission through other delay paths, interference from other base stations, and noise. If one could integrate over the whole period of the PN sequence, then all the residual terms would be close to zero (assuming that the transmitted bits spread by the Walsh sequence have white noise characteristics). However, due to the limited integration length, these terms have small but nevertheless nonzero values.

Although none of the three above-mentioned interference terms is statistically independent of the first term, for all practical purposes one can assume that they behave like white noise. Moreover, because the rate of variation of propagation channel characteristics, that is, $\alpha(k,l)$, is in the order of the Doppler frequency and is much smaller than the symbol rate (19.2 kbps), one can assume in practice that $\alpha(k,l)$ is constant during the integration period equal to 64 chips or one symbol length. Hence, one can approximate the above equation with

$$r(n,l,j) = \alpha(n,l)s(n,j) + \beta(n,l,j)e(n,l,j)$$

where $e(n,l,j)$ is assumed to be a white noise sequence and $\beta(n,l,j)$ reflects the variations of the propagation channel characteristics, and therefore, is slowly varying.

The objective of the demodulator, therefore, is to calculate the optimum estimate

$$\hat{s}(n,j,l)$$

(or soft-detection) of $s(n,j)$ for each finger, and to combine these estimates so as to maximize the resulting signal-to-interference ratio.

To combine the demodulated data from different rake fingers, simply add the soft-detected symbols together. In other words, the symbol representing soft detection of $s(n,j)$ is given by

$$\hat{s}(n,j) = \sum_{l=1}^{L} \hat{s}(n,j,l) = \sum_{l=1}^{L} \mathrm{Re}(r(n,l,j)H^{*}(n,l))$$

where L is the number of rake fingers.

This is based on the assumption that the distribution functions of the rake finger outputs are independent, which means that the joint log-likelihood function of the outputs is simply the sum of the individual log-likelihood functions. If the time-alignments of different fingers are sufficiently far apart, then this is a valid assumption.

See Also

IS-95A Fwd Ch Rake Demodulator
IS-95A Fwd Ch Rake Finger
IS-95A Forward Traffic Channel Detection Demo

Specification References

IS-95A 7.1.3.1.8, 7.1.3.1.9
J-STD-008 3.1.3.1.9, 3.1.3.1.10

Purpose

Interleave or deinterleave the symbols of the incoming frame

Library

IS-95 Base Station Transmitter

Description

This block interleaves or deinterleaves the symbols of the incoming frame. The Paging and Traffic channel interleaver time span is 20 ms equivalent to 384 symbols. The Sync channel, however, uses an interleaver spanning 26.67 ms equivalent to 128 symbols. The interleaver writes the code symbols into the matrix on a column-by-column basis for all of the input data symbols and reads the stored data into the output buffer in accordance with the IS-95A specification. The deinterleaver restores the interleaved symbols into the original order.

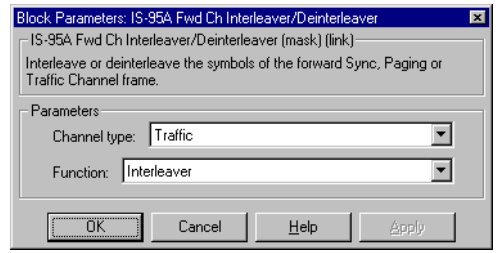
Inputs

Real vector (of binary data for interleaver functionality) that represents the data frame to be interleaved or deinterleaved. The size of the input is 384 for the Paging and Traffic channels and 128 for the Sync channel.

Outputs

Real vector (of binary data for interleaver functionality) that represents the symbols of the interleaved or deinterleaved output frame. The size of the input is 384 for the Paging and Traffic channels and 128 for the Sync channel.

Dialog Box



Parameters

Channel type

The channel type, either Sync, Paging, or Traffic.

Function

Interleaver or Deinterleaver.

IS-95A Fwd Ch Interleaver/Deinterleaver

See Also

IS-95A Rev Ch Interleaver/Deinterleaver
IS-95A Fwd Ch Repeater/Derepeater
IS-95A Forward Traffic Channel Codec Demo

Specification References

IS-95A 6.1.3.1.5, 7.1.3.1.5
J-STD-008 2.1.3.1.5, 3.1.3.1.6

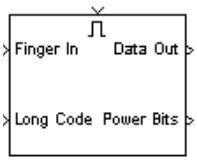
Purpose

Extract power bits from the received data

Library

IS-95A Mobile Station Receiver

Description



This block extracts the power control bit(s) from the received scrambled data of a rake finger output. The long code bits used for scrambling the data in the previous frame determine the position of power control bit(s) in each power group (a group of 24 symbols) according to the IS-95A specification. The decimated long code in the present power group determines the power bit position(s) in the following power group. For the initial power group, the **Initial power bit location** parameter sets the position of the power control bits. The power control bits for Rate Set I are combined before output is produced, and the punctured power control bit position values are set to zero in the data stream.

Inputs

Enabler (on top)

A positive value at the top port enables the block.

Finger In

Real vector representing the demodulated data.

Long Code

Binary vector representing the decimated long code used in determining the power bit position(s) in each power group.

Outputs

Data Out

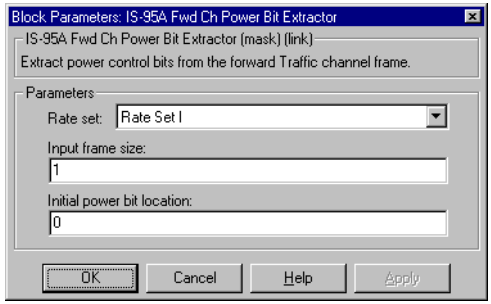
Real vector representing the demodulated data after puncturing the power bit symbols.

Power Bits

Real vector representing the power control bit for each power group.

IS-95A Fwd Ch Power Bit Extractor

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Input frame size

Integer scalar that specifies the number of symbols in the Finger In input. The default value, 1, represents symbol-by-symbol demodulation.

Initial power bit location

Integer scalar that specifies the initial position of the power bit.

See Also

IS-95A Fwd Ch Scrambler
IS-95A Fwd Ch Descrambler

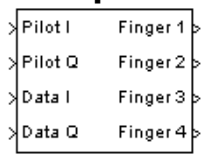
Purpose

Demodulate the data obtained from the rake receiver fingers

Library

IS-95A Mobile Station Receiver

Description



This block demodulates the data obtained from the rake receiver fingers. The block operates on data from four fingers of a rake receiver and generates the demodulated symbols for each finger. The demodulation process utilizes the channel estimate or pilot signal for each finger to demodulate the symbols for each finger.

Inputs

Pilot I

Real vector of size 4***Input frame size per finger*** representing the in-phase component of the channel estimates for the four rake fingers.

Pilot Q

Real vector of size 4***Input frame size per finger*** representing the quadrature component of the channel estimates for the four rake fingers.

Data I

Real vector of size 4***Input frame size per finger*** representing the in-phase component of the correlator output for the four rake fingers.

Data Q

Real vector of size 4***Input frame size per finger*** representing the quadrature component of the correlator output for the four rake fingers.

Outputs

Finger 1

Real vector of size ***Input frame size per finger*** representing the demodulated symbol data for finger 1.

Finger 2

Real vector of size ***Input frame size per finger*** representing the demodulated symbol data for finger 2.

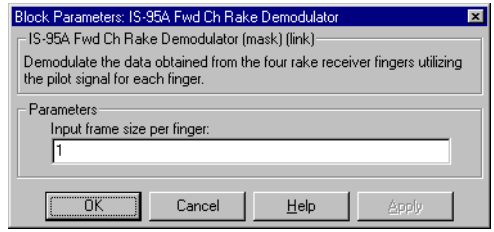
Finger 3

Real vector of size ***Input frame size per finger*** representing the demodulated symbol data for finger 3.

Finger 4

Real vector of size **Input frame size per finger** representing the demodulated symbol data for finger 4.

Dialog Box



Parameters

Input frame size per finger

Integer scalar that specifies the number of demodulation symbols per finger.

See Also

IS-95A Fwd Ch Detector

Specification References

IS-95A 7.1.3.1.8, 7.1.3.1.9
J-STD-008 3.1.3.1.9, 3.1.3.1.10

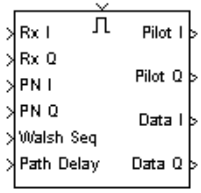
Purpose

Correlate the input signal over each Walsh code interval with the short PN code and Walsh code sequences

Library

IS-95A Mobile Station Receiver

Description



This block correlates the input signal over each Walsh code interval with the supplied PN and Walsh code sequences to estimate the channel and the data from the received signal. Using the known Walsh sequence for pilot symbols, this block estimates the in-phase and quadrature components of the channel. The input Walsh sequence, used to encode the data symbols, is used to estimate the in-phase and quadrature components of the received data.

The input signal is oversampled at eight times the chip rate, which is known as the tick rate. Initial PN phases are used to set the appropriate downsampling phase and timing for the delayed incoming multipath rays.

Inputs

Enabler (on top)

A positive value at the top port enables the block.

Rx I

Real vector representing the oversampled in-phase component of the input signal. The vector size is the Walsh Seq input vector size times the **Oversampling rate** parameter times the **Input frame size** parameter.

Rx Q

Real vector representing the oversampled quadrature component of the input signal. The vector size is the same as that of the Rx I input vector.

PN I

Real vector of bipolar data representing the in-phase component of the short PN code sequence. The vector size is the Walsh Seq input vector size times the **Input frame size** parameter.

PN Q

Real vector of bipolar data representing the quadrature component of the short PN code sequence. The vector size is the same as that of the PN Q input vector.

Walsh Seq

Real vector of bipolar data representing the Walsh code sequence. Its vector size is 2^W, where W is the **Walsh order** parameter.

Path Delay

Integer scalar representing the rake finger PN phase, in samples.

Outputs

Pilot I

Real vector of size **Input frame size** representing the in-phase component of the channel estimate.

Pilot Q

Real vector of size **Input frame size** representing the quadrature component of the channel estimate.

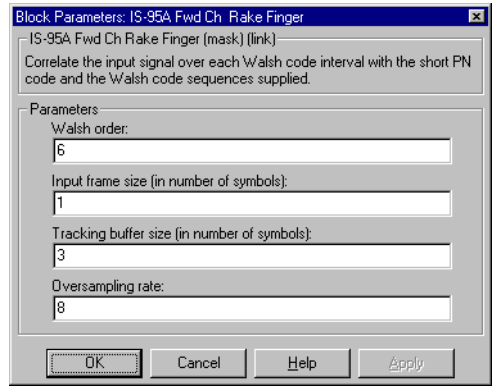
Data I

Real vector of size **Input frame size** representing the in-phase component of the output data symbol.

Data Q

Real vector of size **Input frame size** representing the quadrature component of the output data symbol.

Dialog Box



Parameters

Walsh order

Integer scalar that specifies the order of the Walsh code used to encode each data symbol. The default value, 6, corresponds to a Walsh code length of 64.

Input frame size (in number of symbols)

Integer scalar that specifies the number of symbols at the input of the rake correlator.

Tracking buffer size (in number of symbols)

Integer scalar that specifies the size of the buffer (in number of symbols) used to correct for differences in path delays of the received multipath rays. The length of this buffer limits the maximum phase difference between the received rays.

Oversampling rate

Integer scalar that specifies the number of samples per chip.

See Also

IS-95A Fwd Ch Detector

Specification References

IS-95A 7.1.3.1.8, 7.1.3.1.9
J-STD-008 3.1.3.1.9, 3.1.3.1.10

IS-95A Fwd Ch Repeater/Derepeater

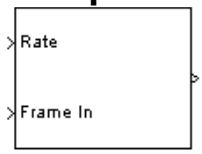
Purpose

Repeat, puncturing if necessary, or derepeat the symbols of the encoded frame

Library

IS-95 Base Station Transmitter

Description



This block repeats or derepeats the convolutionally encoded frame symbols for the forward channel. While repeating, the block also punctures the data if necessary.

For the Paging and Traffic channels and in repeater operation mode, this block repeats the convolutionally encoded symbols prior to block interleaving whenever the information rate is lower than full rate. In the half rate, quarter and one-eighth data rate modes of operation, each symbol repeats once, three and seven times, respectively. In Rate Set I, the repetition operation results in a frame size of 384 symbols for all data rates. For Rate Set II, the repetition operation results in a frame size of 576 symbols for all data rates, but the repeated data is punctured before leaving the output port. The puncturing pattern deletes the third symbol in a group of three symbols while retaining the first and the second symbols. After puncturing, the output frame size is 384 symbols.

For the Sync channel in repeater operation mode, the block repeats each symbol once. Therefore, each output frame consists of 128 symbols.

The derepeater inserts zeros in the positions of the punctured symbols and averages over the repeated data symbols.

For repeater operation mode, the input and output frame sizes equal 576 and 384, respectively, in Paging and Traffic channels. For the Sync channel, the input frame size is 576 and the output frame size is 128.

For derepeater operation mode, the output port size is the maximum possible size of 576 (Rate Set II, Traffic Full Rate). The following table shows the number of relevant bits at the input of the repeater and output of the derepeater, for each channel type and data rate.

Channel Type	Number of Relevant Bits for Repeater Input and Derepeater Output
Sync	64
Paging, Full Rate	384

Channel Type	Number of Relevant Bits for Repeater Input and Derepeater Output
Paging, Half Rate	192
Traffic, Rate Set I, Full Rate	384
Traffic, Rate Set I, Half Rate	192
Traffic, Rate Set I, Quarter Rate	96
Traffic, Rate Set I, Eighth Rate	48
Traffic, Rate Set II, Full Rate	576
Traffic, Rate Set II, Half Rate	288
Traffic, Rate Set II, Quarter Rate	144
Traffic, Rate Set II, Eighth Rate	72

Inputs

Rate

Integer scalar that specifies the data rate for the input frame. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3

Channel Type	Data Rate (bps)	Input Value
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

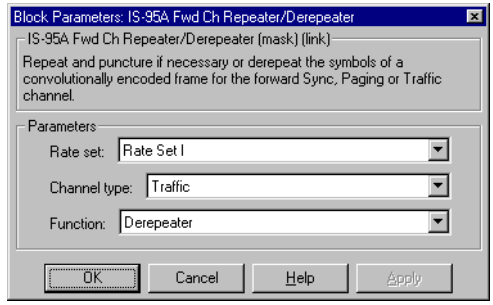
Frame In

For repeater operation mode, this is a binary vector of size 576 representing the convolutionally encoded data. For derepeater operation mode, this is a real vector of size 384.

Outputs

For repeater operation mode, this is a binary vector of size 384 (128 for the Sync channel) representing the output frame. For derepeater operation mode, this is a real vector of size 576.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Sync, Paging, or Traffic.

Function

Repeater or Derepeater.

See Also

IS-95A Rev Ch Repeater/Derepeater
IS-95A Forward Traffic Channel Codec Demo

Specification References

IS-95A 7.1.3.1.4
J-STD-008 3.1.3.1.4

IS-95A Fwd Ch Scrambler

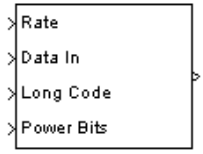
Purpose

Scramble the data on the Paging and forward Traffic channels by the decimated long code and insert power control bits into the Traffic channel

Library

IS-95A Base Station Transmitter

Description



This block scrambles and inserts power control bits into the input data stream. The scrambling function is applicable to Paging and forward Traffic channels. The power bit insertion, which applies only to the Traffic channel, occurs at an 800 Hz data rate. For the case of the Sync channel, this block does not scramble or insert power bits; instead, it merely repeats the input symbol four times to operate the Sync channel output at 19.2 ksps.

To scramble the Frame In data, this block first combines it with the decimated Long Code input data modulo two, and then replaces some bits with the Power Bits input data. The Traffic channel Rate Set I and Rate Set II include two and one power control bits, respectively, per power group (a group of 24 symbols). In case of Rate Set I, the power bits are inserted in two consecutive symbol locations. The last four decimated long code bits in a power group determine the power bit insertion position in the next power group.

Finally, this block adjusts the power level for the Paging and Traffic channel data bits, to take into account the power levels for rates lower than full rate, before they are output.

This block operates at the sample time of the Data In signal. The Power Bits input signal has a slower sample time.

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1

Channel Type	Data Rate (bps)	Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Data In

Binary scalar representing the frame symbol to be scrambled.

Long Code

Real scalar of bipolar data representing the decimated long code bit used for scrambling of the input data and determining the power bit location.

Power Bits

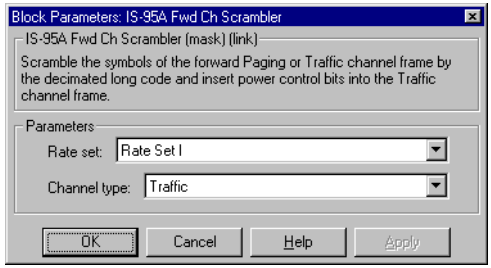
Binary scalar representing the power control bit.

Outputs

Real vector or scalar of bipolar data that represents the scrambled data and inserted power bit symbols or repeated symbols. The default size is four for the Sync channel and one for the Paging and Traffic channels.

IS-95A Fwd Ch Scrambler

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Sync, Paging, or Traffic.

See Also

IS-95A Fwd Ch Descrambler
IS-95A Long Code Generator
IS-95A Forward Traffic Channel Detection Demo

Specification References

IS-95A 7.1.3.1.6
J-STD-008 3.1.3.1.7

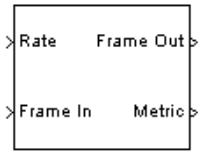
Purpose

Decode a convolutionally encoded information sequence optimally

Library

IS-95A Mobile Station Receiver

Description



This block uses the Viterbi algorithm to optimally decode a frame of convolutionally encoded information. The Viterbi algorithm searches through the trellis, as defined by the encoder, for the most probable sequence and outputs the decoded data along with the metric for the final detected path. The decoder operates in continuous and noncontinuous operation modes.

Continuous operation mode applies to the Sync and Paging channels. In this mode, the decoder will not be initialized to an all-zero state between frames of input data. This operation mode will not introduce any processing delay whenever the frame length is a multiple of the decode length. Otherwise, there will be a processing delay of one frame.

Noncontinuous operation mode applies to the Traffic channel. In this mode, the decoder initializes to an all-zero state value prior to processing each input frame. As a result, this operation mode does not introduce any processing delay. Note that in the noncontinuous operation mode, the decoder starts the traceback in the zero state. This means that the last 8 bits of the sequence being decoded must be 0.

The input frame port size is equal to the maximum possible size of 576 (Rate Set II, Traffic Full Rate). As the table below shows, the number of relevant input symbols applied to the Viterbi decoder, however, depends on the channel type, rate set, and rate. Similarly, the output port frame size is the maximum possible size of 288 (Rate Set II, Traffic Full Rate). The table below also gives the number of relevant output bits for different channel types, rate sets and rates. Valid bits start from the beginning of a frame. The block ignores input bits beyond the relevant bits, and sets output bits beyond the relevant bits to zero.

Channel Type	Number of Relevant Input Bits	Number of Relevant Output Bits
Sync	64	32
Paging, Full Rate	384	192

Channel Type	Number of Relevant Input Bits	Number of Relevant Output Bits
Paging, Half Rate	192	96
Traffic, Rate Set I, Full Rate	384	192
Traffic, Rate Set I, Half Rate	192	96
Traffic, Rate Set I, Quarter Rate	96	48
Traffic, Rate Set I, Eighth Rate	48	24
Traffic, Rate Set II, Full Rate	576	288
Traffic, Rate Set II, Half Rate	288	144
Traffic, Rate Set II, Quarter Rate	144	72
Traffic, Rate Set II, Eighth Rate	72	36

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Rate Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2

Channel Type	Data Rate (bps)	Rate Input Value
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Frame In

Real vector of size 576 representing the input frame to be decoded.

Outputs

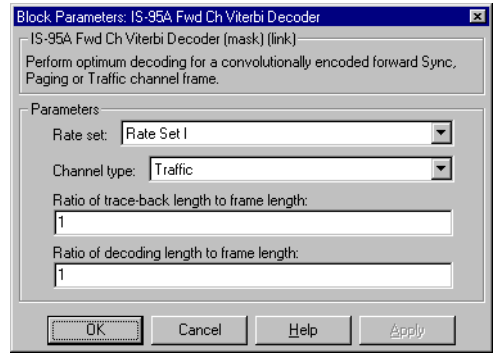
Frame Out

Binary vector of size 288 representing the decoded frame.

Metric

Real scalar representing the final path metric of the Viterbi decoder.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Sync, Paging, or Traffic.

Ratio of trace-back length to frame length

Real scalar that specifies the maximum number of bits traced back in the trellis structure as a fraction of the frame length. The maximum value is one. If the product of this ratio and the frame length is not an integer, then the product is truncated to an integer.

Ratio of decoding length to frame length

Real scalar that specifies the number of bits decoded in a traceback operation as a fraction of the frame length. If the product of this ratio and the frame length is not an integer, then the product is truncated to an integer. The ratio of decoding length to frame length should be less than or equal to the ratio of traceback length to frame length. When the decoding length is less than the traceback length, the decoded bits are the last bits traced back.

See Also

IS-95A Fwd Ch Convolutional Encoder
IS-95A Rev Ch Convolutional Encoder
IS-95A Rev Ch Viterbi Decoder
IS-95A Forward Traffic Channel Codec Demo

Purpose

Generate the long code sequence used for reverse channel spreading and scrambling and for forward channel scrambling

Library

IS-95A Common

Description

This block generates either the long code sequence or a decimated version of the long code sequence. The long code sequence is used for reverse channel spreading, while the decimated long code sequence is used for forward channel scrambling. This long code is periodic with period 2^{41} - 1 chips and uses the specified characteristic polynomial in the IS-95A specification.

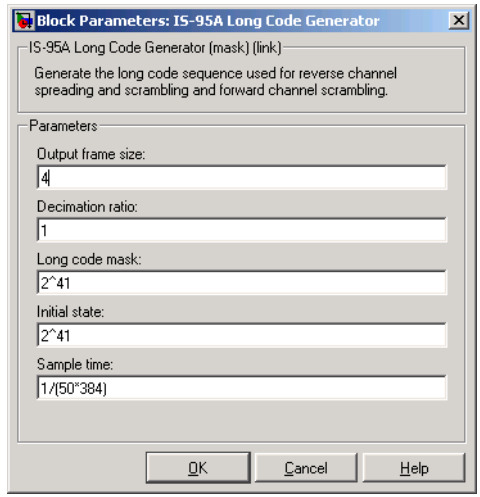
The block generates the long code sequence based on the 42-bit **Initial state** parameter and a 42-bit mask. A mask value applied to the long code generator results in an instantaneous shift in the sequence output with respect to the unmasked long code generator. A mask value of 0 results in no shift between the masked and unmasked long code generator.

The block outputs a decimated version of the long code sequence when the **Decimation ratio** parameter is an integer greater than one. If the **Decimation ratio** parameter is N, then the block's output consists of the 1st, (N+1)st, (2N+1)st, etc. values from the long code sequence.

Outputs

Binary vector corresponding to the long code sequence values or the decimated long code sequence values as specified by the parameters. The output is a frame-based signal with frame size specified by the **Output frame size** parameter.

Dialog Box



Opening this dialog causes a running simulation to pause. See “Changing Source Block Parameters” in the online Simulink documentation for details.

Parameters

Output frame size

Integer scalar that specifies the number of output long code bits at each sample time.

Decimation ratio

Integer scalar that specifies the decimation factor by which the long code sequence output is decimated.

Long code mask

Integer scalar that specifies the mask to be applied when generating the long code sequence. The mask value can be between 0 and 2^{42} - 1.

Initial state

Integer scalar that specifies the initial state of the long code generator. Its value is between 0 and 2^{42} - 1. If this parameter is an empty matrix, then the block uses a default value of 2^{41}.

Sample time

Real scalar that specifies the block output sample time.

See Also

IS-95A Fwd Ch Descrambler
IS-95A Fwd Ch Scrambler
IS-95A Rev Ch Burst Randomizer
IS-95A Forward Traffic Channel Detection Demo

Specification References

IS-95A 6.1.3.1.8
J-STD-008 2.1.3.1.8.1, 2.1.3.1.8

IS-95A Rev Ch Burst Randomizer

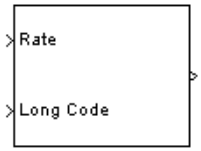
Purpose

Generate a pseudorandom masking pattern of zeros and ones that can be used to mask redundant power groups

Library

IS-95A Mobile Station Transmitter

Description



This block generates a pseudorandom masking pattern of zeros and ones. The pattern can be used to mask redundant power groups in a data frame. The data rate and 14 bits from the long code determine the masking pattern.

Each 20-ms frame is composed of 16 power groups. The 14 long code bits that determine the data burst pattern for the current frame are the last 14 long code bits in the 15th power control group of the previous frame. Using these 14 bits and the algorithm specified in the IS-95A specifications, this block outputs a pattern that can select the valid power groups in each frame. The pattern contains a value of one in the positions that contain the valid power groups in a frame. Otherwise the pattern value is zero. The pattern will be an array of all ones for the full-rate Traffic channel.

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1

Channel Type	Data Rate (bps)	Input Value
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

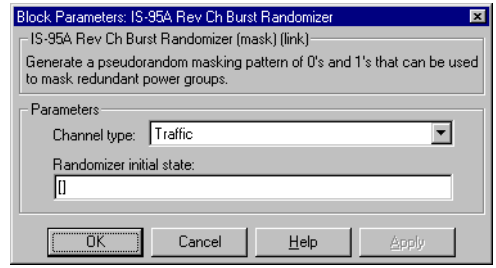
Long Code

Binary vector representing the long code bit values. The vector size is a multiple of 256. The default size is 256.

Outputs

Binary vector representing a switch that indicates the power group validity. A value of one implies that a power group is valid and a value of zero implies that the power group is redundant. The vector output size is a multiple 16. The default value is 16.

Dialog Box



Parameters

Channel type

The channel type, either Access or Traffic.

Randomizer initial state

Real vector of size 16 that specifies the initial value of the randomizer's state. This gives the location of gated-off power groups in the first frame.

See Also

IS-95A Long Code Generator

Specification References

IS-95A 6.1.3.1.7.2
J-STD-008 2.1.3.1.7.2

IS-95A Rev Ch Convolutional Encoder

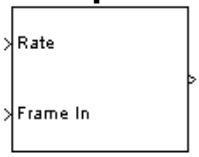
Purpose

Convolutionally encode the input data frame

Library

IS-95A Mobile Station Transmitter

Description



This block convolutionally encodes the input data frame. The data frame is composed of information bits, CRC and tail bits for reverse channels according to the IS-95A specification. Note that as a result, the last 8 bits of each Traffic or Access channel input frame must be 0.

The encoder's constraint length is nine. For the Access channel and Traffic channel Rate Set I, the convolutional encoder implements a rate 1/3 code whose generator functions are 557 (octal), 663 (octal), and 711 (octal). For Traffic channel Rate Set II, the convolutional encoder implements a rate 1/2 code whose generator functions are 753 (octal) and 561 (octal).

The input frame size is equal to the maximum possible size of 288 (Rate Set II, Traffic Full Rate). Similarly, the output size equals the maximum possible size of 576 (Rate Set II, Traffic Full Rate). The relevant number of input and output bits is specified in the table below. The block ignores input bits beyond the relevant bits, and sets output bits beyond the relevant bits to zero.

Channel Type	Number of Relevant Input Bits	Number of Relevant Output Bits
Access	96	288
Traffic, Rate Set I, Full Rate	192	576
Traffic, Rate Set I, Half Rate	96	288
Traffic, Rate Set I, Quarter Rate	48	144
Traffic, Rate Set I, Eighth Rate	24	72
Traffic, Rate Set II, Full Rate	288	576
Traffic, Rate Set II, Half Rate	144	288
Traffic, Rate Set II, Quarter Rate	72	144
Traffic, Rate Set II, Eighth Rate	36	72

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Rate Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

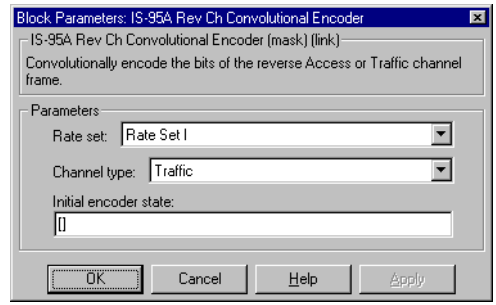
Frame In

Binary vector of size 288 representing a frame to be encoded.

Outputs

Binary vector of size 576 representing an encoded frame.

Dialog Box



IS-95A Rev Ch Convolutional Encoder

Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Access or Traffic.

Initial encoder state

Integer vector of size three that provides the initial state of the convolutional encoder. If this is an empty vector, then the block defaults to zero for all states.

See Also

IS-95A Fwd Ch Convolutional Encoder
IS-95A Fwd Ch Viterbi Decoder
IS-95A Rev Ch Viterbi Decoder
IS-95A Reverse Traffic Channel Transmitter Demo
IS-95A Reverse Traffic Channel Codec Demo

Specification References

IS-95A 7.1.3.1.3
J-STD-008 3.1.3.1.3

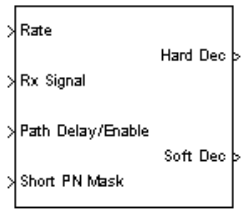
Purpose

Perform noncoherent detection of the input data frame

Library

IS-95A Base Station Receiver

Description



This masked hierarchical block performs finger-based despreading and noncoherent detection of the input data during the gate-on transmission time period. Note that the data at the output of the noncoherent rake finger are time aligned; as a result, the receiver combines the rake finger output data before the detection procedure. The rake receiver supports three fingers and provides both hard and soft output decisions.

In multirate cases (Traffic channels), where data transmission occurs in bursts, the block sends out zeros in the gate-off time slots. The data gate-on and gate-off times occur in accordance with the data burst randomization procedure specified in the IS-95A standard.

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Rx Signal

Complex vector representing the received data sequence. The vector size is the **Oversampling rate** parameter times the Walsh code length times the **Input frame size** parameter. The Walsh code length is 2^W, where W is the **Walsh order** parameter. The default size, 512, corresponds to **Oversampling rate** = 8, **Walsh order** = 6 and **Input frame size** = 1.

Path Delay/Enable

Integer vector of size six comprising three pairs of integers. Each pair of integers is associated with a finger. The first element of the pair represents the PN phase offset, in number of input sample intervals, and the second element represents the finger enable signal. A value of one for finger enable enables the finger and a value of zero disables the finger.

Short PN Mask

Integer vector of size two representing the short PN code mask for the in-phase and quadrature components. The two elements represent a complex number whose real and imaginary parts are integers between 0 and 2^{15}-1. If a scalar value is given, then the in-phase and quadrature PN generators use the same mask. If the given mask is zero, then both in-phase and quadrature PN generators take the default value of 1.

Outputs

Hard Dec

Binary vector of size 576 representing the detected frame as hard decision values.

Soft Dec

Binary vector of size 576 representing the detected frame as soft decision values.

Dialog Box

Parameters

Channel type

The channel type, either Access or Traffic.

Walsh order

Integer scalar that specifies the order of the Walsh code used to encode each data symbol. The default value, 6, corresponds to a Walsh code length of 64.

Long code mask

Integer scalar that specifies the mask value for the long code generator. This parameter can be between 1 and 2^{41}-1.

Oversampling rate

Integer scalar that specifies the number of samples per chip. The default value, 8, corresponds to a chip interval.

Sample time

Real scalar that specifies the block output sample time.

Algorithm

For more about the rake receiver algorithm, see the “Algorithm” section on the reference page for the IS-95A Fwd Ch Detector block.

IS-95A Rev Ch Detector

See Also

IS-95A Rev Ch Rake Finger
IS-95A Rev Ch Walsh Demodulator
IS-95A Reverse Traffic Channel Detection Demo

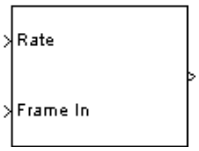
Purpose

Interleave or deinterleave a data frame for the Access channel or the reverse Traffic channel

Library

IS-95A Mobile Station Transmitter

Description



This block interleaves or deinterleaves a data frame for the Access channel or reverse Traffic channel. The interleaving time span is 20 ms. The interleaver is a matrix of 32 rows and 18 columns. The interleaver writes the code symbols into the matrix on a column-by-column basis for all of the input data symbols. It then reads the stored data into the output buffer on a row-by-row basis in accordance with the IS-95A specification to achieve the interleaving functionality. The deinterleaver restores the interleaved symbols to the original order.

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

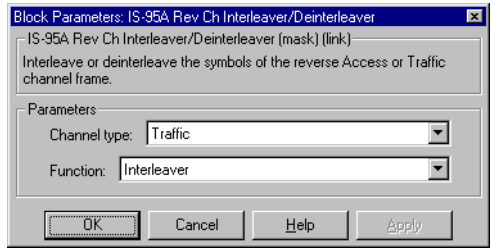
Frame In

Real vector (of binary data for interleaver functionality) of size 576 representing the input frame to be interleaved or deinterleaved.

Outputs

Real vector of (of binary data for interleaver functionality) of size 576 representing the interleaved or deinterleaved output frame.

Dialog Box



Parameters

Channel type

The channel type, either Access or Traffic.

Function

Interleaver or Deinterleaver.

See Also

IS-95A Fwd Ch Interleaver/Deinterleaver
IS-95A Reverse Traffic Channel Transmitter Demo
IS-95A Reverse Traffic Channel Codec Demo

Specification References

IS-95A 6.1.3.1.5
J-STD-008 2.1.3.1.5

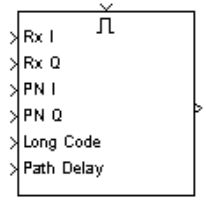
Purpose

Despread the received signal with the short PN and long codes, and then correlate the despread signal with all of the Walsh code sequences

Library

IS-95A Base Station Receiver

Description



This is a hierarchical block that downsamples the input signal using the PN phase offset and despreads the received signal with the short PN code and the long code sequences. The despreading interval equals to one quarter of the Walsh code interval for the short PN code. The long code correlator subsection of this block buffers the despread signal to reconstruct a symbol, then despreads the signal over a symbol interval with the long code and integrates it over four chip intervals to construct a Walsh symbol. Finally, this block correlates the despread signal with all of the 64 Walsh code sequences, and outputs the correlation result. The first output sample corresponds to the correlation result between the despread signal and the zeroth-order Walsh code. Subsequent output samples correspond to the correlation result between the despread signal and the subsequent Walsh codes.

Inputs

Rx I

Real vector representing the in-phase component of the received data sequence. The vector size is the **Oversampling rate** parameter times the Walsh code length. The Walsh code length is 2^W, where W is the **Walsh order** parameter. The default size, 512, corresponds to **Oversampling rate** = 8 and **Walsh order** = 6.

Rx Q

Real vector representing the quadrature component of the received data sequence. The vector size is the same as that of the Rx I input vector.

PN I

Real vector of bipolar data representing the in-phase component of the PN code sequence. The vector size equals the Walsh code length. The default size, 64, corresponds to a symbol interval.

PN Q

Real vector of bipolar data representing the quadrature component of the PN code sequence. The vector size equals the Walsh code length. The default size, 64, corresponds to a symbol interval.

Long Code

Real vector of bipolar data of size four times the Walsh code length representing the long code values. The default size, four times 64, corresponds to a Walsh symbol interval.

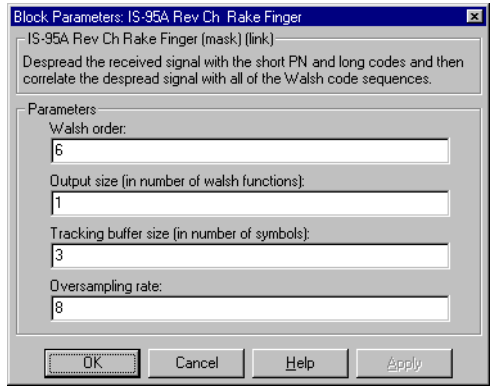
Path Delay

Integer scalar representing the PN phase offset in number of samples applied to a finger.

Outputs

Real vector representing the output of all Walsh correlators. The output vector size equals the Walsh code length. The default size is 64.

Dialog Box



Parameters

Walsh order

Real scalar that specifies the order of the Walsh code.

Output size (in number of Walsh functions)

Integer scalar that specifies the number of output values per Walsh correlator.

Tracking buffer size (in number of symbols)

Integer scalar that specifies the buffer length in the short code despreader. This buffer length limits the maximum PN phase offset between received multipaths. The default value, 3, corresponds to 192 chips.

Oversampling rate

Integer scalar that specifies the number of samples per chip.

See Also

IS-95A Rev Ch Short Code Despreader
IS-95A Rev Ch Walsh Demodulator

IS-95A Rev Ch Repeater/Derepeater

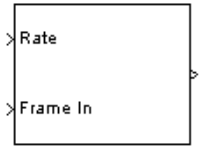
Purpose

Repeat or derepeat the symbols of the input frame for the reverse Access and Traffic channels based on the data rate

Library

IS-95A Mobile Station Transmitter

Description



This block repeats or derepeats the symbols of the input frame for the reverse Access and Traffic channels based on the data rate.

In repeater operation mode, the block repeats each input symbol whenever the information rate is lower than full rate. Each symbol is repeated once for half rate, three times for quarter rate, and seven times for eighth rate. This results in the same modulation symbol rate for all data rates.

In derepeater operation mode, the block obtains the symbols by adding together the repetitions of the transmitted symbols.

The size of the input port is equal to the maximum possible size of 576 (Rate Set II, Traffic Full Rate) for the repeater mode of operation. Similarly, the output size is equal to the maximum possible size of 576 (Rate Set II, Traffic Full Rate) for the derepeater mode of operation. Any unused bits in the output are set to zero. The number of relevant input bits for repeater operation mode and relevant output bits for derepeater operation mode are specified in the table below.

Channel Type	Number of Relevant Input Bits (Repeater)	Number of Relevant Output Bits (Derepeater)
Access	288	288
Traffic, Rate Set I, Full Rate	576	576
Traffic, Rate Set I, Half Rate	288	288
Traffic, Rate Set I, Quarter Rate	144	144
Traffic, Rate Set I, Eighth Rate	72	72

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, or Quarter, use an input value of 0, 1, or 2,

respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2

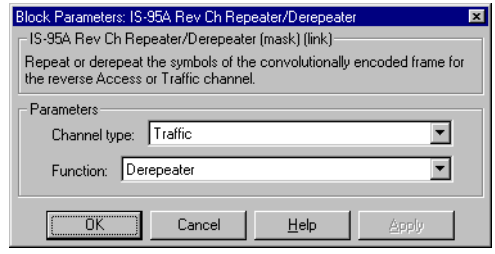
Frame In

Real vector (of binary data for repeater functionality) of size 576 containing the input frame.

Outputs

Real vector (of binary data for repeater functionality) of size 576. It represents the repeated or derepeated output frame, depending on the **Function** parameter.

Dialog Box



Parameters

Channel type

The channel type, either Access or Traffic.

IS-95A Rev Ch Repeater/Derepeater

Function

Repeater or Derepeater.

See Also

IS-95A Fwd Ch Repeater/Derepeater
IS-95A Reverse Traffic Channel Transmitter Demo
IS-95A Reverse Traffic Channel Codec Demo

Specification References

IS-95A 6.1.3.1.4
J-STD-008 2.1.3.1.4

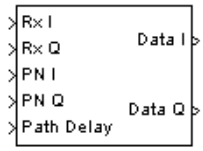
Purpose

Despread the input sequence with the I and Q short PN codes

Library

IS-95A Base Station Receiver

Description



This block despreads the input sequence with the I and Q short PN codes. The block downsamples the in-phase and quadrature components of the input complex signal at the sampling phase selected and despreads them with the delayed version of the I and Q short PN codes. By using the appropriate delay with the PN codes, it despreads the signal of interest without despreading those signals that are received from different mobiles or through different propagation paths. The block keeps an internal buffer for the received sequence. The delay adjustment for the despreading involves selecting the samples for despreading based on the input path delay information. The block then adjusts the received sequence with the delayed version of the PN code.

Inputs

Rx I

Real vector representing the in-phase component of the received data sequence. The vector size is the **Oversampling rate** parameter times the **Walsh size** parameter.

Rx Q

Real vector representing the quadrature component of the received data sequence. The vector size is the **Oversampling rate** parameter times the **Walsh size** parameter.

PN I

Real vector of bipolar data of size **Walsh size** representing the in-phase component of the PN code sequence.

PN Q

Real vector of bipolar data of size **Walsh size** representing the quadrature components of the PN code sequence.

Path Delay

Real nonnegative integer scalar that represents the delay in the desired signal path to be used for selecting the samples for despreading, in terms of number of sampling intervals or ticks. This delay should include the filtering and multipath delays that the PN sequence based despreading should take into account.

IS-95A Rev Ch Short Code Despreader

Outputs

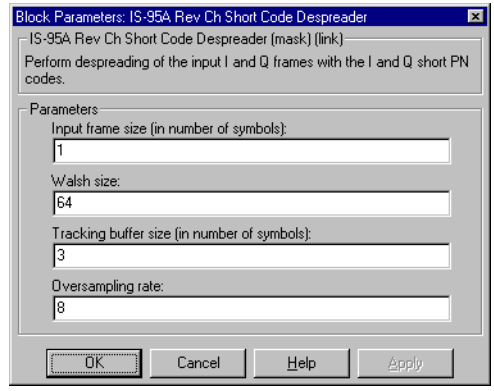
Data I

Real vector representing the in-phase component of the despread signal. The default size is 64, corresponding to the default input interval.

Data Q

Real vector representing the quadrature component of the despread signal. The default size is 64.

Dialog Box



Parameters

Input frame size (in number of symbols)

Integer scalar that specifies the number of input symbols.

Walsh size

Integer scalar that specifies the length of the Walsh code sequence.

Tracking buffer size (in number of symbols)

Integer scalar that specifies the length of the buffer needed in the despreader in terms of number of symbols. This buffer is used for the delay adjustment and its length limits the maximum PN phase delay.

Oversampling rate

Integer scalar that specifies the number of samples per chip. The default value, 8, corresponds to a chip interval.

See Also

IS-95A Rev Ch Rake Finger
IS-95A Rev Ch Walsh Demodulator
IS-95A Rev Ch Walsh Modulation and Spreading

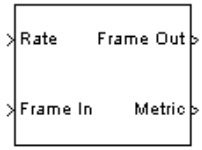
Purpose

Decode a convolutionally encoded information sequence optimally for the Access and reverse Traffic channels

Library

IS-95 Base Station Receiver

Description



This block decodes a convolutionally encoded information sequence optimally for the Access and reverse Traffic channels. For the Access channel and Rate Set I of the Traffic channel, a 1/3 rate Viterbi decoder is used. For Rate Set II of the Traffic channel, a 1/2 rate Viterbi decoder is used.

The Viterbi algorithm searches through the trellis, which defines all possible allowed sequences determined by the encoder, for the most probable sequence. This is done by updating a likelihood metric for each possible choice of the decoded sequence and selecting the paths corresponding to the best metric choices. Finally, it outputs the decoded data along with the metric for the final detected path. The decoder performs a traceback along the survivor paths in the trellis to generate the output bits.

Note This traceback is started in the zero state, so the last 8 bits of the sequence being decoded must be 0.

For this block, the Frame In input size is equal to the maximum possible size of 576 (Rate Set II, Traffic Full Rate). The number of relevant input bits applied to the convolution decoder depends on the channel type, rate set, and rate as specified in the table below. Similarly, the output frame size is equal to the maximum possible size (Rate Set II, Traffic Full Rate). The number of relevant output bits is also specified in the table below. The relevant bits start from the beginning of a frame. The block ignores input bits beyond the relevant bits, and sets output bits beyond the relevant bits to zero.

Channel Type	Number of Relevant Input Bits	Number of Relevant Output Bits
Access	288	96
Traffic, Rate Set I, Full Rate	576	192
Traffic, Rate Set I, Half Rate	288	96
Traffic, Rate Set I, Quarter Rate	144	48
Traffic, Rate Set I, Eighth Rate	72	24
Traffic, Rate Set II, Full Rate	576	288
Traffic, Rate Set II, Half Rate	288	144
Traffic, Rate Set II, Quarter Rate	144	72
Traffic, Rate Set II, Eighth Rate	72	36

Inputs

Rate

Integer scalar that specifies the data rate of the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Rate Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Frame In

Real vector of size 576 containing the input frame to be decoded.

Outputs

Frame Out

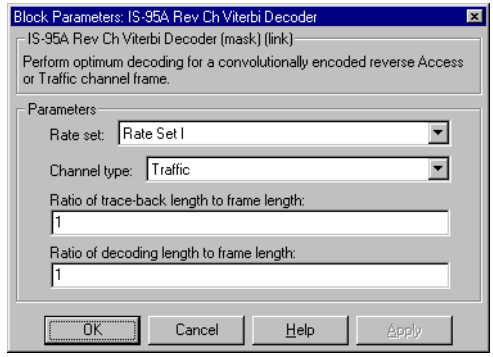
Binary vector of size 288 containing the decoded frame.

Metric

Real scalar representing the final path metric of the Viterbi decoder.

IS-95A Rev Ch Viterbi Decoder

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Access or Traffic.

Ratio of trace-back length to frame length

Real scalar that specifies the maximum number of bits that the block should trace back in the trellis structure while decoding, as a fraction of the frame length. The maximum value is 1. If the product of this ratio and the frame length is not an integer, then the product is truncated to an integer.

Ratio of decoding length to frame length

Real scalar that specifies the number of bits decoded in a traceback operation as a fraction of the frame length. If the product of this ratio and the frame length is not an integer, then the product is truncated to an integer. The decoding length to frame length ratio should be equal to or less than the traceback length to frame length ratio. When the decoding length is less than the traceback length, the decoded bits are the last bits traced back.

See Also

IS-95A Fwd Ch Convolutional Encoder
IS-95A Rev Ch Convolutional Encoder
IS-95A Fwd Ch Viterbi Decoder
IS-95A Reverse Traffic Channel Codec Demo

Specification References IS-95A 6.1.3.1.3
J-STD-008 2.1.3.1.3

IS-95A Rev Ch Walsh Correlator

Purpose

Correlate the input sequence with a bank of Walsh sequences for the reverse Access and Traffic channels

Library

IS-95A Base Station Receiver

Description

This block correlates the input sequence with a bank of Walsh sequences for the reverse Access and Traffic channels. The number of correlators is 2^W, where W is the **Walsh order** parameter. The Walsh sequences used in the correlators are mutually orthogonal.

The input comprises the received Walsh symbol(s), where each Walsh symbol is of the same length as a Walsh sequence. The block correlates each input Walsh symbol with all possible Walsh sequences.

The output of the correlator comprises the output of all correlators for all Walsh symbols input. Therefore, the first 2^Woutput values correspond to the correlation results for the first input Walsh symbol with all the Walsh sequences, the second 2^Woutput values correspond to the correlation results for the second input Walsh symbol with all the Walsh sequences, and so forth.

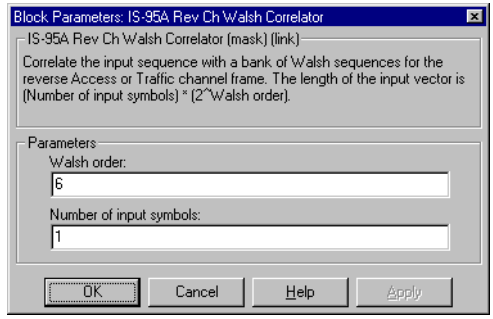
Inputs

Real vector representing the input sequence to be correlated. The vector size is **Number of input symbols** * 2^W.

Outputs

Real vector representing the output of 2^W Walsh correlators. The vector size is a multiple of 2^W. The default vector size is 2^6.

Dialog Box



Parameters

Walsh order

Real integer that specifies the order of the Walsh code.

Number of input symbols

Integer scalar that specifies the number of input Walsh symbols.

See Also

IS-95A Rev Ch Walsh Modulation and Spreading
IS-95A Rev Ch Walsh Demodulator
IS-95A Rev Ch Rake Finger

IS-95A Rev Ch Walsh Demodulator

Purpose

Perform Walsh demodulation of the combined output of the rake fingers

Library

IS-95A Base Station Receiver

Description

This block performs Walsh demodulation of the combined output of the rake fingers. It outputs soft and hard decision values.

This block takes as input an array of 64 values whose value corresponds to the correlation result between the despread received signal and the 64-ary Walsh codes. The correlation result between the zeroth Walsh code and despread signal is the first value in the input array. For the hard decision outputs, this block determines the maximum value of the input array, and the index corresponding to the maximum value is a 6-bit hard decision output. For the soft decision values, this block applies the soft-decision algorithm specified in the book by Viterbi listed in "References" below.

Inputs

Real vector representing the combined output of the rake fingers. The vector size equals **Number of input symbols** times the Walsh code length. The Walsh code length equals 2^W, where W is the **Walsh order** parameter.

Outputs

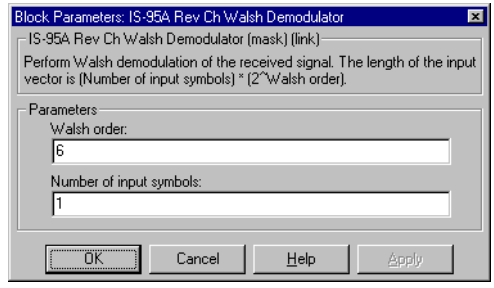
Hard Dec

Binary scalar representing demodulated hard decision values.

Soft Dec

Real scalar representing demodulated soft decision values.

Dialog Box



Parameters

Walsh order

Integer scalar that specifies the order of the Walsh code used to encode each data symbol. The default value, 6, corresponds to a Walsh code length of 64.

Number of input symbols

Integer scalar that specifies the number of input Walsh symbols.

See Also

IS-95A Rev Ch Walsh Modulator
IS-95A Rev Ch Walsh Modulation and Spreading

References

Viterbi, Andrew J. *CDMA Principles of Spread Spectrum Communications*. Reading, Mass.: Addison-Wesley, 1995.

IS-95A Rev Ch Walsh Modulation and Spreading

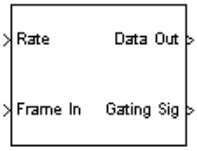
Purpose

Perform Walsh code modulation, spreading with the long code, and power group randomization

Library

IS-95A Mobile Station Transmitter

Description



This block first performs an M-ary Walsh modulation, where $M = 2^W$ and W is the **Walsh order** parameter. For every W input bits, the block generates one of the Walsh symbols from a set of 2^W Walsh sequences. Then the block internally generates the long code with which it spreads the Walsh symbols by a factor of 4.

The reverse channel transmission is gated in such a manner that the transmission in the nonfull-rate cases is only performed for a fraction of the time: half the time for half-rate channel, quarter for quarter-rate and an eighth of the time for the eighth-rate. This gating is done in terms of 1.25-ms intervals known as power groups. Using the long code, the internal data burst randomizer generates a power group rate-gating switch to gate off the redundant power groups after long code spreading.

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1

Channel Type	Data Rate (bps)	Input Value
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Frame In

Binary vector of size 576 representing the input data frame to be modulated.

Outputs

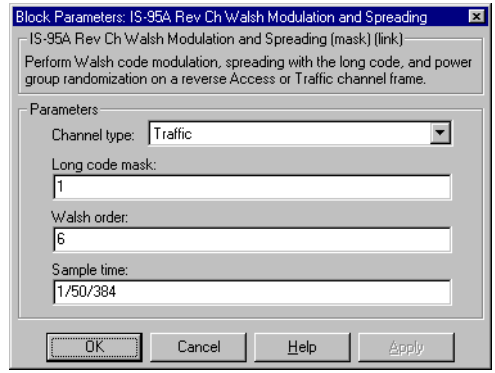
Data Out

Real vector of bipolar data (scaled by $2^{-1/2}$) of size 2^W representing the modulated, spread, and randomized data sequence. The output is provided once per sample period, which results in four outputs for each Walsh symbol. The default size, 64, corresponds to one-fourth of each Walsh symbol interval.

Gating Sig

Binary vector representing the masking signal that indicates whether the data symbol is masked or if it is to be transmitted. The default size is 1 and it is generated once per Walsh symbol.

Dialog Box



Parameters

Channel type

The channel type, either Access or Traffic.

Long code mask

Integer scalar that specifies the mask applied to the long code generator. It must be between 0 and 2^{42} - 1.

Walsh order

Integer scalar that specifies the order of the Walsh code used for modulation.

Sample time

Real scalar that specifies the block sample time.

See Also

IS-95A Rev Ch Walsh Demodulator
IS-95A Rev Ch Walsh Modulator
IS-95A Rev Ch Burst Randomizer
IS-95A Long Code Generator
IS-95A Reverse Traffic Channel Transmitter Demo
IS-95A Reverse Traffic Channel Detection Demo

Specification References

IS-95A 6.1.3.1.6, 6.1.3.1.7, 6.1.3.1.8
J-STD-008 2.1.3.1.6, 2.1.3.1.7, 2.1.3.1.8

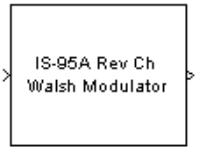
Purpose

Modulate the input data sequence with the Walsh codes

Library

IS-95A Mobile Station Transmitter

Description



This block modulates the input data bits with the Walsh codes. If $M = 2^W$ and W is the **Walsh order** parameter, then the block performs an M-ary orthogonal modulation. This process maps W input bits to one modulation symbol. One of the 2^W possible modulation sequence is transmitted for each input symbol. The modulation symbol is represented as a sequence of 2^W bipolar values. The 64-ary modulation sequences for **Walsh order** equal to 6 are given in a table in the IS-95A specification.

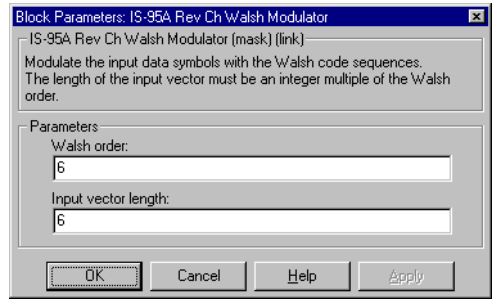
Inputs

Binary vector representing the input data sequence to be modulated. The size of the input vector is a multiple of W. The default vector size is 6.

Outputs

Real vector of bipolar data representing the Walsh code modulated data sequence. The size of the output vector is a multiple of 2^W. The default vector size is 64.

Dialog Box



Parameters

Walsh order

Integer scalar that specifies the order of the output Walsh code and therefore the corresponding modulation scheme.

Input vector length

Integer scalar that specifies the number of bits in the input. Since these bits are to be mapped to modulation symbols, this should be an integral multiple of the **Walsh order** parameter.

IS-95A Rev Ch Walsh Modulator

See Also
IS-95A Rev Ch Walsh Demodulator
IS-95A Rev Ch Walsh Correlator
IS-95A Rev Ch Walsh Modulation and Spreading

Specification References
IS-95A 6.1.3.1.6
J-STD-008 2.1.3.1.6

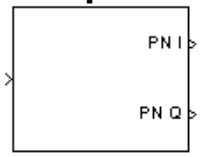
Purpose

Generate the I and Q short PN code sequences

Library

IS-95A Common

Description



This block generates the I and Q short PN code sequences to be used in the quadrature spreading and despreading of CDMA signals. At the IS-95A chip rate of 1.2288 Mcps, these binary sequences are periodic with period of 2^{15} chips (32768 chips) and are based on the specified characteristic polynomials in the IS-95A specification. The sequences repeat exactly 75 times every 2 seconds.

The specified polynomials generate sequences of length 2^{15}-1 with 14 consecutive 0's in each period of the I and Q codes. A zero is inserted after the 14 zeros to produce a sequence of 15 consecutive zeros, making the period 2^{15}. The sequence generators are implemented as linear feedback shift registers with polynomials specified in the IS-95A specification.

The state of the shift register is masked (bit-wise AND) with a 15-bit number and a bit generated as the modulo-2 addition result. The sequence of bits thus generated is the PN code sequence. The selection of different masks shifts the time to different points in the sequence being generated without changing the actual periodic sequences. The masks involve an AND operation between the LSB of the mask and the bit of the shift register used for feedback.

Inputs

Integer vector of size 2 representing short PN code mask for the in-phase and quadrature components. This determines the phase of the PN sequences' output. It is given as a complex number whose real and imaginary parts are integers between 0 and 2^{15}-1. If a real mask is given, then real and imaginary parts use the same mask. If the given mask is 0, then the default value of 1 is used for both masks of I and Q sequences resulting in a zero shifted sequence. The value of 1 implies that the feedback bit of the PN state registers will be output.

Outputs

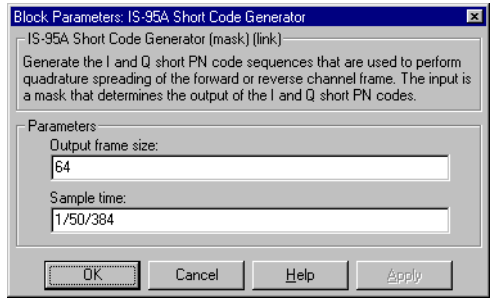
PN I

Integer vector representing the in-phase PN sequence in bipolar format. The output is a frame-based signal with frame size specified by the **Output frame size** parameter.

PN Q

Integer vector representing the quadrature PN sequence in the bipolar format. The output is a frame-based signal with frame size specified by the **Output frame size** parameter.

Dialog Box



Parameters

Output frame size

Integer scalar that specifies the length of the PN I and PN Q output vectors. The default size, 64, corresponds to the number of chips in a forward channel symbol.

Sample time

Real scalar that specifies the block sample time.

See Also

IS-95A Rev Ch Short Code Despreader
IS-95A Rev Ch Rake Finger
IS-95A Fwd Ch Rake Finger
IS-95A Fwd Ch Detector
IS-95A Reverse Traffic Channel Transmitter Demo
IS-95A Forward Traffic Channel Detection Demo
IS-95A Reverse Traffic Channel Detection Demo

Specification References

IS-95A 6.1.3.1.9
J-STD-008 2.1.3.1.9

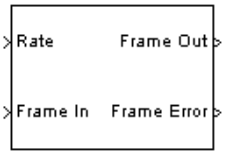
Purpose

Check the CRC bits to determine the frame quality, and output the decoded data and frame quality decision

Library

IS-95A Common

Description



This block detects a frame error in the Traffic frame by checking the received CRC bits. It regenerates the CRC bits using the received data and compares them with the received CRC. After checking for a frame error, the block removes the CRC and tail bits according to the IS-95A specification. The Frame In input size equals the maximum possible size of 288 (Rate Set II, Traffic Full Rate). Similarly, the Frame Out output size equals the maximum possible size of 268 (Rate Set II, Traffic Full Rate).

For the Sync and Paging channel, the block passes the data to the output without any processing. In the case of the Access channel, the block removes the tail bits before passing the data to the output.

The following table shows the number of relevant input and output bits. Any unused output bits are set to zero.

Channel Type	Number of Relevant Inputs Bits	Number of CRC Bits	Number of Tail Bits	Number of Relevant Output Bits
Sync, Eighth Rate	32	0	0	32
Paging, Full Rate	192	0	0	192
Paging, Half Rate	96	0	0	96
Access	96	0	8	88
Traffic, Rate Set I, Full Rate	192	12	8	172
Traffic, Rate Set I, Half Rate	96	8	8	80
Traffic, Rate Set I, Quarter Rate	48	0	8	40
Traffic, Rate Set I, Eighth Rate	24	0	8	16
Traffic, Rate Set II, Full Rate	288	12	8	268

Channel Type	Number of Relevant Inputs Bits	Number of CRC Bits	Number of Tail Bits	Number of Relevant Output Bits
Traffic, Rate Set II, Half Rate	144	10	8	126
Traffic, Rate Set II, Quarter Rate	72	8	8	56
Traffic, Rate Set II, Eighth Rate	36	6	8	22

Inputs

Rate

Integer scalar that specifies the data rate for the input signal. To indicate a rate fraction of Full, Half, Quarter, or Eighth, use an input value of 0, 1, 2, or 3, respectively. The table below shows all valid channel types and rate fractions, along with their associated data rates and Rate input values.

Channel Type	Data Rate (bps)	Input Value
Sync (always Eighth rate)	1200	3
Paging, Full	9600	0
Paging, Half	4800	1
Access (always Half rate)	4800	1
Traffic, Rate Set I, Full	9600	0
Traffic, Rate Set I, Half	4800	1
Traffic, Rate Set I, Quarter	2400	2
Traffic, Rate Set I, Eighth	1200	3
Traffic, Rate Set II, Full	14400	0
Traffic, Rate Set II, Half	7200	1
Traffic, Rate Set II, Quarter	3600	2
Traffic, Rate Set II, Eighth	1800	3

Frame In

Binary vector of size 288 representing the decoded data followed by the CRC and tail bits.

Outputs

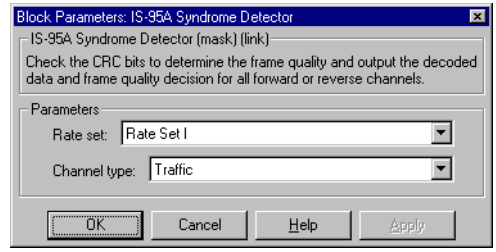
Frame Out

Binary vector of size 268 representing the data without CRC and tail bits.

Frame Error

Real scalar of binary data representing the output of the CRC syndrome. A value of zero indicates that the detected frame was decoded correctly. A value of one indicates that the detected frame was decoded erroneously.

Dialog Box



Parameters

Rate set

The rate set, either Rate Set I or Rate Set II.

Channel type

The channel type, either Sync, Paging, Access, or Traffic.

See Also

IS-95A CRC Generator
IS-95A Frame Quality Detector

IS-95A Walsh Code Generator

Purpose

Generate the Walsh code sequences corresponding to the specified Walsh order and indices

Library

IS-95A Common

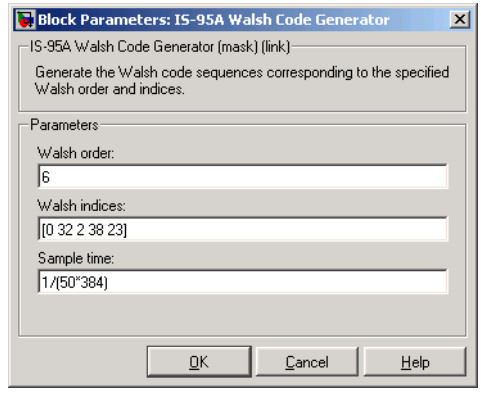
Description

This block generates the Walsh code sequences corresponding to the specified Walsh order and indices, according to the IS-95A specification. If W is the **Walsh order** parameter, then the block generates one of the 2^W sequences for each index specified in the parameters. Walsh code sequences have length 2^W and the different Walsh code sequences are mutually orthogonal. The Walsh code sequences corresponding to a Walsh order of 6 are given in a table in the IS-95A specification.

Outputs

Real vector of bipolar data representing the Walsh code sequences. The output size is an integer multiple of 2^W. The block outputs the Walsh code sequences of length 2^W in the order corresponding to the **Walsh indices** parameter. The output is a frame-based signal.

Dialog Box



Opening this dialog causes a running simulation to pause. See “Changing Source Block Parameters” in the online Simulink documentation for details.

Parameters

Walsh order

Integer scalar that specifies the Walsh code order. The output of the block consists of vector(s) of 2^W elements. The default value for IS-95A is 6.

Walsh indices

Integer vector that specifies the index or indices of the Walsh code sequences to be generated. If this parameter contains one index, then the output is a vector of length 2^W elements containing the Walsh code sequence of that index. If this parameter has length N, then the output is N concatenated vectors. Each output segment of length 2^W contains the Walsh code sequence for one of the indices given in the vector.

Sample time

Real scalar that specifies the block sample time.

See Also

IS-95A Fwd Ch Rake Finger
IS-95A Fwd Ch Detector
IS-95A Fwd Ch Base Station Transmitter Interface
IS-95A Forward Traffic Channel Detection Demo

Specification References

IS-95A 7.1.3.1.8
J-STD-008 3.1.3.1.8, 3.1.3.1.9

Index

F

I

S

T

V

W

Z