

**Raw Telemetry Format at the RSC  
(Using the \_all\_long\_96k gather)**

*Version 1.1*

April 1, 1998

John M. McGlinchey

Lockheed Martin Technical Operations  
Research, Development, Test and Evaluation Support Complex  
3550 Aberdeen Avenue SE  
Kirtland Air Force Base  
Albuquerque, NM 87117

mcglinjm@plk.af.mil

## Table Of Contents

1.0 Introduction.....	3
2.0 Data Gatherers.....	3
3.0 Data Packets.....	3
4.0 The Raw Telemetry File.....	4
5.0 Standard Parameters Contained In Raw Telemetry File.....	5
6.0 Summary.....	5

## 1.0 Introduction

The purpose of this document is to summarize the raw telemetry files that are stored at the RSC during a contact. The `_all_long_96k` gather is described as well as the standard parameters that are contained in a raw telemetry file.

The term “raw telemetry” applies to telemetry that has been decrypted and passed from the key generator (KG) to the front end’s telemetry input/output (TIO) module.

## 2.0 Data Gatherers

The mechanism used to transfer the raw telemetry data from the front end to the string server is called a data gather. A data gather is an algorithm that executes on one of the front end’s field programmable processors (FPP). The purpose of this algorithm is to read a parameter’s value from the front end’s MUXbus and store it in a buffer. The type of gather is dependent on the size of the data being gathered (two-byte or four-byte chunks) and the size of the buffer being used on the FPP.

The most common gather used for raw telemetry is the `_all_long_96k` gather. This algorithm uses four-byte (long) chunks and has a 96 kilobyte buffer. The “all long” gathers store MUXbus data, along with identifying indices for the data, in dual-sample arrays. For each parameter, the “all long” gathers store a 32-bit data value and a 16-bit index. To ensure that the “all long” transactions contain only 32-bit values, the gathers use a dual-sample structure in which two parameter values and two indices are combined.<sup>1</sup>

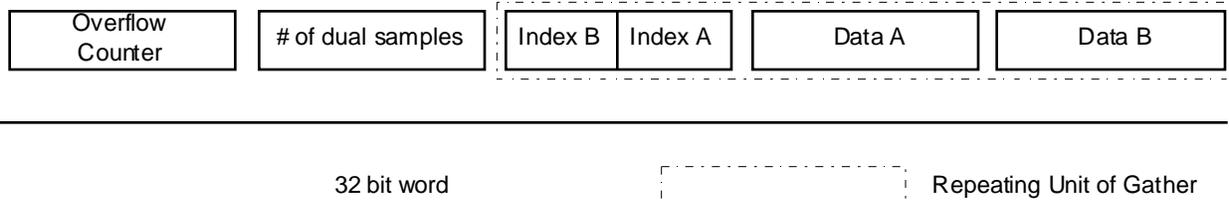
The other half of the gather is a process on the string server that reads the contents of the buffer at a specified rate. The rate is to be optimized such that the algorithm uses as much of the buffer as possible without any overflow errors.

## 3.0 Data Packets

The data is transferred across the network from the front end to the string server in data packets. Each data packet represents the contents of the buffer at a specific time. Each data packet has the following format:

---

<sup>1</sup> Lockheed Martin Telemetry and Instrumentation, System 500 User’s Manual: System Administration, 1994.



The overflow counter is the number of samples the gather algorithm had to discard because its buffer was full. The number of dual samples refers to the number of index/data pairs that follow in the data packet. This indicates the length of the data packet. The actual number of samples can be calculated as twice the number of dual samples.

## 4.0 The Raw Telemetry File

The exact format of the raw telemetry file follows. The beginning of the file contains a text portion which describes the data contained in the file. The text portion of the raw telemetry file is as follows:

```

lfe: A
gather: _all_long_96k
protocol: tcp
rate: 0.500000
maxFileSize: 10000000
convertInput: No
output.format: shb
param.0: A:TIOIN1_DATA
param.1: A:TIOIN1_START
param.2: A:IRIG_FS_MAJ
param.3: A:IRIG_FS_MIN1
param.4: A:IRIG_FS_MIN2
param.5: A:TIOIN1_STAT
param.6: A:IRIG_FS_MIC
dgHdrLen: 24
maxDatagramLen: 33120
byteOrder: bigEndian

```

The binary data that follows in the file is in the format discussed in the Data Packet section of this document (the italicized headings do *not* appear in the file):

```

Overflow   Sample   Index   Data   Data   Index   Data   Data   Index
Count     Count     1  0     0     1     0  2     2     0     0  3
00000000  00000006  00010000  000020ff  0000faf3  00010000  00001111  0000ffff  00010003

   Data     Data
   3       0
00001111  00010004 ...

```

This continues until a number of dual samples equal to the dual sample count has been reached. At this point another overflow count and sample count are found. This format repeats until the end of the file.

## 5.0 Standard Parameters Contained In Raw Telemetry File

There are seven parameters contained in all of the raw telemetry files. They are defined below.

<b>Parameter</b>	<b>Description</b>
TIOIN1_START	The frame synchronization data.
TIOIN1_DATA	The frame data that follows the synchronization data.
TIOIN1_STAT	A status word from the TIO.
IRIG_FS_MAJ	The IRIG major time word at the time the frame was synchronized at the TIO.
IRIG_FS_MIN1	The first IRIG minor time word at the time the frame was synchronized at the TIO.
IRIG_FS_MIN2	The second IRIG minor time word at the time the frame was synchronized at the TIO.
IRIG_FS_MIC	The IRIG microsecond word at the time the frame was synchronized at the TIO.

## 6.0 Summary

This document provides basic information on data gather algorithms, data packets and the raw telemetry file itself. Sample files are available from the RSC on several common media.

Attachment A

Additional Information

on the

all\_long\_96K Gather

There are [# dual samples] occurrences of the () section and then another repeat of the whole thing. The [# dual samples] should not contain BYTES. The dual sample count entry indicates the length of the transaction. It is the number of dual samples in the array of dual samples that follow the header. The actual number of samples can be calculated as dual samples \* 2. The "all long" gather store MUXbus data, along with the identifying indexes for the data, in dual sample arrays. For each parameter, the "all long" gathers store a 32-bit data value and the 16-bit index. To ensure that the "all long" transactions contain only 32-bit values, the gathers use a dual sample structure in which two parameter values and two indexes are combined, as shown below. Additionally, the dual sample structure makes it necessary for the Field Programmable Processor (FPP) to add a "padding sample" to a data packet if the gather algorithm has an odd number of samples when the time comes to transfer a data packet. The FPP fills in the last dual sample in the packet with an extra sample whose index value is set to the value 0xFFFF.

All Long Transaction Structure	
Over Flow Count	Header
Dual Sample Count	
IndexB IndexA	Dual Sample
Data A	
Data B	
IndexB IndexA	Dual Sample
Data A	
Data B	
.	
.	
.	

param.n: A:<mnemonic>

Specifies the name of the parameter on the logical front end to be gathered at the nth index for the gather. The n is the desired index of the parameter in the gather. The index is an identifier that the system uses to associate data with the correct parameter in a data packet.

Anyone using the "all long" file to re-create minor frames (frame sync and data) should use both param.0 and param.1, param.1 will contain, for ARGOS, the first 16-bits of the frame sync pattern and param.0 will contain the next 8-bits of the sync pattern followed by data that follows the frame sync, SOH data. The data for param.5, TIOIN1\_STAT, should contain values other than 0 since this is the status word from the Telemetry Input/Output module. A status value is placed at the end of each minor frame when the TIO locks on the frame sync and minor frame length. If the TIO status word does in fact contain all 0's then param.0 and param.1, TIOIN1\_DATA and TIOIN1\_START, will not contain data other than 0. If param1. and param.0 are used to recreate the minor frame then what should be seen is that for every param.1 value there should be 124 values for param.0. The TIO parameters, TIOIN1\_START and TIOIN1\_DATA, are set at 16-bits, which means the TIO will break up the data stream into 125, 16-bit samples. The reason that the incoming data was broken up into chunks of 16-bit words, the first 16-bit word being the beginning of the frame sync pattern and the remaining the SOH data, was to enable the TIO to frame synchronize with every minor frame. Therefore, the data parameter, TIOIN1\_DATA or param.0, will be written to the MUXbus 124 times for every appearance of the start parameter, TIOIN1\_START or param.1. Note this does not mean that the Data A and Data B in the "all long" file are 16-bits rather it means that the 16-bit values from the TIO parameters are in the least significant 16-bits of the 32-bit allocated Data A and Data B. The seven parameters contained in all of the raw telemetry files are defined below.

Parameter	Description
TIOIN1_START	The frame synchronization data.
TIOIN1_DATA	The frame data that follows the synchronization data.
TIOIN1_STAT	A status word from the TIO.
IRIG_FS_MAJ	The IRIG major time word at the time the frame was synchronized at the TIO.
IRIG_FS_MIN1	The first IRIG minor time word at the time the frame was synchronized at the TIO.
IRIG_FS_MIN2	The second IRIG minor time word at the time the frame was synchronized at the TIO.
IRIG_FS_MIC	The IRIG microsecond word at the time the frame was synchronized at the TIO.

The dgHdrLen entry specifies the length of the data gather data gram header which is prepended to each data gram in the archive file. The data gram header length is the length in bytes of the binary data gram header included with each data gram written to the output. Data gather data grams contain two parts: a header and a data gram. The data gather data gram is the data gather transaction exactly as sent by the gather, the data gram contains the overflow count, dual sample count and indexBindexA DataA DataB structure. See figure below.

Archive File	
ASCII HEADER	
Data File Data Gram Header	Binary Data
Data Gather Data Gram	
Data File Data Gram Header	
Data Gather Data Gram	
.	
.	
.	

Each data gather data gram begins with a data gather data gram header, therefore 24 bytes need to be skipped prior to gathering the overflow count, dual sample count and the data. The information contained in the data gram header is:

The prevdatalen entry is the length of the previous data gram (excluding the data gather file data gram header) in the archive file. The first datagram always has 0 in this field.

The datalen entry is the length of the data gram (excluding the data gather file data gram header) immediately following; i.e., the number of bytes that were received directly from the LFE in this transaction.

The flags entry is a duplication of bits 16-31 of the xferstatus entry, containing predefined system error flags.

The irigtime entry contains the IRIG time for the data gram, converted to the UNIX secs/usecs format.

The xferstatus entry contains the transfer status from the algorithm running on the Logical Front End (LFE). The gather transfer function can return a value as any standard C function; this is the value returned.