Mercury Computer Systems, Inc.

Bit Packetizer Components

FM3TR Waveform Reference Implementation

SDR Forum Contract

March 23, 2007

Revision 1.0

Table of Contents

1 Component Name

BitPacketizer_5to7, BitPacketizer_7to5

2 Component Processing Summary

The bit packetizer components simply convert a sequence of 5-bit symbols into a sequence of 7-bit symbols (and *vice versa*). This is necessary for the data waveforms because the Reed Solomon (RS) forward error correction coding components operate on symbols rather than individual bits. The RsHopEncoder accepts a sequence of 5-bit symbols and the RsBlockEncoder accepts a sequence of 7-bit symbols. The FM3TR specification states that the RsHopEncoder is an inner code, thus its output would feed directly into the RsBlockEncoder if not for the fact that the bits are packed into a symbol of the wrong size. The bit packetizers resolve this problem by translating a sequence of symbols that the RS encoders and decoders can understand.

3 Where used

The bit packetizer components are used whenever both the RS encoder/decoder components are used. This only occurs in the data waveforms. Because the RS hop is the inner code and the RS block is the outer code, the 7-to-5 packetizer is used between the RsBlockEncoder and RsHopEncoder components, and the 5-to-7 packetizer is used between the RsHopDecoder and RsBlockDecoder components.

4 Data Input and Output Ports

The bit packetizers each have one uses and one provides port. All ports accept the char data type for which each element represents a 5- or 7-bit symbol.

5 Control Interfaces

The bit packetizers inherit the control interfaces from CF::Resource.

6 Component SCA Properties

Aside from the DLL execparams, the bit packetizers have no additional properties.

7 Component Attributes/Key Variables

Below is a list of several key variables to the bit packetizer components with a brief description of their purpose.

`m_input`	Temporary input buffer (char) to store symbols before conversion.
`m_output`	Temporary output buffer (char).

8 Processing Details

The processing behind the packetizers is straightforward; symbols are stored in the input buffer until a sufficient number are received at which point the symbols are converted and stored into an output buffer. The conversion process is achieved with binary & and >> operations. The most notable constraint behind the processing is that the packetizers require a certain number of input symbols before the conversion process begins. More specifically, the BitPacketizer_5to7 component requires seven 5-bit symbols to convert to five 7-bit symbols. The BitPacketizer_7to5 produces the converse.

However, in order to reduce CORBA overhead, the actual number of symbols pushed to and from the bit packetizer components is exactly the number symbols required by the corresponding RS encoders. The table below aims to describe this. Keep in mind that a "block" in the FM3TR specification contains exactly 10 ½ "hops."

Component Name	*Accepts*	*Produces*
`RsBlockEncoder`	(72) 7-bit symbols	(105) 7-bit symbols
`BitPacketizer_7to5`	7-bit symbols	5-bit symbols
`RsHopEncoder`	(14) 5-bit symbols	(16) 5-bit symbols
`RsHopDecoder`	(16) 5-bit symbols	(14) 5-bit symbols
`BitPacketizer_5to7`	5-bit symbols	7-bit symbols
`RsBlockDecoder`	(105) 7-bit symbols	(72) 7-bit symbols