Technical Manual

Lumistar Data Processing System (LDPS)

Part-1

The LDPS Server Application

Document:	U0990101
Editor:	B. Graber
Date:	6/5/2014

Lumistar, Inc.
2270 Camino Vida Roble, Suite L
Carlsbad, CA 92011
(760) 431-2181
www.lumistar.net

This document is provided as is, with no warranties of any kind. Lumistar, Inc. disclaims and excludes all other warranties and product liability, expressed or implied, including but not limited to any implied warranties of merchantability or fitness for a particular purpose or use, liability for negligence in manufacture or shipment of product, liability for injury to persons or property, or for any incidental, consequential, punitive or exemplary damages. In no event, will Lumistar, Inc., be liable for any lost revenue or profits, or other indirect, incidental and consequential damages even if Lumistar, Inc. has been advised of such possibilities, as a result of this document or the usage of items described within. The entire liability of Lumistar, Inc. shall be limited to the amount paid for this document and its contents.

Lumistar, Inc.
2270 Camino Vida Roble,
Suite L
Carlsbad, CA 92011
(760) 431-2181
(760) 431-2665 Fax
www.lumistar.net

TABLE OF CONTENTS

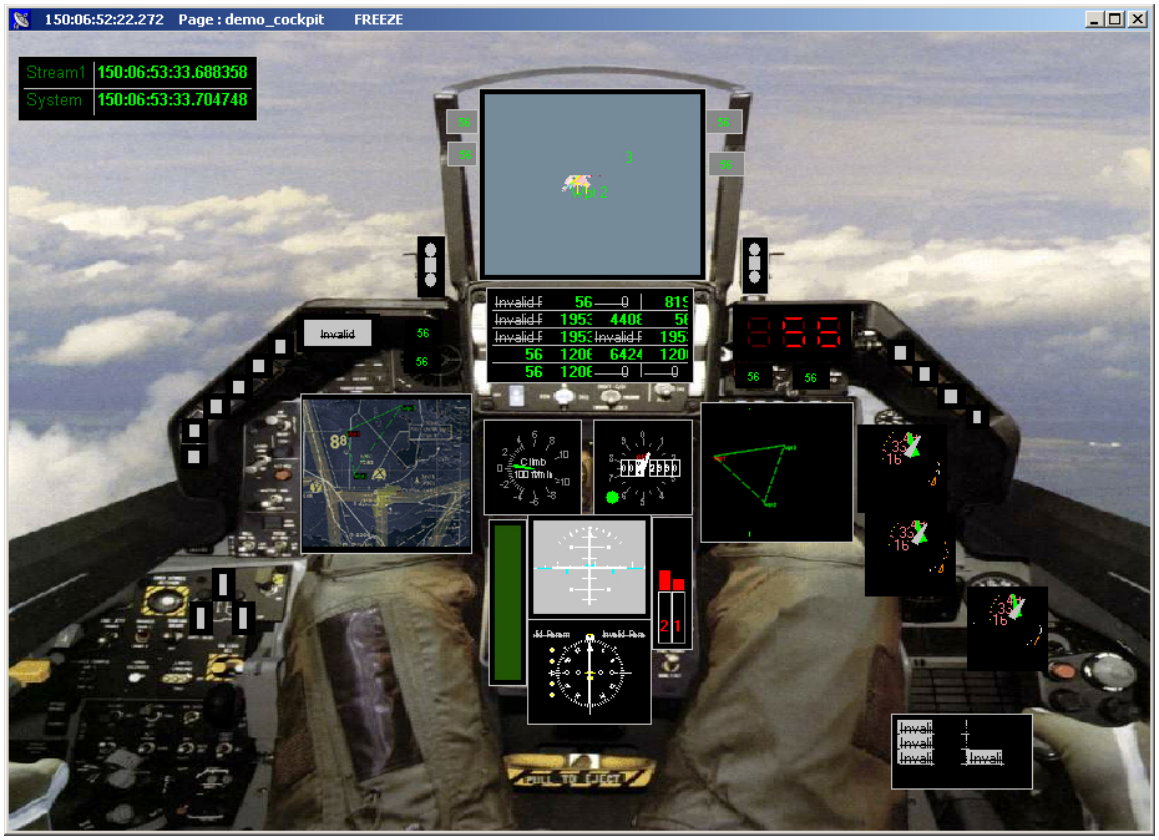
List of Tables

List of Figures

1 Introduction

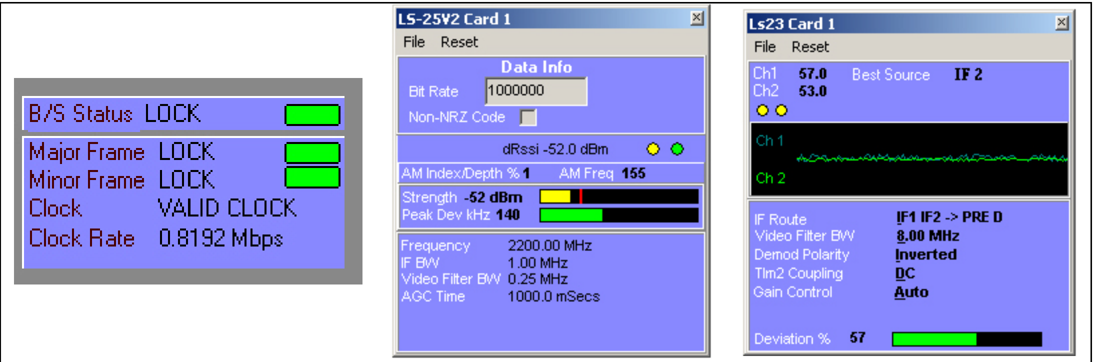
1.1 General

The Lumistar Data Processing System (LDPS) was developed out of a need to replace outdated and obsolete data collection software based on the DOS operating systems. The technology behind data acquisition is continually changing and evolving, but the techniques employed in displaying the data has not. Most users do not change the way they look at their data as often as they change the hardware that acquires it. For this reason, the LDPS application is broken up into two programs, the Lumistar Server and the Lumistar Client. The server program collects data from various sources, archives it, arranges the data into a normalized format, and then passes the data on to the client. As new technologies are developed for collecting data, only the server program need be modified. The client application is primarily a data presentation program, with software hooks to allow new display and processing routines to be added.


150:06:52:22.272 Page : demo_cockpit FREEZE
Stream1 150:06:53:33.688358
System 150:06:53:33.704748
Invalid
Climb
PULL TO EJECT

LDPS can acquire and process information from up to twelve data streams. Processing can aggregate data from any combination of individual streams. Each hardware device will have one or more streams associated with it. The streams include both data and status information for the device. The user may monitor device specific parameters such as hardware status, hardware mode, signal strength, etc. For example, a stream collecting PCM data from a decommutator will receive data from a bit synchronizer, which in turn gets data from a receiver, that in turn receives a signal from a diversity combiner, etc.

These hardware devices all belong to the same data stream and the status of this hardware can be monitored as part of the stream.



B/S Status LOCK 🟩
Major Frame LOCK 🟩
Minor Frame LOCK 🟩
Clock VALID CLOCK
Clock Rate 0.8192 Mbps

LS-25V2 Card 1

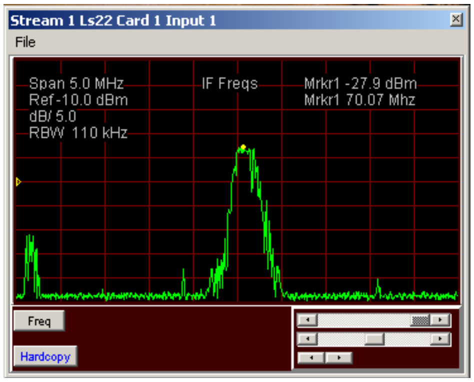
Bit Rate 1000000
Non-NRZ Code □
dRssi -52.0 dBm
AM Index/Depth % 1 AM Freq 155
Strength -52 dBm
Peak Dev kHz 140
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Ls23 Card 1

Ch1 57.0 Best Source IF 2
Ch2 53.0
IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto
Deviation % 57

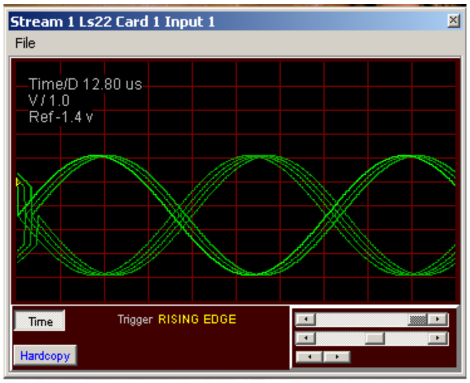
Figure 1-1 Examples of Hardware Status Displays

It is not possible for Lumistar to fully anticipate all user requirements, thus LDPS is designed to allow users to implement their own display and analysis routines. If for example, a new display widget is required and not in the widget gallery, the user may write their own custom display widget, or wrap a third party widget into the program. If the user has a different way of collecting data, or an unconventional source of data, Lumistar can investigate, upon request[1], the possibility of adding a new data collection routine to the server program, without affecting the client program. New data collection routines can also be added by the user.



Stream 1 Ls22 Card 1 Input 1

Span 5.0 MHz
Ref -10.0 dBm
dB/ 5.0
RBW 110 kHz
IF Freqs
Mrkr1 -27.9 dBm
Mrkr1 70.07 Mhz



Stream 1 Ls22 Card 1 Input 1

Time/D 12.80 us
V / 1.0
Ref -1.4 v
Trigger RISING EDGE

LDPS is designed to be extremely easy to use for anyone familiar with any type of display system. This manual is mainly intended for users new to the data collection and display community, and for those who wish to write their own data processing and/or display widget routines.

[1] Contact the factory with your specific requirements.

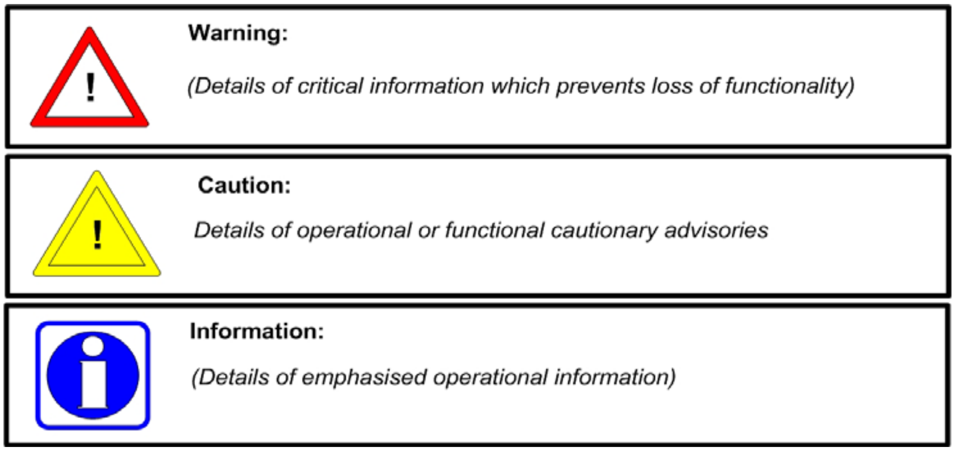
1.2 Manual Format and Conventions

The purpose of this manual is to provide a general overall functional understanding of the software and its many elements. The software will go through many changes in appearance and versions in any given time frame so some of the depicted elements within this document may vary slightly from the version that has been received in a given delivery. Updates to functional elements are typically denoted in notes that are included in "Documentation" sub-directory of the main installation directory. Customers are **strongly** encouraged to use these notes and documents as addendum to this manual.

This manual contains the following sections:

- Chapter 1 Introduction
- Chapter 2 The LDPS Server
- Chapter 3 Machine Configuration
- Chapter 4 Database Editing In LDPS
- Chapter 5 Software Decommutators
- Chapter 6 Getting Started
- Chapter 7 Appendix

Throughout this document, several document flags will be utilized to emphasis warnings or other important data. These flags come in three different formats: Warnings, Cautions, and Information. Examples of these flags appear below.



Warning:

(Details of critical information which prevents loss of functionality)

Caution:

Details of operational or functional cautionary advisories

Information:

(Details of emphasised operational information)

1.1 Proficiency

The best way to really get comfortable with the LDPS system is to use it. Experiment, create several projects, large and small, slow and fast. Create several displays. Test the system with the various projects provided with the installation to see where any limitations are. It is recommended that the user read the entire manual, at least once, cover to cover, to get a thorough understanding of the system. There is no real order it must be read either, each of the chapters and major sections are fairly self-contained.

1.2 Theory of Operation

As with any intelligent interaction, there are three basic steps: INPUT (sense), PROCESS (decide), and OUTPUT (act). The same is also true for LDPS. The data is collected, then processed or manipulated to the users requirements, and then output to a device, whether it is the CRT or disk or DAC or Network.

Data in the system is collected from either a data collection device, or from a disk drive (as archived data). The data is then archived and passed on to the client(s). The client program collects data from the server program via a shared memory arrangement in the stand-alone configuration[2], or via the network. The client program processes the data as the user has defined and outputs the data to the screen and/or to other devices defined by the user.

1.2.1 Project-Based

The LDPS application is primarily *project based*, with a project being defined as a group of related items and configurations necessary to perform a particular test. The definition process begins with the number of data streams. Up to twelve serial data streams and twelve status streams can be included in any one project. A data stream (serial or status) has many elements associated with it. These include:

a) The physical hardware devices that feed it.
b) Each hardware device can have up to 32 status parameters assigned to it. For example, these parameters can include receiver strength, decommutator major frame lock, bit synchronizer lock, etc. Serial streams can also contain up to 32K parameters defined for processing (like a data stream from a decommutator).
c) The configuration and setup of the status streams.
d) The configuration and setup of the serial data stream, contained in a parameter database.
e) Optionally, the display page, function list, derived list to load when a project is loaded.

Once a project is defined with the above information, the user may run and display device information on both the server and client.

Note: the server can control hardware devices and display some hardware information, without being assigned to a project. The client can also process non-serial parameter data as well without a project being loaded.

1.2.2 Configuration

The LDPS application must know what hardware devices are present and how to communicate with them. It must also know about any other client machines on the network. The server program has a device manager where the hardware installed in the system is identified. Configuring the device manager is usually the first thing the user

[2] Client and server program running on the same processor/machine.

does after installing the LDPS program. In addition to configuring hardware devices, if any other computers are to display server data, then the network must be configured. During operation, the sever broadcasts data to everything on the network.

1.2.3 Setup

Setup on the server involves the hardware devices. If the user has any of the Lumistar stand-alone device programs (for the LS-25, LS-50, etc), it should be noted that the user interface is very similar to those found in LDPS. With the exception of one button, they are the same, with the same application code being used. These individual programs are what allow the user to modify the setup configuration of the hardware devices on the machine, even without a project being loaded. They also allow the user to monitor data output by the device, without involving the client application.

1.2.4 Device and Stream Association

There may be scenarios where multiple cards of a single type are installed in a system (like two LS-50 decommutators). The devices are numbered as they are identified in the machine. Streams are numbered starting from the number “1.” For serial streams coming from a decommutator card (LS-50), the streams are numbered from the card as they are placed in the machine. If the decommutator card has a LS-55-DB daughtercard, then the daughtercard is numbered before the next LS-50 card. For instance, assume there are two LS-50 decommutator cards, each with a LS-55-DB daughtercard. Stream 1 would then go to the first LS-50 card. The second stream would go to the LS-55-DB on the first LS-50 card. The third stream would go to the second LS-50 card, and so on…

To complicate matters a bit, serial data can come from devices *other* than a decommutator, like a MIL-STD-1553 card. From the individual device application’s point of view, there is stream 1 for a 1553 card and stream 1 for a decommutator card. But, to the server, the 1553 card is assigned to stream 1, the first decommutator card is assigned to stream 2, the second decommutator card is assigned to stream 3, and so on. To the standalone applications, the first decommutator card is stream 1, and the standalone program has no knowledge of the 1553 program.

More complicated still, streams in LDPS are broken down into two stream types, serial streams and status streams. Status streams begin with stream 1. Serial streams also start with stream 1. The status streams are recognized by the server as streams 1 through 12 for all devices, where the stand-alone application recognizes only the cards it controls.

In general, the stream numbers are dependant on the device application and the number of streams within that application. Depending on what device applications are running, the server will number the two types of streams accordingly. If the hardware devices installed on the machine are not constantly changing, the stream numbering scenario described won’t become confusing.

1.2.5 Data Dictionary

The data dictionary (database, etc.) is the most critical part of this, or any similar system. The data dictionary contains the information required to break out the data from its raw form. Creating this database is the part that can lead to most errors.

With LDPS, the data dictionary is used mostly for breaking out the raw data and putting it into its linear, (mx + b) form. The same data dictionary can be used for many projects. Likewise, the same displays can be used for many projects.

1.2.6 Client Connection

The client program doesn't do much without the server feeding it data. When it starts, the client program will attempt to connect to the server and load the project. There may be times when the server is shut down, and the client needs to remain on. When the server program comes back up, the user will have to re-connect and load the project if the client requires the server data. This is true even if the server and client are on the same machine, unless the server option is set to always load the project.

1.2.7 Administration

Since no two users will want to operate the system in exactly the same way, there are many options to configure on the system. The server and each client have their own set of options. For most stand-alone configurations, the defaults will work just fine. Network configurations are another matter however. The user will need to pay particular attention to the options sections in this manual for both the server and client.

1.2.7.1 Supervisor Password

There are a few things that the administrative user may not want other users of LDPS to have access to, like administering the network, or disabling edit functions. These functions are restricted to the LDPS Supervisor.

Setting the supervisor password is done via the server options (see paragraph 2.11.3 on page 59). The first person that clicks the option to "Set Supervisor Password" gets to choose the password. Once the password is set it cannot be changed except by contacting Lumistar for instructions. DO NOT FORGET THIS PASSWORD. Note, the password cannot be blank. If the password is blank, then no password is created. Without a supervisor password, then sensitive items are accessible by everyone.

1.2.7.2 Password Sensitive Areas

Password sensitive areas in LDPS include the following:

- Network Administration
- Options – Edit Client Display Privileges
- Device Manager

1.2.8 User Hooks

LDPS is designed to let the user create their own application code sections for Input, Process, and Output. This is done via a Windows Dynamic Link Library (DLL) that the user can write and select from with the system.

On the server, the user can write their own data collection DLL that can collect data from any source that can find its way into a PC. The user can also write their own archive and playback DLL, to playback any ones archive format, and archive the data in what ever format that is required.

On the client, the user can write their own decommutation routine (a so-called soft decom) to pull data out of the stream in its raw form and put it into what ever form that is required. Because the DLL gets 100% of the data, the user can output in whatever form required and to any device that is needed. The user may also write their own processing function, to solve the data in whatever manner deemed necessary. On the client, the user can create their own display widgets.

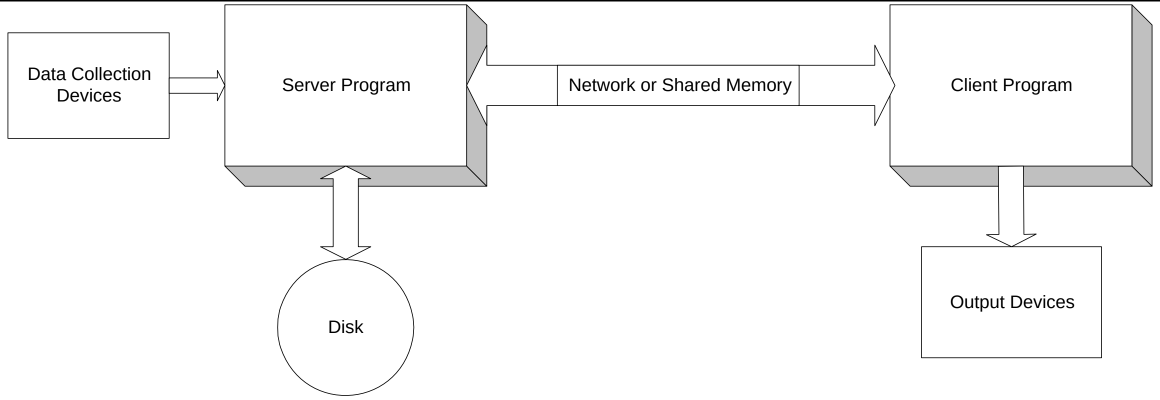


Figure 1-2 LDPS Client/Server Architecture

1.3 Parameter Database

The LDPS application is parameter name based. To the user, all that is required is the name of the parameter to be processed. The parameter database contains all of the other information needed for processing the parameter. The user does not have to know the parameters location in the data stream, or how to break out the value.

The parameter database is the heart of the processing engine (the client program). There are many ways to set up a database. The method used for LDPS is relatively fast, easy to use, and contains all the information needed for most tests. The database is an ASCII text file, with the extension “PDB.” There is also a binary version of the database, with the file extension “PBIN.” The binary version of the file is used by all of the LDPS programs. These database files are contained in the “PDBASE” directory (this directory is set up by the user in the server options – see paragraph 2.11.1 on page 53). Creating this database is the hardest task in setting up the system for a test. For large databases, this process can require many days to enter all of the information, and can be fraught with numerous typographical errors. Often, previously defined “Data Dictionaries” for a test are available to the user. These often come in different flavors, from different vendors. Sometimes they are even different from the same vendor. The task of transforming the data from a particular vendor's data dictionary to the format required by LDPS often requires the user to write a special program to gather the necessary information from the vendor file(s) and place it in the LDPS file format. This helps to keep the number of errors down to a minimum. Even a novice programmer can develop such a program in a very short time. There is also an import tool available from Lumistar that will handle most ASCII files and translate them from the vendor's format to the LDPS format.

The parameter database format is tailored around PCM and IRIG Chapter 8 telemetry streams. MIL-STD-1553 streams are also supported. The database format is flexible however, and other types of data streams may be supported. The main purpose of the parameter database is to identify the parameter, where to get the root data for the parameter, how to break out the bits used for the parameter, and to some extent, how to process it.

Most of the parameters in a telemetry test are solved in either raw (un-scaled) or scaled (Mx + b) formats. Some parameters may require multiple words to be put together prior to solving, while other parameters require that a different parameter equal some value before the parameter in question is deemed valid and can be updated. Still other parameters require a look-up table to be employed in solving them. All of this type of information is in the parameter database. For parameters that have different kinds of processing requirements, the client program has various methods for defining how the parameters are to be processed. This is discussed further in the following paragraph.

1.3.1 Parameter Solving

Parameter solving in LDPS has multiple steps. The steps involved depend on the method defined in the database for solving the parameters. The parameter-solving algorithm is summarized as follows:

1) Obtain the raw data for the current parameter.
   a) If there is a preprocessed formula for the word the parameter uses, apply the preprocessed formula to that word.
   b) Break out the bits required as defined in the database to solve for raw values.
   c) If wordswap/byteswap is set, then perform the wordswap/byteswap operation.

2) If there is a second parameter required to go with the current parameter, then:
   d) Obtain the raw data for the 2nd parameter.
   e) Break out the bits required as defined in the database to solve for raw values.
   f) If wordswap/byteswap is set, then perform the wordswap/byteswap operation.
   g) Apply the EUC[3] and Offset to the raw value for the 2nd parameter. The result is truncated into an unsigned integer.
   h) Take the raw value of the current parameter and shift it left by the number of bits used in the 2nd parameter, then add the 2nd parameter value to it.

[3] EUC – Engineering Unit Conversion.

3) If there is a mode parameter required to solve before this parameter can be solved, then:
   a) Obtain the raw data for the mode parameter.
   b) Break out the bits required as defined in the database to solve for raw values.
   c) If wordswap/byteswap is set, then perform the wordswap/byteswap operation.
   d) Apply the EUC and Offset to the raw value for the mode parameter. The result is truncated into an unsigned integer.
   e) If the result of the mode parameter matches the conditional set in the database for the mask assigned for the current parameter, then the current parameter will continue to finish solving, otherwise the current parameter's solved data will be the last value that was update when the mask and mask conditional matched. Note, the mask conditionals are equals, less than, greater than, OR, and XOR.

4) If there is a timestamp parameter associated with the current parameter, then:
   a) Obtain the raw data for the time parameter.
   b) Break out the bits required as defined in the database to solve for raw values.
   c) If wordswap/byteswap is set, then perform the wordswap/byteswap operation.
   d) Apply the EUC and Offset to the raw value for the time parameter. The result is the timestamp that will be put with the current parameter.

5) Multiply the resulting current parameter raw value (as defined in the above steps) by the EUC of the current parameter and add the Offset of the current parameter to it. Then timestamp the parameter with the timestamp from the CVT[4] for the parameter, or with the time parameter result, if this parameter has a timestamp associated with it. Note, this step is omitted if there is a mode parameter associated with this parameter and the mask for the mode parameter did not fall into the conditional for that mask.

If a parameter has other parameters associated with it, this situation may become very confusing. Some examples of how this scenario would be entered into a database are discussed below.

For parameters that require a second parameter to solve, the result of the calculation is stored in the current parameter, and the current parameter is the one used for display. For example, assume a parameter requires 19 bits, but the word lengths in the stream are only 16 bits. This then requires two words to form the parameter. Since the LDPS system is parameter based, it actually takes two parameters to make up the 19 bit parameter.

In this example, the parameters are called RADALT and RADALT2. The first parameter, RADALT, would contain 16 bits, and have a EUC assigned to it that is multiplied by the 19 bit raw value. Then any offset is added. The second parameter, RADALT2, would contain 3 bits, and normally would have a EUC of 1.0 and an offset of 0.0 assigned to it. The solve type for RADALT would be Twos Complement, while RADALT2 would be a Binary type. The resulting value put back into RADALT would be:

[4] CVT – Current Value Table.

```
((RADALTraw  ShiftLeft number of bits used in RADALT2) + RADALT2solved) * RADALTeuc +
RADALToffset.
```

The scenario described above can be confusing. **IF** the bits required to make up the entire parameter are contiguous, and the length of each word are the same, and the primary (most significant) word uses all the bits in the word, then all the user need do is state the total number of bits, and state that they are contiguous in the database and the server will return all x bits for the value. In this case, the parameter RADALT is set to 19 bits, is set to be contiguous, and the word length (number of bits) for the first and second word are both 16 bits. This only works if the parameter length is 32 or less. If the parameter is greater than 32 bits, then the previous method (using two parameters) must be used. Alternatively, the user may put the two parameters together with the concatenate function in the display system.

1.3.2 Words and Parameters

As mentioned, LDPS is parameter name based system. A parameter (also referred to as a tag) is a subset of a telemetry word. In telemetry systems, data is transmitted in words. Parameters on the other hand, sometimes only use portions of the transmitted word. For example, a transmitted word called GEARSTATE, consisting of 16 bits could have parameters called NOSEGEARUP using bit 0, NOSEGEARLOCK, using bit 1, PORTGEARUP, using bit 2, etc. This is why LDPS is parameter based. The user normally doesn’t have time to break out the bits of a word during a test. Instead, one would just assign the parameter NOSEGEARUP to a display widget, and not have to refer to the data dictionary to see which bit in the telemetry word GEARSTATE to used.

In order for a parameter name based system to work, word names and parameter names must be unique throughout the database. A parameter name can have the same name as the word name, but only once. For example, a parameter named INLAT can come from the word named INLAT, especially if the parameter uses all the bits in the word. When generating the database, this must be taken into account. Some vendors will have only a single word that uses more than 16 bits to define a parameter. Again take INLAT (inertial latitude) for example. This parameter normally takes 32 bits to define, or two 16-bit words. Some vendors will only define INLAT, using 32 bits, assuming the two words are contiguous in the data stream, when it should instead define INLAT1 and INLAT2. In this scenario, it would be ok to use INLAT as the first word, the user would have to generate INLAT2 for the second word, and fill in the information for them (each of the two words would contain 16 bits).

1.3.3 Parameter Database File Format

The parameter database is a file consisting of three main parts. This includes the file header, followed by stream information, and then the data. The file header is nothing more than a set of text instructions on how to read the database. The user may put any kind of text here. The breakpoint for the file header is a triple asterisk. Anything before the “***” is just an area for the user to put notes in. Further, this only applies to the ASCII version of the file. There is no need for a header in the binary version. Stream information immediately follows the triple asterisk. The data structure contained here is for stream information and common items used for solving data in the stream. There are several user defined (UserDef) locations for the user to place information in. These locations are typically used with soft decommutators.

The data is as follows:

```
//...
  BorWriteLn(f,"FileVersion; //dbase file version");
  BorWriteLn(f,"NumSubFrames;     //for pcm and ch8 (ch8 will be 1)");
  BorWriteLn(f,"NumPcmWordsPerSf; //for pcm and ch8");
  BorWriteLn(f,"SfidStartCount; //if sfid counts from 0, or counts from 1");
  BorWriteLn(f,"eDbaseType; //pcm, ch8, 1553");
  BorWriteLn(f,"CommonWordLen; //common word length, in bits of the data from the serial
stream");
  BorWriteLn(f,"CommonParamMsbFirst;     //if the common parameter is msb first");
  BorWriteLn(f,"CommonWordUpdateRateHz; //the common update rate for the word, in hz");
  BorWriteLn(f,(AnsiString)"  user def code word 0 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 1 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 2 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 3 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 4 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 5 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 6 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 7 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 8 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 9 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 10 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 11 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 12 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 13 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 14 //not used, user discretion");
  BorWriteLn(f,(AnsiString)"  user def code word 15 //not used, user discretion");
  BorWriteLn(f,"cLutFileName[512]; //name of lookup table file");
  BorWriteLn(f,"cSoftDecomName[512]; //name of soft decom dll");
  BorWriteLn(f,"cOriginatorText[5][512]; //text the originator can put in");

  BorWriteLn(f,"Ch8SpecificationMode; //which ch8 spec 0=iaw106-93, 1=iaw106-96");
  BorWriteLn(f,"Ch8ParityMode; //what to do with ch8 parity check,
0=dontcare,1=hopeforbest,2=dontprocessword,3=resetbus");
  BorWriteLn(f,"Ch8HasFrameTime; //ch8 has emb frame time");
  //
  BorWriteLn(f,"cEmbTimeDllName[24]  //name of embedded time dll if used");

  //
  BorWriteLn(f,"HasHotMic; //stream contains embedded audio");
  BorWriteLn(f,"HotMicStartWord; //1 based");
  BorWriteLn(f,"HotMicWordInterval;");
  BorWriteLn(f,"HotMicFormat; //0 = cvsd, 1 = pcm");
  BorWriteLn(f,"HotMicSampleRate; //0=8khz,1=11.025khz,2=12khz,see defines for match freq
to enum");
  BorWriteLn(f,"HotMicSpareBool;");
  //
  BorWriteLn(f,"HasEmbVideo; //stream contains embedded video");
  BorWriteLn(f,"EmbVideoStartWord; //1 based");
  BorWriteLn(f,"EmbVideoWordInterval;");
  BorWriteLn(f,"EmbVideoFormat;      //0 = h.261 -CIF, 1 = h.261 -CIF FEC, 2 = h.261 -
QCIF, 3 = h.261 -QCIF FEC");
  BorWriteLn(f,"EmbVideoFrameRate;  //fps");
  BorWriteLn(f,"EmbVideoSpareBool;");
  BorWriteLn(f,"EmbVideoReserved1;");
  BorWriteLn(f,"EmbVideoReserved2;");
  BorWriteLn(f,"EmbVideoReserved3;");
  BorWriteLn(f,"HasEmbTime; //stream contains embedded time (use with cEmbTimeDllName)");
  BorWriteLn(f,"EmbTimeStartWord; //1 based index");
  BorWriteLn(f,"EmbTimeNumWords; //number of words used to put emb time together");
  BorWriteLn(f,"EmbtimeUserDef; //user def for embtime dll, for things like mode, etc");

  //
  BorWriteLn(f,"HasEmbPcm1; //embedded pcm stream 1");
  BorWriteLn(f,"EmbPcm1StartWord; //1 based where emb stream is in main stream");
  BorWriteLn(f,"EmbPcm1WordInterval;");
  BorWriteLn(f,"EmbPcm1WordsPerFrame;//words per minor frame");
  BorWriteLn(f,"EmbPcm1NumSubframes; //num subframes");
  BorWriteLn(f,"EmbPcm1SfIdWord; //word number for sfid, 1 based");
  BorWriteLn(f,"EmbPcm1Cwl; //common word length, in bits");
  BorWriteLn(f,"EmbPcm1FspLength; //length of fsp, in bits");
  BorWriteLn(f,"EmbPcm1FspWord; //word number for fsp start, 1 based");
  BorWriteLn(f,"EmbPcm1Fsp1; //frame sync pattern for first 32 bits of fsp");
  BorWriteLn(f,"EmbPcm1StopWord;//1 based the stop word in main stream the emb words 
are");
  BorWriteLn(f,"EmbPcm1StopFrame; //0 based the stop frame in main stream the emb words 
are");
  BorWriteLn(f,"EmbPcm1MsbFirst; //is emb data msb first");
  BorWriteLn(f,"EmbPcm1FrameStart; //0 based where emb stream in in main stream");
  BorWriteLn(f,"EmbPcm2FrameInterval;");
  BorWriteLn(f,"EmbPcm1Reserved1; //future use");
  BorWriteLn(f,"EmbPcm1Reserved2;");
  //
  BorWriteLn(f,"HasEmbPcm2; //embedded pcm stream 2");
  BorWriteLn(f,"EmbPcm2StartWord; //1 based where emb stream is in main stream");
  BorWriteLn(f,"EmbPcm2WordInterval;");
  BorWriteLn(f,"EmbPcm2WordsPerFrame;//words per minor frame");
  BorWriteLn(f,"EmbPcm2NumSubframes; //num subframes");
  BorWriteLn(f,"EmbPcm2SfIdWord; //word number for sfid, 1 based");
  BorWriteLn(f,"EmbPcm2Cwl; //common word length, in bits");
  BorWriteLn(f,"EmbPcm2FspLength; //length of fsp, in bits");
  BorWriteLn(f,"EmbPcm2FspWord; //word number for fsp start, 1 based");
  BorWriteLn(f,"EmbPcm2Fsp1; //frame sync pattern for first 32 bits of fsp");
  BorWriteLn(f,"EmbPcm2StopWord;//1 based the stop word in main stream the emb words 
are");
  BorWriteLn(f,"EmbPcm2StopFrame; //0 based the stop frame in main stream the emb words 
are");
  BorWriteLn(f,"EmbPcm2MsbFirst; //is emb data msb first");
  BorWriteLn(f,"EmbPcm2FrameStart; //0 based where emb stream in in main stream");
  BorWriteLn(f,"EmbPcm2FrameInterval;");

  BorWriteLn(f,"Bitrate  //expected data rate");
  BorWriteLn(f,"SfIdWordNum  //1 based index, the sfid word number");
  BorWriteLn(f,"SfidMsb  //0 based index, the sfid msbit");
  BorWriteLn(f,"FramesPerInterrupt  //number minor frames of data on new data");

  //................................
  "//----- the columns of data are (tab delimited so keep the number of characters within 
range specified below. .no error checking)");
  BorWriteLn(f,(AnsiString)"//----- the columns of data are (tab delimited so keep the 
number of characters within range specified below. .no error checking)");
  BorWriteLn(f,(AnsiString)"Word Name                  (19 chars, 18 chars usable with a 
space afterwords)");
  BorWriteLn(f,(AnsiString)"Parameter Name             (19 chars, 18 chars usable with a 
space afterwords)");
  BorWriteLn(f,(AnsiString)"CVT Location (hex)         (5 chars)");
  BorWriteLn(f,(AnsiString)"PTT Location (hex)         (5 chars)");
  BorWriteLn(f,(AnsiString)"Tag Type (decimal)         (2 chars the type of tag the 
parameter is (normal, contiguous)
  BorWriteLn(f,(AnsiString)"Unit Of Measure Label      (7 chars)");
  BorWriteLn(f,(AnsiString)"Parameter Description      (32 chars)");
  BorWriteLn(f,(AnsiString)"Classification             (1 chars)  (NONE, UNCLASSIFIED, 
FOUO, FMS, CONFIDENTIAL, SECRET, TOPSECRET, USER1, USER2, USER3, USER4, USER5, USER6 )");
  BorWriteLn(f,(AnsiString)"Update Rate (hz)           (3 chars)");
  BorWriteLn(f,(AnsiString)"Is MSB first               (1 chars)  (0 = no, 1 = yes)");
  BorWriteLn(f,(AnsiString)"MSB                        (2 chars) (decimal), 0 based 
index");
  BorWriteLn(f,(AnsiString)"LSB                        (2 chars) (decimal), 0 based 
index");
  BorWriteLn(f,(AnsiString)"Parameter Number Of Bits   (2 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Word Number Of Bits        (2 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Supercom Process Mode      (1 chars) (0 = first, 1 = last, 2 
= least, 3 = greatest, 4 = average)");
  BorWriteLn(f,(AnsiString)"Data Format                (2 chars) (BINARY, TWOS, SINGLE, 
OFFSETBINARY, IEEEINTEGER, IEEESINGLE, DECSINGLE, MS1750, ONES, LUT, DOUBLE ) 
(decimal)");
  BorWriteLn(f,(AnsiString)"Special Process Flag       (2 chars) (//how to solve the
special things, like contiguous bits longer than word, byteswap,wordswap,etc");
  BorWriteLn(f,(AnsiString)"Mode Param Name            (18 chars)");
  BorWriteLn(f,(AnsiString)"Mode Tag                   (5 chars) (hex) FFFFF means not
used");
  BorWriteLn(f,(AnsiString)"Mode Mask                  (4 chars) (hex)");
  BorWriteLn(f,(AnsiString)"Mode Mask Function         (2 chars) (AND,>,<,OR,XOR mask
with value)");
  BorWriteLn(f,(AnsiString)"Time Param Name            (18 chars)");
  BorWriteLn(f,(AnsiString)"Time Tag                   (5 chars) (hex) FFFFF means not
used");
  BorWriteLn(f,(AnsiString)"Second Tag Name            (18 chars)");
  BorWriteLn(f,(AnsiString)"Second Tag                 (5 chars) (hex) FFFFF means not
used");
  BorWriteLn(f,(AnsiString)"Lookup Table Number        (2 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"EUC                        (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Plus B                     (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Hardware Field 0           (5 chars) (hex), pcm/ch8 = pcm
start frame");
  BorWriteLn(f,(AnsiString)"Hardware Field 1           (5 chars) (hex), pcm/ch8 = pcm
start word");
  BorWriteLn(f,(AnsiString)"Hardware Field 2           (5 chars) (hex), pcm = frame
interval, ch8/1553 = bus");
  BorWriteLn(f,(AnsiString)"Hardware Field 3           (5 chars) (hex), pcm = frame stop,
ch8/1553 = rt");
  BorWriteLn(f,(AnsiString)"Hardware Field 4           (5 chars) (hex), pcm = word
interval,  ch8/1553 = msg");
  BorWriteLn(f,(AnsiString)"Hardware Field 5           (5 chars) (hex), pcm = word stop,
ch8/1553 = word number");
  BorWriteLn(f,(AnsiString)"Hardware Field 6           (5 chars) (hex), pcm not used,
ch8/1553 = tr bit");
  BorWriteLn(f,(AnsiString)"Hardware Field 7           (5 chars) (hex), spare user use");
  BorWriteLn(f,(AnsiString)"Preprocessed Formula       (128 chars)");
  BorWriteLn(f,(AnsiString)"Access Code                (2 chars) (hex)");
  BorWriteLn(f,(AnsiString)"Default Max Display Value  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Default Min Display Value  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Param Display Label        (18 chars)");

  BorWriteLn(f,(AnsiString)"StreamPart                 (8 chars)(hex) part0 = main, part1
= embeddedstream1, part2 = embeddedstream2");
  BorWriteLn(f,(AnsiString)"UserDefI1                  (8 chars)(hex) //user defined,
normally for soft decoms");
  BorWriteLn(f,(AnsiString)"UserDefI2                  (8 chars)(hex)");
  BorWriteLn(f,(AnsiString)"UserDefI3                  (8 chars)(hex)");
  BorWriteLn(f,(AnsiString)"UserDefI4                  (8 chars)(hex)");
  BorWriteLn(f,(AnsiString)"UserDefD1                  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"UserDefD2                  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"UserDefD3                  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"UserDefD4                  (17 chars) (decimal)");
  //future use
  BorWriteLn(f,(AnsiString)"Future1  use at own risk (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future2  but will be     (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future3  implemented at  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future4  any time        (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future5                  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future6                  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future7                  (17 chars) (decimal)");
  BorWriteLn(f,(AnsiString)"Future8                  (17 chars) (decimal)");
  //...
  BorWriteLn(f,(AnsiString)"//-----");
  BorWriteLn(f,(AnsiString)"");
  BorWriteLn(f,(AnsiString)"");
  BorWriteLn(f,(AnsiString)"");
  BorWriteLn(f,(AnsiString)"After the header is *.*, and then the tag data starts");
  BorWriteLn(f,(AnsiString)"");

  s = GetParamDbaseHeaderLabels();
  BorWriteLn(f,s);
  BorWriteLn(f,(AnsiString)"");
  BorWriteLn(f,(AnsiString)"");
  //write the end of text header flag
  s = "***";
  BorWriteLn(f,s);
```

1.3.4 Hardware Fields

In the data area of the parameter database file there are eight (8) hardware fields for each definition. These fields have different meanings, depending on the type of data stream. The hardware fields are summarized here as follows:

Hardware Field 0	(5 chars) (hex), pcm/ch8 = pcm start frame
Hardware Field 1	(5 chars) (hex), pcm/ch8 = pcm start word
Hardware Field 2	(5 chars) (hex), pcm = frame interval, ch8/1553 = bus
Hardware Field 3	(5 chars) (hex), pcm = frame stop, ch8/1553 = rt
Hardware Field 4	(5 chars) (hex), pcm = word interval, ch8/1553 = msg
Hardware Field 5	(5 chars) (hex), pcm = word stop, ch8/1553 = word number
Hardware Field 6	(5 chars) (hex), pcm not used, ch8/1553 = tr bit
Hardware Field 7	(5 chars) (hex), spare user use

1.3.5 Special Notes for Embedded PCM Streams

If the user has embedded PCM stream 1 or 2 checked in the parameter editor GUI, then the appearance of the parameter editor GUI will change so that the spare words have meaning. This change is because it is assumed that the soft decom called “SoftDecom_StdEmbFrame” will be used. The soft decom is distributed and maintained by Lumistar. It is not mandatory to use it when the embedded streams are checked in the GUI.

The Lumistar soft decom uses the following ‘spare’ fields in the parameter information, as follows:

- Hardware Field 6 – Embedded Stream Start Frame
- Hardware Field 7 – Embedded Stream Start Word
- UserDefI1 – Embedded Stream Frame Interval
- UserDefI2 – Embedded Stream Word Interval
- UserDefI3 – Embedded Stream Frame Stop
- UserDefI4 – Embedded Stream Word Stop

These spare fields are stored in the parameter database as 0 based indexes. The GUI is a 1 based index for the word numbers and 0 based indexes for the frame numbers. The program does nothing with these values, so they can be used in any “custom” soft decom the user may wish to create.

For the Lumistar soft decom, the embedded frames have the following assumptions:

- Sub frames count up
- SFID counts from 0
- All embedded words have the same word length
- SFID bits are right justified (LSB is bit 0)
- Data is right aligned

- Data is bit aligned. Even if your data is word aligned, the soft decom will re-frame sync it.

1.3.6 Parameter Definition

The columns in the parameter definition portion of the database are defined as follows:

Word Name (19 chars) - This is the root word name for the parameter. It is possible that many parameters will have this word name assigned to them. For device parameters, all word names are assigned by the program, and begin with a 'D__' (D followed by two underscores), therefore, do not start a word name with this character. Only 18 characters are printable, the 19th character is a null character in the program. In the database file it is a space.

Parameter Name (19 chars) – This is the name for the parameter. It must be unique in the database. The user will reference this name when defining any processing. For device parameters, all parameter names are assigned by the program, and begin with a '$', therefore, do not use the '$' anywhere in a parameter name. Parameter names cannot start with a number, but must start with a letter or underscore, or any allowable characters. Only 18 characters are printable, the 19th character is a null character in the program. In the database file it is a space.

CVT Location (5 chars) – Is a hex number indicating the location in the Current Value Table for the root word. This location is assigned as the database is built. After the database has been built, it needs to be compiled, and all parameters in the database need to point to the corresponding CVT location. Normally, the parameters are sorted on Word Name, then on Parameter Name (see 1.3.7 on page 19). Then the PTT[5] location is assigned and the Mode Tag, Time Tag, and Second Tag are assigned.

PTT Location (5 chars) – A hex number indicating the location in the Parameter Tag Table for the root word. There can be up to 32768 parameters in each database. This PTT location is really an index into an array that stores all the information and process results for the parameters.

Tag Type (2 chars) – A decimal number indicating the type of tag. There are only two types currently defined:

0 – Normal Processing – Nothing special going on. All the bits are taken from the root word.

1 – Contiguous – The number of bits in the parameter exceed the number of bits in the root word, and they are contiguous in the stream

Unit of Measure Label (8 chars) – The text used for the unit of measure for the parameter.

Parameter Description (32 chars) – A short text description on the parameter.

[5] PTT - Parameter Tag Table

Classification (1 char) – Indicates the security classification for the parameter. For the Client displays, this will be looked at and if the option is turned on, the highest classification of all parameters on a display will be displayed in the security bars at top and bottom. The classification levels are as follows:

- NONE – No classification
- UNCLASSIFIED – Not classified
- FOUO – For Official Use Only
- FMS – Foreign Military Sales
- CONFIDENTIAL
- SECRET
- TOP SECRET
- USER1 //user def classifications
- USER2
- USER3
- USER4
- USER5
- USER6

Update Rate (3 chars) – Decimal integer number indicating the normal rate the parameter is expected to update in the stream, in hertz.

MSB First (1 char) – To solve the parameter, is the data in the root word MSB first? Defined as: 0 = no, 1 = yes.

MSB (2 chars) – A decimal number indicating the Most Significant Bit of the root word for the parameter. If all the bits were laid out horizontally, in base 2, the bit numbers would start with 0, and continue to 15. Bit 0 would be on the far right, bit 15 on the far left.

LSB (2 chars) – A decimal number indicating the Least Significant Bit of the root word for the parameter.

Parameter Number Of Bits (2 chars) – A decimal number indicating the number of bits used to solve the parameter. The Max is 16 bits.

Word Number Of Bits (2 chars) – A decimal number indicating the number of bits for the root word.

Supercom Process Mode (1 char) – In live mode, supercommutated data cannot normally be seen. To get a 'feel' for what the supercommutated data is doing, select one of the modes described below. If the scenario is to process data using supercom in live mode, then the supercom process mode is ignored. In playback, this mode is also ignored. The options for the supercom process mode are:

- SUPERCPROCMODE_FIRSTV – 0 – The first word value in the minor frame. Use this if there is no supercom.
- SUPERCPROCMODE_LASTV – 1 – The last word value in the minor frame.

- SUPERCPROCMODE_MINV – 2 – The least value of all appropriate words in the minor frame.
- SUPERCPROCMODE_MAXV – 3 - The greatest value of all appropriate words in the minor frame.
- SUPERCPROCMODE_AVGV – 4 - The average value of all appropriate words in the minor frame

Data Format (2 chars) – The data type for the number represented by the parameter for the root word. The types include:

- BINARY – Represented as binary data (normal for most parameters), like an integer.
- TWOS – The data is a twos complement number.
- SINGLE – The data is a 32-bit number, single precision (float) number.
- OFFSETBINARY – Offset binary representation.
- IEEEINTEGER – Bytes represented as IEEE 754 integer.
- IEEESINGLE – Bytes represented as IEEE 754 single.
- IEEEDOUBLE – Bytes represented as IEEE 754 double.
- MS1760 – Bytes represented as Mil Std 1760.
- ONES – Ones compliment number.
- LUT – Requires a Look Up Table.
- DOUBLE – The data is a contiguous 64-bit, double precision number.
- MW_DOUBLE – The 64-bit number using two 32-bit words.

eSpecialSolveFlag (2 chars) – Special ways to solve data for particular parameters. This is the catch all for future use and currently used for byte ordering, i.e., Endean issues. Most parameters will be set to zero for normal byte ordering. Currently, the special solve flag is for byte order changing. The following is what occurs to the raw concatenated data, just prior to applying the scale factor and offset. This will take care of Endean issues.

Enum		**Byte Re-ordering Result**		
Code	**Definition**	**16-bit**	**32-bit**	**64-bit**
0	no swapping	1 2	1 2 3 4	1 2 3 4 5 6 7 8
1	byte swapped only	2 1	2 1 4 3	2 1 4 3 6 5 8 7
2	word swapped only	n/a	3 4 1 2	3 4 1 2 7 8 5 6
3	byte & word swapped	n/a	4 3 2 1	4 3 2 1 8 7 6 5
4	word reversed	n/a	3 4 1 2	7 8 5 6 3 4 1 2
5	word reversed byte swap	n/a	4 3 2 1	8 7 6 5 4 3 2 1

Mode Param Name (19 chars) – If in order to solve a parameter, another parameter has to be evaluated first, and if that other parameter value has conditions (see mask conditional) for the required value, then this parameter value will be processed. Otherwise the processed value for this parameter does not change.

Mode Tag (5 chars) – A hex number pointing to the PTT location of the other parameter that needs solving before this parameter can be solved.

Mode Mask (4 chars) – A hex number representing the value that the mode tag must equal in order to cause this parameter to be solved.

Mode Mask Compare Type (1 char) – If there is a mode tag, then the mode tag value uses a conditional against the mode mask. If the conditional is met, the tag is processed. The following conditionals for the mode mask condition are:

- MODEMASK_EQUALS – 0 - //value == mask
- MODEMASK_GREATER - //value > mask
- MODEMASK_LESS – 2 - //value < mask
- MODEMASK_OR – 3 - //value || mask > 0
- MODEMASK_Bit_OR – 4 - //value | mask > mask
- MODEMASK_AND – 5 - //value && mask > 0
- MODEMASK_Bit_AND – 6 - //value & mask >= mask
- MODEMASK_Bit_XOR – 7 - //value ^ mask > 0
- MODEMASK_NOTEQUAL – 8 - //value != mask

Time Param Name (19 chars) – If the parameter has a separate time stamp parameter assigned to it, use this to identify it. Most PCM streams do not have these parameters, but there are many of them in IRIG 106 Chapter 8 and 1553 streams.

Time Tag (5 chars) – A hex number pointing to the PTT location of the time tag parameter.

Second Tag Name (19chars) – If the parameter requires a second parameter to solve (like two words are needed to solve the entire parameter).

Second Tag (5 chars) – A hex number pointing to the PTT location of the second tag.

Lookup Table Number (2 chars) – A decimal number representing the LUT to use in the lookup table list identified for the stream.

EUC (18 chars) – A decimal number (double precision) representing the Engineering Unit Conversion for the parameter. Once the bits are broken out of the root words, the resulting number is multiplied by this number to form a sub-result, then the Plus B is added for the final result. This is the NORMAL processing for the processing engine. Most parameters are solved with the linear equation: Mx + b.

Plus B (18 chars) – Decimal number (double precision), the +b used for Mx + b for normal processing of the parameter. For many parameters, the EUC will be 1.0 and the Plus B will be 0.0.

Hardware Field 0 - Hardware Field 7 (5 chars each) – Hex number representing different things about the root word and parameter. See paragraph 1.3.4 on page 14 for more information.

Preprocessed Formula (128 chars) – Parameters get their data from words. The user may elect to have all parameters associated with a word to have some processing performed first on the word. That formula goes here (see paragraph 4.3.1.5.4 on page 203).

Access Code (2 chars) (hex) – This field is used to enable/disable the displaying or calculation of the data for this parameter, depending on how the administrator has set up the system. There are 5 levels of access. The administrator will have the documentation that explains the value to enter. If the administrator is not using this function, put the value “FF” in this place.

Default Max Display Value (17 chars) (decimal) – Some of the graphical display widgets have maximum values they will show graphics for. When the popup list is used to add a parameter to a widget, the maximum display value will be set to this value.

Default Min Display Value (17 chars) (decimal) – Some of the graphical display widgets have minimum values they will show graphics for. When the popup list is used to add a parameter to a widget, the maximum display value will be set to this value.

Param Display Label (16 chars) – When the popup list is used to add a parameter to a widget, this will be the text used for the label (the default is to use the same text as the parameter name).

The following parts are not used by LDPS, and can be used anyway the user sees fit. They are user defined, normally used by the soft decom.

```
int StreamPart;  // part0 = main, part1 = embeddedstream1, part2 = embeddedstream2
int UserDefI1; //user defined, normally for soft decoms
int UserDefI2;
int UserDefI3;
int UserDefI4;
double UserDefD1;
double UserDefD2;
double UserDefD3;
double UserDefD4;
//future use
double Future1;
double Future2;
double Future3;
double Future4;
double Future5;
double Future6;
double Future7;
double Future8;
```

1.3.7 Database Compile

In order for the parameter database to work correctly, it must be compiled. This will put the date in the correct order with the correct pointers. If the user wishes to build a custom parameter database, using their custom software, then it is imperative that the user be very familiar with the parameter definitions discussed thus far in this document. The CVT and PTT locations are the key pointers for this endeavor. After the custom parameter database is created, it is recommended that the ASCII version of the parameter database be loaded into the Lumistar editor and then saved. This will compile it correctly and also save a binary version.

There are two CVTs and two PTTs for each stream used in LDPS. For example, a hardware device like a decommutator has a CVT/PTT pair for the actual serial stream data, as well as another CVT/PTT pair for device status. The CVT for the serial data stores information as 32 bit words into an array, or table. The CVT for device status stores data as 64 bit double precision numbers into an array. The PTTs for both serial data and device status have the same structure, and store a great deal of information about each parameter. The data for each parameter in the database comes from a CVT. The CVT in turn gets its data from the hardware. The CVT and its data are archived as 16 bit words. The PTT is used to process tags for display. As LDPS is a parameter name based application, there can be no duplicate parameter names in the database, nor can there be any duplicate word names in the database.

If the user wishes to also compile a custom parameter database, using a custom application, i.e., without using LDPS, then the steps outlined below are required to compile the database.

Step 1 – Read the vendor file(s) and extract the necessary information for each parameter in the file, based on the parameter definition described above. Note, don’t forget the Hardware Fields, as they are extremely important.

Step 2 – Go through the list and change any parameter names that are duplicates and ensure the parameter names and word names do not exceed 18 characters. Also ensure not to exceed 65,535 parameters. Ensure that the characters used are legal. Legal characters are ‘_’ (underscore), ‘A’ through ‘Z’, and ‘0’ through ‘9’. Any other character is illegal. No spaces are allowed between characters, only as fill characters at the end of the parameter and word names. Refrain from using parameter names that begin with “R1__” through “R12__” and “N1__” through “N12__” because of the syntax requirements for formulas typed in the client math engine. Parameters also may NOT start with a number. They MUST start with a letter or an underscore.

Step 3 – Sort the list of parameters by Parameter Name.

Step 4 – Go through the list of parameters and assign a PTT location for each tag. Number them in sequence, starting with 0. This is because derived parameters must have names that are legal to a compiler.

Step 5 – Go through the list of parameters and determine if each requires a Time

Tag, Mode Tag, or Second Tag in order to process. If so, find the respective Aux Tag Name in the list and assign the PTT location for that Aux Tag corresponding to where it was found in the list. For example, the current parameter name is INLAT (PTT Loc = 7), and it requires a Time Tag parameter called MS1010200. Go through the list and find the parameter name MS1010200. Searching finds it at location 61 (therefore its PTT Location is 61). Then insert 0003D (HEX) (61 decimal) for INLAT's Time Tag value. If an Aux Tag is not used, the Aux Parameter Name is blank (18 space characters) and the PTT Location for the Aux Tag is set to 00000. If the Mode Aux Tag is not used, set the Mode Mask to FFFF.

Step 6 – Sort the list on Word Name.

Step 7 – Go through the list, stating with CVT Location 0. Assign the parameter CVT Location. Go to the next parameter. If the Word Name for that parameter is different than the Word Name for the last parameter, bump the CVT Location.

Step 8 – Sort the list on Parameter name.

Step 9 – Write the required Header data to the dbase file.

Step 10 – Go through the list, formatting the data for each parameter into a string and write each string to the file.

The validate the user's custom software, compare the results with a database generated with LDPS to see if they look similar, particularly the ***, the *.*, and the alignment of characters for the parameters. If so, then there's a good chance the database will work (assuming the pointers to the PTT and CVT were done correctly).

1.3.8 Parameter Database Import Program

The data dictionary, or parameter database, is the most critical part of the system, and is generally the most difficult to create. If it is not correct, or has errors, the data is meaningless. Normally, the master data dictionary is created by a single person, or a group of people. The other major component of the project, the display and processing functions, will most likely have the same format as the master data dictionary. The ultimate goal then is to put the master information from the data dictionary into a form that the display and processing system can understand. There are two ways to do this. The user can either manually enter (type in) the information into the needed format, which is very error prone, or employ some form of software program to read the master and translate that data into the form the processing system requires.

The LDPS server offers such a program. To invoke it, from the server, select Tools, and then LdpsImportPdbase (*Tools → LdpsImportPdbase*). This tool can also be invoked in a stand-alone fashion from the ../User Tools/ subdirectory. This program will not import every type of format likely to be encountered, but it will handle most of them. If it will not support a particular master, the user can always write their own custom application, using the supplied LdpsPdbaseConvert.dll.

1.3.8.1 Requirements

In order for the LDPS parameter database import function to work correctly with the user's master database file, there are a few basic rules that must be followed. This may require a little up front preparation by the user. Using a tool like Microsoft Excel to create the input file with the proper format is relatively easy. The input file needs to comply with the following requirements:

1. The Pro version of LDPS is required to import a database file.
2. The program and its DLL (LdpsPdbaseConvert.dll) must be in a subdirectory of LDPS.
3. The fields in the master file must be tab delimited.
4. The first line of the master file must contain the field meanings.
5. The second and subsequent lines must be the data.
6. The user must have a thorough knowledge and understanding of the LDPS parameter data base items.

1.3.8.2 The Users Primary Task

The main task of performing the translation is matching the fields in the master file with those fields that LDPS requires. If the master file does not have a required field, the missing field LDPS requires is calculated/generated based on the fields the master file actually has. The users task is to fill out the PCM frame information, and match up the required fields once that is done, the master file may be imported.

1.3.8.3 Program Limitations

The import function uses the supplied LdpsPdbaseConvert.dll. However, the user may write their own program if the supplied DLL does not do enough. The supplied DLL has the following limitations:

- It will not concatenate parameters that do not use contiguous word bits.
- It will truncate all parameters that are greater than the word length to the word length if the bits are not contiguous.
- Maximum parameter length is 32 bits. Longer parameters are truncated to 32 bits. Unless the type is double or byte-swapped double, with contiguous true.
- Parameter names are truncated to 18 characters.
- Illegal parameter name characters are replaced with an underscore.
- Parameter names are set to upper case.
- If the word length is greater than the common word length, it is truncated to the common word length.
- Functions and derived type parameters are not created.
- Duplicate parameters are discarded.
- The first blank line encountered ends the import session.

If the import program modifies any parameter, the user is notified of the offending parameter and the line number, as well as the line text from the master file.

1.3.8.4 Missing Fields

If there are missing fields in the master file, then the program will attempt to fill out what is needed based on what the master file does have. At a bare minimum, the PCM word number is required. If there is more than one minor frame per major frame, then the PCM frame number is also required. Other than that, the import function will work correctly with none of the other matching fields filled in.

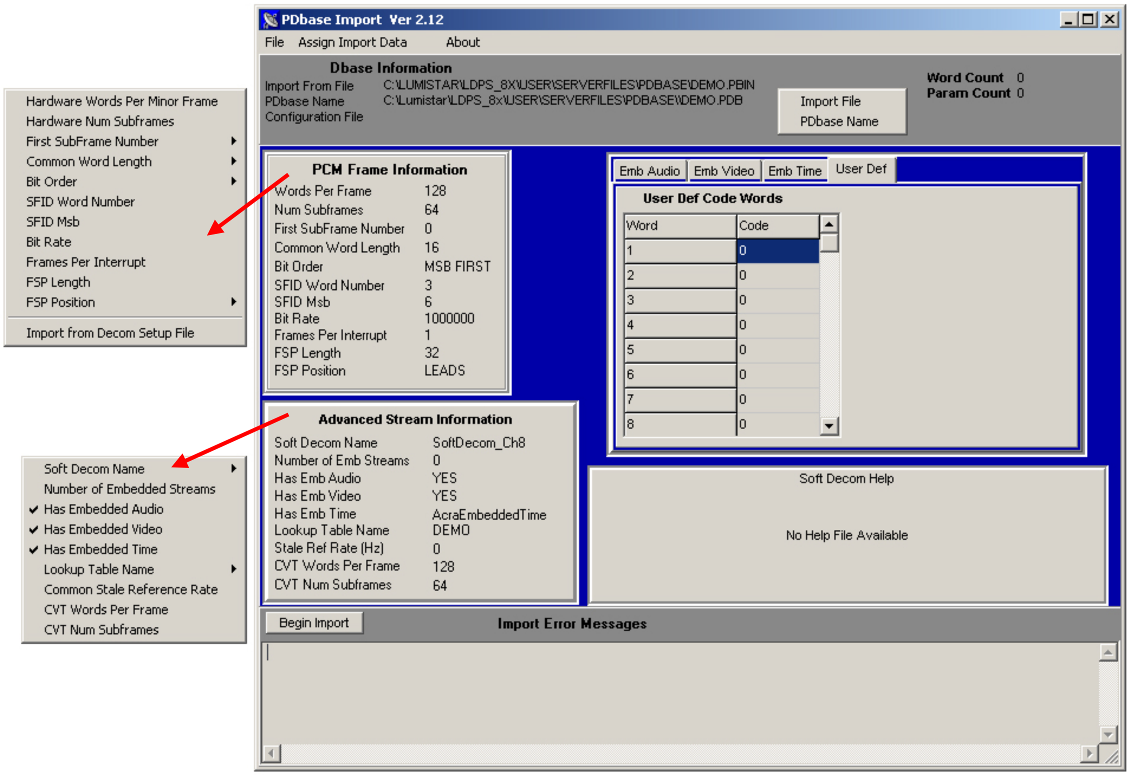
The following describes the way the calculations to fill in missing fields is performed. The steps are performed in the following order as listed:

- **Parameter and Word Name** – If there is no Parameter Name field, the Parameter Name is assigned to the Word Name field. If there is no Word Name field, one is created using the frame number and word number, i.e. FR1WD32.
- **Data Format** – The data format of the number will default to BINARY if the format field is blank, or no matching text was found.
- **Bit Position** – Many rules:
    - If the word number of bits is greater than the common word length, it is truncated to the common word length unless the check box indicating multiword parameters are contiguous is selected.
    - If the parameter number of bits is greater than the word number of bits, it is truncated to the word number of bits if the multiword parameter checkbox is **not** checked. If it is checked, and if the parameter number of bits is greater than 32, then it is truncated to 32.
    - If the Parameter Number Bits is 0, and the MSB is 0, and the LSB is 0, then Parameter Number Bits is set to the word number of bits.
    - If the Parameter Number Bits is not 0 and the MSB is not 0, then LSB is set to MSB, and Parameter Number Bits is set to "+1"
    - If the Parameter Number Bits is not 0 and the MSB is 0, then LSB is set to MSB and Parameter Number Bits is set to 1.
- **EUC (Scale Factor)** – Again, more rules:
    - If EUC is 0, and Max Value not filled in, and Min Value not filled in, then:
        - EUC is set to 1
        - If Data Format is a twos compliment type, then:
            - Max Value = $(2^{\text{Parameter Number Bits}} - 1) - 1$
            - Min Value = -Max Value – 1
        - If Data Format is **not** twos compliment type, then:
            - Max Value = $(2^{\text{Parameter Number Bits}}) - 1$
            - Min Value = 0
    - If EUC is 0, and Max Value is filled in, and Min Value is not filled in, then: EUC = (Max Value – Min Value) / $(2^{\text{Parameter Number Bits}}) - 1$;
    - Hardware Settings:
        - If Word Interval $\geq$ Words Per Sub Frame, Then Word Interval is set to 0.

  - If Frame Interval ≥ Number Sub Frames, Then Frame Interval is set to 0.
  - If Word Start ≥ Words Per Sub Frame, Then Word Start is set to 0. Note. This will generally cause future errors because of duplicate Word Names and location in the frame.
  - If Frame Start ≥ Number Sub Frames, Then Frame start is set to 0. Note. This will generally cause future errors because of duplicate Word Names and location in the frame.
- The Parameter Update Rate is solved based on the hardware settings and the bit rate if there is no import field for it.

1.3.8.5 The Program's GUI

The Graphical User Interface (GUI) for the parameter database import function is shown in Figure 1-3 below. By right clicking the mouse, the resulting menus are shown.



Dbase Information

Import From File C:\LUMISTAR\LDPS_8X\USER\SERVERFILES\PDBASE\DEMO.PBIN
PDbase Name C:\Lumistar\LDPS_8x\USER\SERVERFILES\PDBASE\DEMO.PDB
Configuration File

Word Count 0
Param Count 0

PCM Frame Information

Words Per Frame	128
Num Subframes	64
First SubFrame Number	0
Common Word Length	16
Bit Order	MSB FIRST
SFID Word Number	3
SFID Msb	6
Bit Rate	1000000
Frames Per Interrupt	1
FSP Length	32
FSP Position	LEADS

User Def Code Words

Word	Code
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0

Advanced Stream Information

Soft Decom Name	SoftDecom_Ch8
Number of Emb Streams	0
Has Emb Audio	YES
Has Emb Video	YES
Has Emb Time	AcraEmbeddedTime
Lookup Table Name	DEMO
Stale Ref Rate (Hz)	0
CVT Words Per Frame	128
CVT Num Subframes	64

Soft Decom Help

No Help File Available

Import Error Messages

Figure 1-3 Parameter Database Import Window & Menus

The import program remembers the previous configuration settings, so if one is continually doing the same master file import, one only need enter the data once. Different settings can be recalled and saved as well. The users' files are saved by default in the program directory with the file extension "IPDB."

The first step in setting up the import is to enter all of the **PCM Frame Information**. Right click on the window and enter all of the parameters in the resulting menu shown in the figure above. Optionally, one may import all of these settings from the decommutator setup file. This action will fill in all the fields at once.

If a Soft Decom is to be used, then check the box "Uses a Soft Decom" and manually enter the information as required. For software decommutators, the CVT size will be filled in automatically, based on other "Advanced" data settings. For hardware decommutators, the CVT size will be the same as you PCM frame size.

NOTE. If the hardware decommutator frame size is changed, the CVT size will automatically revert to the hardware decommutator frame size. The CVT size information must be entered AFTER the hardware decommutator information is filled in.

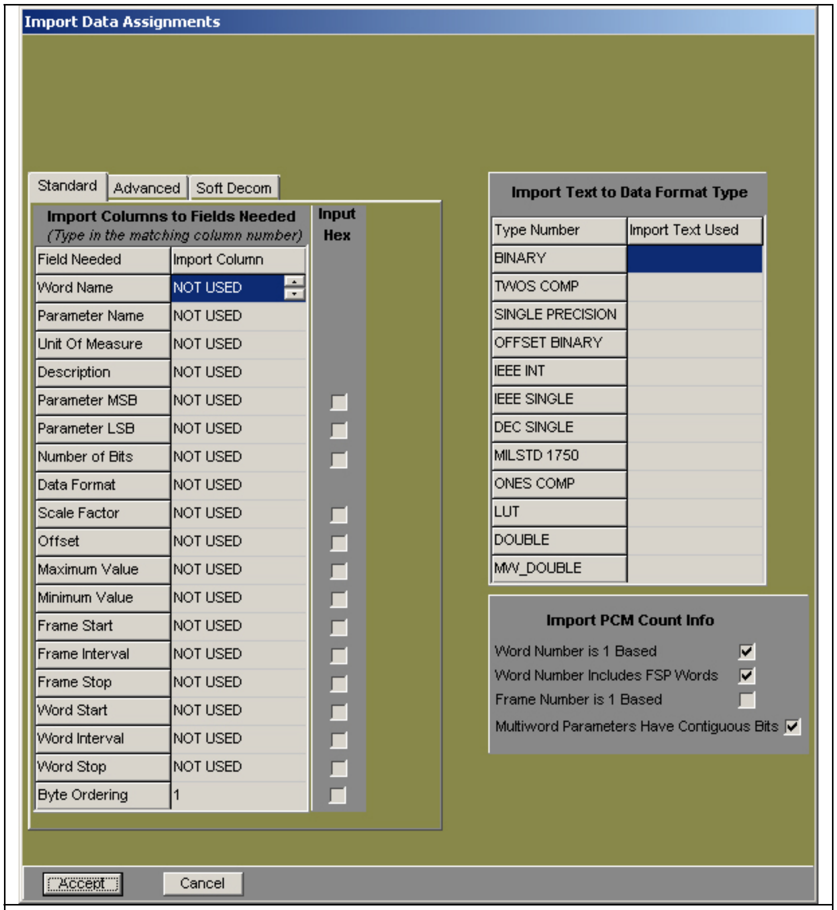
The next step is to select an import file. Right click in the **Dbase Information** area and select the import file. When selected, the **PDbase Name** will be assigned the same name as the import file. The name may be overridden by right clicking and selecting a different name. The locations of both the import and export files are retained by the application. It is suggested however, that they be kept separate.

Proceed by assigning the fields in the import file to their corresponding required fields in the export file. To do this, click on the **Assign Import Data** command in the programs main menu. This will result in the window shown in Figure 1-4 below. If the import file name has not yet been established, the program will prompt the user for one.

Across the top of the **Import Data Assignments** window are the first two lines of the import file, which should contain the field header and the first data line of the import file.

The fields that LDPS uses are shown lower left in the Figure 1-4 below. There are three tabs with information that can be imported. The **Standard** tab shown above has the most common information. Most users can ignore the **Advanced** and **Soft Decom** tabs shown in Figure 1-5 on page 29. The task then is to match the fields (columns) of the import file with the respective field names in the export file. Just type them in, or use the scrollbar. Fill in as many as possible. The number entered must be the column number for each field in the import file. For the number type fields, next to each field is a check box. Check the box if the import data field is in hex (base 16), or unchecked if it is base 10.

In the upper right of Figure 1-4 below is the **Import Text to Data Format Type** area. This is a special field where the user identifies what text was used in the import file to define a particular data type. For example, a twos complement data type could be identified with the text, “two’s” or “2’s,” or both (note: separate each text with a comma). Any text not found results in the Data Format being set to BINARY. All text in this field is converted to upper case when read from the file. Also note that no spaces in the field are allowed. For example, a “2’s Comp” will revert to BINARY because the space is removed by the program and the resulting search for “2’sComp” will yield in not text being found. Either replace the space with an underscore or just remove the Comp from the field.



Import Data Assignments

Standard | Advanced | Soft Decom

Import Columns to Fields Needed
(Type in the matching column number)

Field Needed	Import Column	Input Hex
Word Name	NOT USED	
Parameter Name	NOT USED	
Unit Of Measure	NOT USED	
Description	NOT USED	
Parameter MSB	NOT USED	□
Parameter LSB	NOT USED	□
Number of Bits	NOT USED	□
Data Format	NOT USED	
Scale Factor	NOT USED	□
Offset	NOT USED	□
Maximum Value	NOT USED	□
Minimum Value	NOT USED	□
Frame Start	NOT USED	□
Frame Interval	NOT USED	□
Frame Stop	NOT USED	□
Word Start	NOT USED	□
Word Interval	NOT USED	□
Word Stop	NOT USED	□
Byte Ordering	1	□

Import Text to Data Format Type

Type Number	Import Text Used
BINARY	
TWOS COMP	
SINGLE PRECISION	
OFFSET BINARY	
IEEE INT	
IEEE SINGLE	
DEC SINGLE	
MILSTD 1750	
ONES COMP	
LUT	
DOUBLE	
MW_DOUBLE	

Import PCM Count Info

Word Number is 1 Based ☑
Word Number Includes FSP Words ☑
Frame Number is 1 Based □
Multiword Parameters Have Contiguous Bits ☑

Figure 1-4 Import Data Assignments Window - Standard Tab

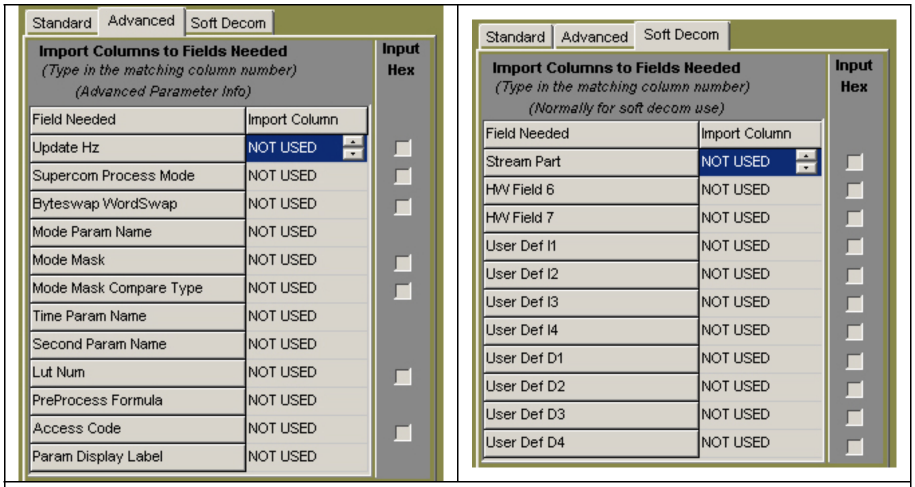
The last step is to fill out is the gray box called, **Import PCM Count Info** shown lower right in Figure 1-4. This area allows the user to clarify how numbers are counted in the import database file. Users often count numbers in different ways. Some have word numbers counting from 0, while others count from 1. The same is true with frame numbers. Also, some users don't count the frame sync pattern words as word numbers, while others do. Therefore, the user must set these check boxes to match the way the import file uses word and frame numbers. In addition to the count type information, there is a checkbox for multiword parameters with bits that are contiguous across words. If multiword parameters are not used, it is still ok to leave it checked (It is checked by default). However, if multiword parameters are used and have bits that are not contiguous in the stream, then this checkbox **must not** be checked, or erroneous data will result.

After all of the information has been entered, click the **Accept** button on the Import Data Assignments window. This will take the user back to the main Parameter Database Import window shown in Figure 1-3 on page 25. Once back to the main window, notice that another box below the Dbase Information and PCM Frame Information area has appeared. This is where the **Begin Import** button is, as well as a list of modifications or errors that were found when converting the file. Press the Begin Import button and the data will be imported, converted, and saved. During the conversion process, the Dbase Information box will display the current count of Words and Parameters it has created. When complete, a trill sound will emit from the speaker and the Import Error Messages will fill up the white area below the button, and state it has FINISHED IMPORT TASK.

Every error and/or modification will have the error/modification message and the offending line number from the import file, as well as the text from the import file line. Use Excel at this point to view and correct the offending line. If not, the line will be modified or omitted (errors are omitted from the conversion). Fix any errors or modifications and move the ".PDB" and ".PBIN" files to the LDPS PDbase directory for use.

As shown left in Figure 1-5 on page 29, the **Advanced Tab** for column assignments has a few fields that rely on a number (not text) to set the meaning. These fields are defined as follows:

- Supercom Process Mode – For super commutated parameters that have supercommutation turned off, the parameter solved for can be one of the following values:
  - 0 – The first word value in the minor frame is used.
  - 1 – The last word value in the minor frame is used.
  - 2 – The word with the least value in the minor frame is used.
  - 3 – The word with the most value in the minor frame is used.
  - 4 – The average value of all words in the minor frame is used.
- Byte swap Word Swap – A value other than 0 will be interpreted as meaning the word is byte swapped and/or word swapped.
- Mode Mask Compare Type – If a mode parameter is used, the mask is applied to the parameter value to trigger whether the parameter is to be updated. The following mask compare types are used:
  - 0 – AND (value && mask)
  - 1 – GREATER (value > mask)
  - 2 – LESS (value < mask)
  - 3 – OR ((value | mask) > 0)
  - 4 – XOR ((value ^ mask) > 0)
- Access Code – For supervisor use only. If not a LDSP supervisor, it is best to leave this field as NOT USED. Documentation for this field is given to the LDPS supervisor only.



Standard | Advanced | Soft Decom

Import Columns to Fields Needed
(Type in the matching column number)
(Advanced Parameter Info)

Field Needed	Import Column	Input Hex
Update Hz	NOT USED	□
Supercom Process Mode	NOT USED	□
Byteswap WordSwap	NOT USED	□
Mode Param Name	NOT USED	
Mode Mask	NOT USED	□
Mode Mask Compare Type	NOT USED	□
Time Param Name	NOT USED	
Second Param Name	NOT USED	
Lut Num	NOT USED	□
PreProcess Formula	NOT USED	
Access Code	NOT USED	□
Param Display Label	NOT USED	

Standard | Advanced | Soft Decom

Import Columns to Fields Needed
(Type in the matching column number)
(Normally for soft decom use)

Field Needed	Import Column	Input Hex
Stream Part	NOT USED	□
HW Field 6	NOT USED	□
HW Field 7	NOT USED	□
User Def I1	NOT USED	□
User Def I2	NOT USED	□
User Def I3	NOT USED	□
User Def I4	NOT USED	□
User Def D1	NOT USED	□
User Def D2	NOT USED	□
User Def D3	NOT USED	□
User Def D4	NOT USED	□

Figure 1-5 The Advanced & Soft Decom Tabs

Shown upper right in the figure above, the **Soft Decom Tab** for column assignments is totally dependant on how the developer of the soft decom wants them used. Consult the developer of the soft decom for what goes into these fields.

⚠	If you don't have a software decommutator, Don't Modify These Fields!

For soft decoms developed by Lumistar, the fields have the following meaning:

- Stream Part – The stream the parameter belongs to:
  - 0 – Main PCM stream
  - 1..n – Embedded stream 1…Embedded stream n
- HW Field 6 – Embedded Stream Start Frame, 0 based index
- HW Field 7 – Embedded Stream Start Word, 0 based index
- User Def I1 – Embedded Stream Frame Interval
- User Def I2 – Embedded Stream Frame Stop, 0 based index
- User Def I3 – Embedded Stream Word Interval
- User Def I4 – Embedded Stream Word Stop, 0 based index
- User Def D1 – n/a
- User Def D2 – n/a
- User Def D3 – n/a
- User Def D4 – n/a

1.4 Time

For most projects, it is very critical that time be accurate. The problem with Windows NT (and even worse on other versions of Windows) is the CPU time is not very accurate and the resolution is only 0.054945 seconds (less than 20 hertz). Usually, for this type of system use, at least one data stream will have it's own embedded time source, which is much more accurate. The Server program allows the user to determine the source of the System Time.

There is a method to get time from the CPU that is higher in resolution, but it is expensive in CPU clocks. This time is used for slower processes, like time-stamping the data broadcast to the Client(s), calculating the rates between processes, etc.

1.4.1 System Time

System time is used for time stamping the archived data and for time stamping the arrival of hardware status data from devices.

1.4.2 Stream Time

Stream time is used to time stamp the serial data in the stream. Normally, stream time is generated from the stream source (whether it be an IRIG time generator or embedded in the serial stream). If there is no time source for the stream, stream time is obtained by using the higher resolution CPU clock.

1.4.3 System Time Source

LDPS has the ability to select which time source is used for the System time. System Time can come from the high-resolution CPU clock, or any of the valid streams. If the CPU clock is used, the update rate is only 20 hertz if a project is loaded, so be warned that for faster archiving rates, the time stamps will be the same. The system time on the display will be in yellow if CPU clock is used for the system time.

If a project is not loaded, then the System time source is set to the CPU clock, and updated at a 1 hertz rate. Time stamping is not critical if the project is not loaded.

1.5 Directory Structure

There are many files and directories created and used by LDPS. Some directories can be changed by the user via the options, while others are fixed. By default, all directories used are under the "LDPS_8x" directory where the program is installed.

Under the root directory are seven (7) main subdirectories, which are really just categories of file types.

- Bin – This subdirectory is for DLL[6]s that support user hooks, like soft decoms, aux inputs, etc.
- Documentation – This subdirectory is for documentation for the system, as well as source code and examples for user hooks.
- System – This subdirectory is for files belonging to the system, like error logs, program options, last setups, etc.
- Include – This subdirectory contains more source code for user hook APIs.
- User – This subdirectory contains the bulk of the files the user will be most interested in. Things like displays, setup files, archive data, etc.
- User Tools – This subdirectory contains a variety of tools and additional source code.
- Driver Tools – This subdirectory contains some troubleshooting and debugging tools intended for use during factory support sessions.

1.5.1 LDPS ROOT Files

Applications and DLLs for LDPS are stored here. In addition, some user application tools store configuration data here (because they don't know anything about LDPS). When LDPS is first installed, the following files are stored here. Unless specified, these applications are started by the Server program, but can be operated in stand-alone operations:

- Ldps8xServer.exe – The server program (invoked by the user)
- Ldps8xClient.exe – The client program (invoked by the user)
- EditLut.exe – Editor to edit lookup table
- EditPDbase.exe – Editor to edit parameter databases
- EditProject.exe – Editor to edit projects
- ViewLogs.exe – Viewer to view error logs
- A1553BusMonitor_8x.exe – Application for 1553 bus monitor
- Ldps8xCustomNonSerial.exe – Application for Custom Aux input
- Ldps8xCustomSerial.exe – Application for Custom Aux input
- Ls22V3_8x.exe – Application for this type card.
- Ls23V2_8x.exe – Application for this type card
- Ls25V1_8x.exe – Application for this type card
- Ls25V2_8x.exe – Application for this type card
- Ls50_8x.exe – Application for this type card
- Ls71_8x.exe – Application for this type card
- Ldps8xCustomNonSerial.dll – Support for the application
- Ldps8xCustomSerial.dll – Support for the application
- Ls22V3_8x_Dll.dll – Support for the application
- Ls23V2_8x_Dll.dll – Support for the application
- Ls25V1_8x_Dll.dll – Support for the application
- Ls25V2_8x_Dll.dll – Support for the application

[6] DLL – Dynamic Link Libraries.

- Ls50_8x_Dll.dll – Support for the application
- Ls71_8x_Dll.dll – Support for the application
- RunClient.bat – Batch file to run the client (called by the server if user launches client from the server)
- H261.dll – View H.261 video
- VLMPEGVideoDecoder.dll – View MPEG video using Elecard CODEC

1.5.2 ROOT\BIN

These files are not under option control (user doesn't define their location.) They must reside in the correct subdirectory. These are the user-hook DLLs created by the user. If they are used, place the user hook DLL files in the respective directory for use. The program will only accept files from here.

- **1553Dlls** – For 1553 input DLLs.
- **ArchiveDlls** – For archive/playback DLLs.
- **AuxInputDlls_NonSerial** – For custom aux input, non serial type DLLs
- **AuxInputDlls_Serial** – For custom aux input, serial type DLLs
- **EmbTimeDlls** – For embedded time DLLs
- **ExportPdbDlls** – For export pdbase DLLs
- **SimLs22Data** – Don't modify this directory. Simulated data for Ls22 is kept here.
- **SoftDecomDlls** – For soft decom DLLs

1.5.3 ROOT\DOCUMENTATION

This is the directory most users ignore. It contains the documentation for the system. Under this subdirectory are more subdirectories, as follows

- **Drawings** – Diagrams of flow for the system.
- **ICD Related** – Many subdirectories under here with documentation for various parts of the system that the user may hook in to. Also contains example code.

1.5.4 ROOT\SYSTEM

Contains files used by the system. With the exception of the Help subdirectory, the user may delete all the files in the rest of the subdirectories, thus restoring the default settings. The other directories are blank when LDPS is first installed on the system. The directories are as follows:

- **ErLogs** – Every application and DLL component of LDPS generates a ".log file" that contains information about the application or DLL. The user may be called upon to send these files to Lumistar if there are problems that can't be resolved via telephone support. These files are erased and started anew each time the particular application/DLL is started.
- **Help** – Do not erase the files here. They are help files used by the math engine.

- **Options** – These are where the user options are stored. If deleted, the options will be back at their default state.
- **HardwareConfiguration** – These files are for things like last setup, last position, etc. for the various applications. If deleted, the applications will be back at their default position and default setup values.

1.5.5 ROOT\USER TOOLS

This directory allows the user to store any executable file here, and it will become part of the dropdown list of Tools launched from the Server or Client. A few are given to start with, but the user may add more. The user tools are not tied to LDPS, but are rather stand alone applications.

There are a several subdirectories here with a few things some users may want. They are ICD related and belong more to the client than the server.

- **MathFunction** – These files are needed to write user math functions.
- **PDbaseConvert** – These files are needed to write custom pdbase import or export programs.

1.5.6 ROOT\USER

This set of subdirectories is where most users will be interested. There are four main subdirectories, or categories, here:

- **ArchiveData** – This is where data is archived on the Server. Subdirectories under here are created when data is archived from a project.
- **ClientFiles** – Many subdirectories under here are used by the cient.
  - **Displays** – Display pages and lists
  - **DpyArchives** – Display page ASCII archive files
  - **DpyBackgrounds** – Display page background BMP files
  - **Enumerations** – Enumeration files
  - **EventLogs** – Display page event logs
  - **Formulas** – Derived formula and function lists
  - **FunctionDlls** – Function DLLs and their associated help files. When a function is created, it goes into this directory.
  - **Hardcopies** – Hardcopy images generated by the Client go here.
  - **Maps** – Map BMB files for maps, targets, waypoints.
  - **WaveFiles** – WAV files for audio triggers.
  - **AvArchives** – Default directory where audio/video files are recorded
- **LDRS Files** – Now implemented.
  - **ProcessLists** – Stores lists of parameters to process during LDRS runs.
  - **ProcessReports** – Stores the results of running the process, the data and a report.
- **ServerFiles** – Many subdirectories under here used by the Server.
  - **DDbase** – Device setup files, various extensions
  - **Hardcopies** – Hardcopy images generated by the Server or device applications go here.

- **Misc User** – Nothing yet, future use
- **PDbase** – The parameter database files, both ASCII and binary versions
- **Project** – The project files

1.5.7 File Extensions

This is a list of file extensions used by LDPS:

LOG – Error logs
CFG – Program configuration files
SIM – If the file exists, the hardware DLL goes into simulate mode
PDB - PDbase ascii format
PBIN - PDbase binary format
PRJ - Project
LUT – Lookup tables
ARK - Archive file for serial data
STAT - Archive file for non-serial status data
NSDAT – Archive file for non-serial data
ENM - Enumerations
DPY – Display pages
DPS – Display lists
PGLOG – Display page archive file
MAP – 2D Map files
EVT – Event logs
TXT – DLL help file
WAV – Audio trigger file
DER – Derived formula lists
FUN – Function lists
FSNP – View Frame Dump snap shot files

BMP – Hardcopy image
JPG – Hardcopy image
1553 – ASCII setup file for 1553 application
B50 – Binary setup file for Ls50 application
LS25 – ASCII setup file for Ls25 v1 application
LS25V2 – ASCII setup file for Ls25 v2 application
LS22V3 – ASCII setup file for Ls22 v3 application
LS23V2 – ASCII setup file for Ls23 v2 application
LSDAC– ASCII setup file for Ls71 application
CASX– ASCII setup file for Custom Aux Input Serial application
CANSX– ASCII setup file for Custom Aux Input Non Serial application
MPEG – Video file recorded (MPEG)
P64 – Video file recorded (H.261)
CVSD – Audio file recorded (CVSD)
PCM – Audio file recorded (PCM)

1.5.8 Special Note for Remote Clients

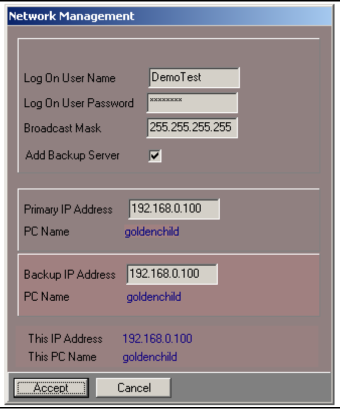
For remote clients, the user needs to be aware of where files are stored and retrieved. The parameter database files and project files are all located on the server. The display files are located on the local machine by default. These are the ".DPY," ".DPS," and ".ENM" files, as well as the widget DLL and TXT files. Also the ".EVT" files are stored on the local machine. The reason for this is because the files in LDPS are not part of a relational database. If there are many clients with many users, they all may want the same name for a file, but have different information for them. Also, different clients may log different events. It is up to the project manager or system administrator to establish the naming convention for these files. If everyone is using the same display list, display pages and/or enumerations, then the system user options should be edited to have the default directories point to the desired places.

1.6 Network Management

Network management is very important in LDPS and is necessary for the server-to-client communication mechanism. The client programs must know who to request data from, and also be able to read project files. During operation, the server broadcasts data out to the world on the net, or on the user defined subnet mask, if it was optionally set (the default is 255.255.255.255). The client programs pick up the data from the network and uses it as needed. The clients' configuration options define the network address of the server it accepts data from, as well as the address of the Backup Server, if one has been assigned.

To set up the network for the system, perform the following steps:

1. The user must have administrator privileges.
2. Start the server program and select System → Options and select the Operations tab. Make sure that the **Disable Network** checkbox is NOT checked (see paragraph 2.11.2 on page 55).
3. Restart the server program and select System → NetAdmin. From the resulting window, shown in Figure 1-6 below, enter in **the Log In User Name** and **Password**. Also, if the backup server is to be used, check the **Add Backup Server** check box, and enter the **Backup IP Address** for the back up server. Note: The password is not displayed when entered, so type carefully. The resulting password in the "LdpsNetServer.cfg" file is encrypted so don't forget it.
4. Close the server program.
5. Start the client program and select System → Options and select the Operations tab. Ensure that the **Disable Network** check box is NOT checked.
6. Restart the client program and select System → Options and select the Directory tab. Click on **Server Name**, and navigate to the PC and directory where the server program resides. Note: if the server is on the same PC as the client, the user still must navigate through the network neighborhood with the browser so the PC name can be picked up. Do the same thing with the rest of the optional directories on this tab. The client program picks up the "LdpsNetServer.Cfg" file from the server's options directory and saves it to its local options directory as the file, "LdpsNetClient.Cfg". This step allows the client program to start up without the server program being running and still have the information it needs when the server starts up. The client program can therefore change which server it wants to get data from (useful for live/playback operations).
7. Close the client program.



Network Management

Log On User Name: DemoTest
Log On User Password: ********
Broadcast Mask: 255.255.255.255
Add Backup Server ☑

Primary IP Address: 192.168.0.100
PC Name: goldenchild

Backup IP Address: 192.168.0.100
PC Name: goldenchild

This IP Address: 192.168.0.100
This PC Name: goldenchild

Figure 1-6 Network Management Setup

8. Repeat step 6 for each client on the network. Note: one can just as easily copy the "LdpsNetClient.Cfg" file from the Ldps_8x\System\Options directory on the first machine to all the other client machines. Clients must be able to access the server, both reading and writing. The default directory for the install is in Lumistar. If the clients are not to have access to this directory, then don’t accept the default install directory. The clients will also need the log on name and password for the server. The clients do not have to have administrative privileges, but they do have to have read/write access to the directories set up for LDPS.

After performing the steps outlined above, copy the “LdpsNetServer.cfg” file from the Systems\Options directory on the server to the backup server, if a backup server is used.

For system administrators that need to restrict things, the following files must have full access by the user, including read/write/delete, and not have the read only attribute set.

winnt/system32/drivers
- ls50drvr.sys
- windrvr6.sys
- mslxa.sys

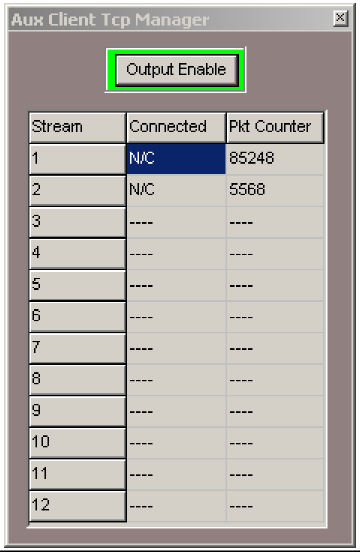
winnt/inf
- lumistar.inf
- mslxa.sys
- windrvr6.inf

winnt
- ad.ocx
- agauge.ocx
- air.ocx
- car.ocx
- joystick.ocx
- knob.ocx
- led.ocx
- lgauge.ocx
- mmap.ocx
- numled.ocx
- odometer.ocx
- percent.ocx
- selector.ocx
- slider.ocx
- strip.ocx
- toggle.ocx
- 98_windriverload.bat
- bcbsmp60.bpl
- borlndmm.dll
- cc3260mt.dll
- nmfast60.bpl
- nt_widriverload.bat
- pci_dump.exe
- pci_scan.exe
- rtl60.bpl
- stlpmt45.dll
- vcl60.bpl
- vcldb60.bpl
- vcljpg60.bpl
- wd_utils.dll
- wdreg.exe
- wdreg16.exe
- windriverload.bat
- windrvr6.sys

Also, the entire ldps_8x and subdirectories, wherever it may be, must have full access by the user, including read/write/delete, and not have the read only attribute set.

WARNING
The user must close the client program and restart it if any changes are made to the server it receives data from.

For the auxiliary TCP manager on the server (shown in Figure 1-7 right), one only needs to turn it on *(View → Aux TCP Client Manager)*. This manager displays the number of connections to each stream, and the packet counter for data going out. One may turn the data on and off (Output Enable) with this manager. Note: this manager only works if the network option is turned on in the server options.



Stream	Connected	Pkt Counter
1	N/C	85248
2	N/C	5568
3	----	----
4	----	----
5	----	----
6	----	----
7	----	----
8	----	----
9	----	----
10	----	----
11	----	----
12	----	----

Figure 1-7 Aux Client TCP Manager

1.6.1 Playback & Network Issues

For playback, control is through the server program. If the clients need playback data, a server must be controlling it. However, assume a test is running with a client assigned to the main server, and then later during the test, one of the clients needs to replay part of the test that was recorded, without affecting the rest of the clients. To affect this, the client now needs to close the project, and be assigned to a different server (normally one on the same machine the primary server is running on). Then the alternate server can be put into playback mode and the client can connect and get data that has already been archived by the primary server. The easiest method to quickly change from one server to another is to delete the net server cfg file. One would want to pre plan this, and make a subdirectory that has the network configuration using the primary server files, and a playback configuration with those files in it, and then just quickly copy the correct 'mode' configuration back and forth as one changes. This scenario will not occur for most users, but is quite handy for those that need it. Otherwise, one must manage the network as outlined above.

Note. When making any changes to network settings, the applicable program must be restarted before the changes take affect.

1.6.2 Notes on User Options

Each client must be pointed to the correct machine to access shared files (like project files, archived files, database files, etc). The correct path is established using the browser navigating through the network neighborhood so it can attach the network machine name as well as the directory to the desired location. Mapping drives (as used in previous versions of LDPS) is no longer used. See the Network Administrator to ensure that the correct permissions are set to read and write files of any type in the directories used by LDPS.

Note. There is also a Network Wizard in the User Tools directory (or select it from the Tools menu on the server). Use the Network Wizard to setup the network as described here. See paragraph 7.14 on page 255 for more information.

2 The LDPS Server

The server program collects data from external sources, like a decommutator, network, disk, etc. It associates the data sources with stream numbers that are tied to the hardware. Next the server archives the data and arranges the data into a normalized format. Then the data is passed on to the client(s) for further processing and display. To support these functions, the server manages all of the collection programs associated with the hardware in the system. As new technologies are developed for collecting data, only the server program need be modified. The primary functions of the server are discussed in more detail in the following paragraphs.

2.1 Data Collection

As mentioned, the server collects data from external sources. These sources can be anything that can feed data into a personal computer.

In LDPS, data sources are divided into two basic categories. Serial type data and status type data. Serial type data comes from devices like a decommutator, or MIL-STD-1553 card, where the data is expected to refresh relatively quickly. Status type data generally comes from devices that do not produce large quantities of data. Status data merely represents the state of various items on the device. For example, a status stream would come from a receiver, or a digital-to-analog converter. Serial type devices can also have status type data (like frame lock state, bit sync lock state, etc), but they are categorized as serial type devices because of the large quantity of data they produce. To clarify, some status type devices do produce large amounts of data, like the LS-22-SE analyzer/scope card described in paragraph 3.6 on page 143 of this manual. This card does have status data, but it also has a buffer of 1024 data words for the scope display. These types of devices are categorized as status type devices, but with the term Non Serial data.

In LDPS, data is collected at a 20-hertz rate for status type devices. For serial type devices, data is collected at the bit rate of the data.

For a variety of reasons, the number in input streams that LDPS can handle simultaneously is not infinite. Currently, the limit is set to twelve (12) serial type devices and twelve (12) status type devices. All of the devices currently supported by LDPS are documented in this manual. The remaining devices fall into the **Custom Aux Input Device** category (one for serial type and one for status type devices). Each device is controlled by a stand-alone application program that controls all the cards of that particular card type. For instance, a decommutator program controls and manages all the LS-50-P/LS-55-DB cards in the system. A separate program manages all the receivers in the system (like LS-25V2). The server program communicates with these stand-alone programs, collecting data from each in the required format.

Control of the standalone programs is handled by the programs, each with their own GUI. Communication with the server is implemented via an API-like structure that each program has to adhere to. This makes for easy expansion of new cards that may be developed and added to the server in the future.

Custom auxiliary programs are how data gets into the system when the device is not listed as one supported by LDPS. Only one Custom Aux Serial program and one Custom Aux Status program may be running at a time. However, these custom auxiliary programs can select which DLL to run. The DLL is actually what controls the data collection for the stand-alone custom auxiliary program. Each DLL has its own API. With this method, if data can come into the system via a DLL, it can become a part of the LDPS system. Each custom auxiliary program can support up to twelve (12) devices (streams).

2.2 The Device Manager

As mentioned previously, LDPS can support up to twelve (12) serial and twelve (12) status type streams. This could be a combination of decommutators, MIL-STD-1553 cards, Custom Aux serial or status devices, etc.

The LDPS **Device Manager** allows the user to select which of the data acquisition programs to run, and which not to run. The device manager is responsible for controlling the stand-alone programs, and for collecting data from them. Control is implemented mainly via sent messages containing the program state and GUI positions. Those stand-alone programs that are running are docked to the server program window. Data is passed back to the server as a result of events in the hardware. The device manager associates the data from each stand-alone program into a stream number and stream category (serial and status and non serial).

If LDPS is in P**layback** mode, the device manager will be collecting data from the playback manager, which is acting like the stand-alone programs did when in **Live** mode. During playback, the stand-alone programs are put into an idle state for collecting data.

At this point, regardless of mode (Live/Playback), all data is formatted as required, and passed off to the stream manager.

2.3 The Stream Manager

When the stream manager gets data, it is always in a fixed format. The kind of format depends on the type of data (Serial, Status, or Non-Serial). The stream manager doesn’t care what kind of device the data came from, or if it is live or playback. Only one buffer for each stream type is used. The pointer to the buffer is passed to the various “managers” for them to act upon as required.

Status type data (including non serial data) is always sent to the other managers, including the output to clients, regardless if a project is loaded or not. Serial data is not sent to the output manager unless a project is loaded. The exception to this is the remote TCP client manager. Also, the archive manager does not get any of the data unless a project is loaded and not in playback mode.

Serial data is always sent to the frame dump, embedded audio, embedded video, and remote TCP client managers. If a project is loaded, the data is fed to the output manager. If LDPS is in live mode, the data is also sent to the archive manager.

2.4 Embedded Audio & Video (A/V) Managers

The server program no longer extracts the audio/video and sends it on to the client(s) as separate streams. The client(s) now get 100% of the PCM stream and extract their own data via the A/V managers. The embedded audio and video managers are the same for both the server and client applications. These managers control the extraction of audio and video embedded in a stream and plays them back to the user regardless of mode (live or playback), with or without a project loaded. The user can also record the audio/video streams previously extracted from a telemetry stream. The video manager is limited to eight (8) streams. The audio manager is limited to twelve (12) streams for legacy reasons.

The audio manager uses the Windows multimedia player to play audio, and depending on the CODEC[7], the video manager either uses the Windows multimedia player or its own driver (H.261 uses its own driver, MPEG4 uses Windows). Depending on the operating system version, the user may have to install extra CODECs to get some audio/video types to play. These extra CODECs may be part of the LDPS install script, or if some non-standard format is being used (like MPEG4 in an MPEG2 transport), the user will have to find and install the CODECs on the system.

For playing back audio streams on a system where the server and client(s) are running on the same machine, it is recommended that the user only enable the server audio playback function (***View → Emb Audio/Video Manager***). If the audio playback function is enabled on BOTH the server and client, there will likely be an echo or a reverb effect, due to the slight processing delay between the server and client.

When a parameter database is initially created, the user has the ability to set the word locations, CODEC requirements, frame rate, etc. for the embedded audio and/or video streams. After a project is loaded, the user can override the setup information if desired. If a project is not running, the user may still enter the setup information via the A/V manager and play audio/video.

The A/V managers' GUI is fairly self-explanatory (see Figure 2-6 on page 48). The user specifies the data word locations, the CODEC to use, the sample size and rate, etc. One item not quite so obvious is the stream type. For most applications, PCM is the type used. For Chapter 8 type streams, the user can select either PCM if the words are at fixed locations in the stream, or Chapter 8, if the data is put into a USERDEF (1 or 2) word in the stream.

For MPEG video streams with audio embedded in them, the audio must be setup from within the video manager. Audio and a video on the same embedded stream cannot be played at the same time. In the scenario with an embedded audio stream, and an embedded video stream with embedded audio, the audio stream function will play the video streams embedded audio. The user will not be able to control the codec for the audio on the audio manager. It must instead be set up on the video manager where the

[7] CODEC – Coder Encoder.

embedded audio is. If the auto stream is to be played, rather than the video streams audio, then the audio type on the video stream must be set to NONE.

Selecting the video size is critical to getting the video rendered correctly. There are a couple of standards to select from, depending on the CODEC used. For the H.261 CODEC, only CIF (352 x 288) and QCIF (176 x 144) are supported. For the MPEG CODEC, 2CIF (704 x 576), CIF, QCIF, and USERDEF (anywhere between 10 and 2000 x 10 and 2000) are supported. The USERDEF was added for those vendors that don't observe standards, or who vary them from time to time.

ℹ	TIP: If the video size is unknown, record the video (as described in paragraph 2.4.1), then using a video player (like the Elecard video player), play the video and look at its setup information. This should give the video size.

2.4.1 Audio & Video Recording[8]

In LDPS, recording the audio and video data is an all or nothing proposition. If the user clicks the record button on the A/V manager, then all audio and video streams that are enabled will be recorded. However, if any one of the enabled streams cannot open the file for some reason (disk space, write protect, etc), then none of the streams enabled will be recorded.

Recording can be controlled on the server either by using the record button on the A/V manager or by using the record button on the normal archive window, depending on the option set in the A/V manager (Recorder Control Options). The user may select, **Main Archive Coupling** or **No Coupling**. With the "No Coupling" option, control is via the A/V manager record button. By selecting "Main Archive Coupling", control is via the normal archive record button. If the server is playback mode, then the coupling is set to "No Coupling." This enables one to record while in playback mode.

On the client application, there is no coupling mode. All recording functions are controlled by the A/V managers' record button.

The filenames used for audio and video recordings are fixed. The user can change where the archives are located (option on the A/V manager), but the file names themselves may not be changed. The naming convention and file types used are as follows:

[8] The term, "recording" applies to the storing of the actual sound or video on the system's hard drive.

a. For audio, the file name is AUD_STREAM_X.YYY, where:
   X is the stream number, and
   YYY is the file extension. The possible types include:
   1. CVSD, for Continuously Variable Slope Delta modulation type audio.
   2. PCM, for Pulse Code Modulation type audio.
   3. ADPCM, for Adaptive PCM audio.
   4. This list will grow.

b. For Video, the name is VID_STREAM_X.YYY where:
   X is the stream number, and
   YYY is the file extension, based on the video type that include:
   1. P64 for H.261 video.
   2. MPEG, for Moving Pictures Experts Group video.

The directory name where A/V recording takes place is fixed, with the location assigned by the A/V manager. The user may change the root directory name, but you cannot change the name of the directory under the root directory. The name of the recording directory uses a Julian date and timestamp naming convention. The name of the directory where the A/V data is recorded is: XXX_julian_timestamp, where

- XXX is either the project name if a project is loaded, or AV if a project is not loaded.
- Julian is the 4 digit Julian date.
- Timestamp is the time of day (hhmm) the directory was created (files opened for archiving).

The default A/V recording file locations are also different for the server and client. On the server, the default location is the same as the normal archiving location. The client uses the User\ClientFiles\AvArchives directory.

2.5 Archive Manager

The archive manager will not be active if LDPS is in playback mode. Also, it will not be active if a project is not loaded. The archive manager records all data coming from the data acquisition hardware in the system. As there are three types of stream buffers in LDPS, there are also three archive files for each stream. The status type data goes into one file, the non-serial type data goes into another file, and the serial type data (along with its status data) goes into a third file. This implementation scheme keeps the file size down, but it does create a bit of a challenge during playback. The playback manager collects all of the data from the disk, and then time-merges it with each of the three files as needed.

2.6 Aux Managers

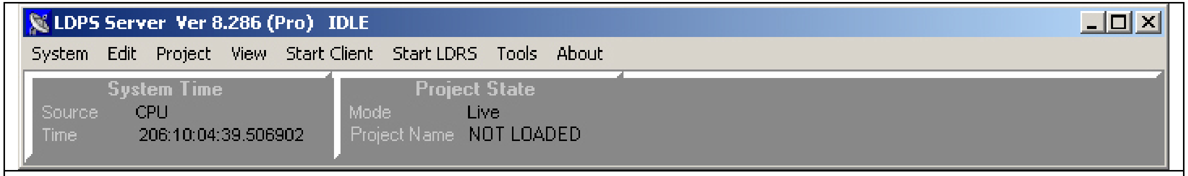
The auxiliary managers get their data from the stream manager. The embedded managers (audio & video) extract data from a stream and feed it to the rest of the system. The TCP manager transmits (via the TCP protocol) data to the clients that are connected to it.

2.7 Output Manager

The output manager accepts data from the stream manager and sends it out to the respective UDP port for network communications, or to a shared memory for stand-alone operation.

2.8 The Server Program

The server program functions as the collector and distributor of data coming from hardware devices in the system. The server sets up and controls this hardware, as well as the distribution mediums for the resulting data. It also handles the archiving of data from those devices. User control of the server program is through a GUI as shown in Figure 2-1 below.



System Time
Source CPU
Time 206:10:04:39.506902

Project State
Mode Live
Project Name NOT LOADED

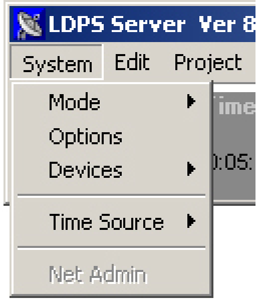
Figure 2-1 Main LDPS Server Window

The server program has a very simple GUI as shown above. The server GUI is divided into five main sections that include: a caption section, a menu section, an information section, a device control section, and a project section.

The caption section is at the top of the window. The caption displays the name of the program and the current version number. The caption also contains the standard window controls to minimize, maximize, and close the window and/or program. When closing the program, note that the following events will occur. First, any project that is loaded will be unloaded. Any open archive file will be closed. Running client programs will be notified that the server is shutting down, which will cause the clients to unload any loaded project. Any network connection with the server will be disconnected. Finally, all system configuration information will be saved to disk.

Below the caption section is the menu section. There are eight items in the main menu that include: the **System** menu, the **Edit** menu, the **Project** and **View** menus, the **Start Client** and **Start LDRS** commands, the **Tools** menu, and the **About** this window command. Each of these commands and menus are discussed in more detail in the following paragraphs.

2.8.1 The System Menu

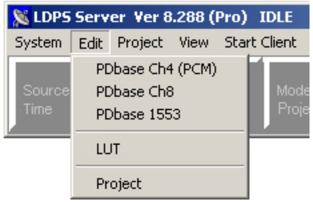


The system menu has five commands including: **Mode**, **Options**, **Devices**, **Time Source**, and **Net Administration** as shown right.

The Options command, described in detail in paragraph 2.11 on page 53, will launch a window (if a project is not loaded) to enable the user to configure the server. Usually, the server configuration is set once after LDPS is installed on the system, and seldom changed thereafter.

The Mode command, described in more detail in paragraph 2.8.9 on page 50, allows the user to place the server into **Live** or **Playback** mode. The Devices menu, described in more detail in paragraph 2.12 on page 63, allows the user to manage the hardware devices installed in the system. The Time Source menu allows the user to select the hardware device that will provide system time. Finally, the Net Admin command, described in more detail in paragraph 1.6 on page 35, allows the user to configure the network used in the system.

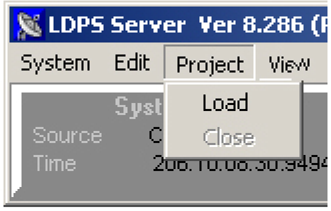
2.8.2 The Edit Menu



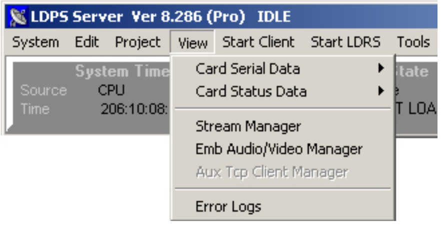
The edit menu has three commands including: **PDbase** (parameter database), **LUT** (look-up table), and **Project** as shown right.

The edit PDbase command, described in more detail in paragraph 4.3 on page 190, invokes the parameter database editor. The edit LUT command, described in more detail in paragraph 4.1 on page 185, invokes the lookup table editor. The edit project command, described in more detail in paragraph 4.2 on page 187, invokes the project editor.

2.8.3 The Project Menu



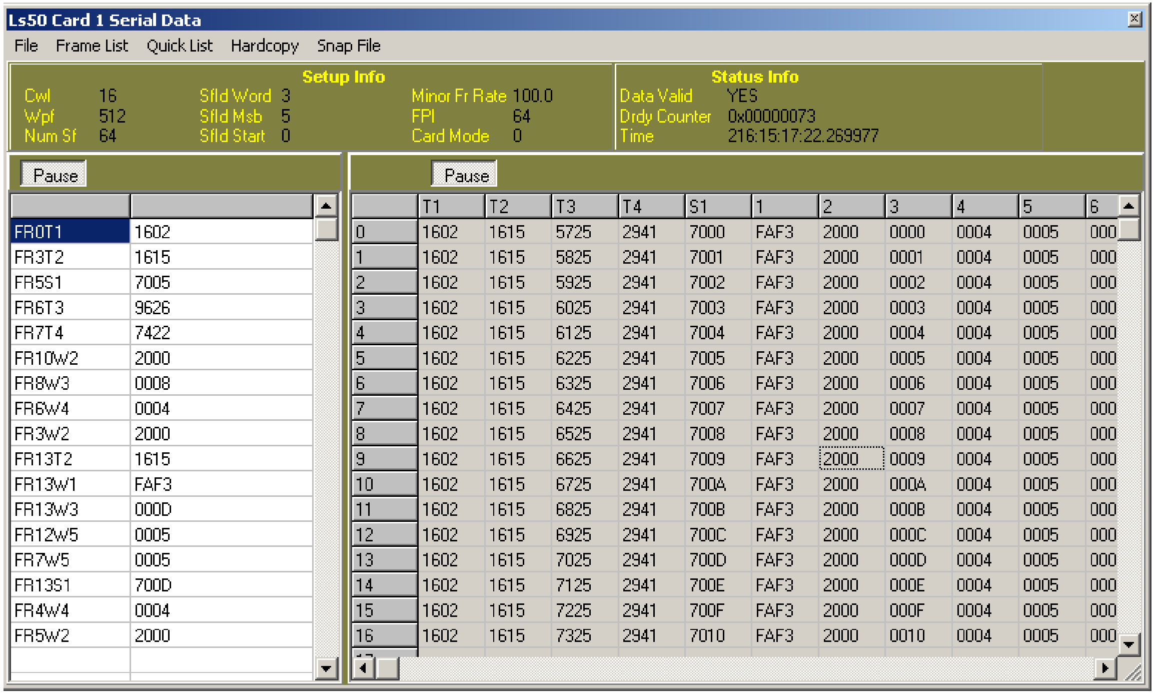
The Project menu is where the state of the project is control. Here, one may **Load** a project, or **Close** the project. These sub-menus will be grayed out appropriately, depending on the state of the project.



2.8.4 The View Menu

The View menu, shown left, has six commands including: **Card Serial Data**, **Card Status Data**, **Stream Manager**, **Emb Audio/Video Manager**, **Aux TCP Client Manager**, and **Error Logs**.

Invoking the **Card Serial Data** command will produce a list of serial device types installed in the system (LS-50, 1553, etc.). For each device type, the user may then select a specific card number. The resulting window, an example of which is shown in Figure 2-2 below, provides a real-time updating view of the raw data coming from the selected serial device. There will be one such display for each stream. This command functions in both live and playback modes, and allows hardcopy and ASCII snap shots of the frame data. Individual words may be selected for display in the Quick List (shown below left) by double clicking the desired word from the Frame List display (shown below right). The selective word display, and the Frame List display, may individually be paused by clicking the **Pause** button. Click Pause again to resume the real-time update of the display. The **Frame List**, and **Quick List** commands allow the user to specify the number type for the display. The available number types include: decimal, HEX, Binary, and Octal.



Setup Info

Cwl	16	Sfld Word	3	Minor Fr Rate	100.0
Wpf	512	Sfld Msb	5	FPI	64
Num Sf	64	Sfld Start	0	Card Mode	0

Status Info

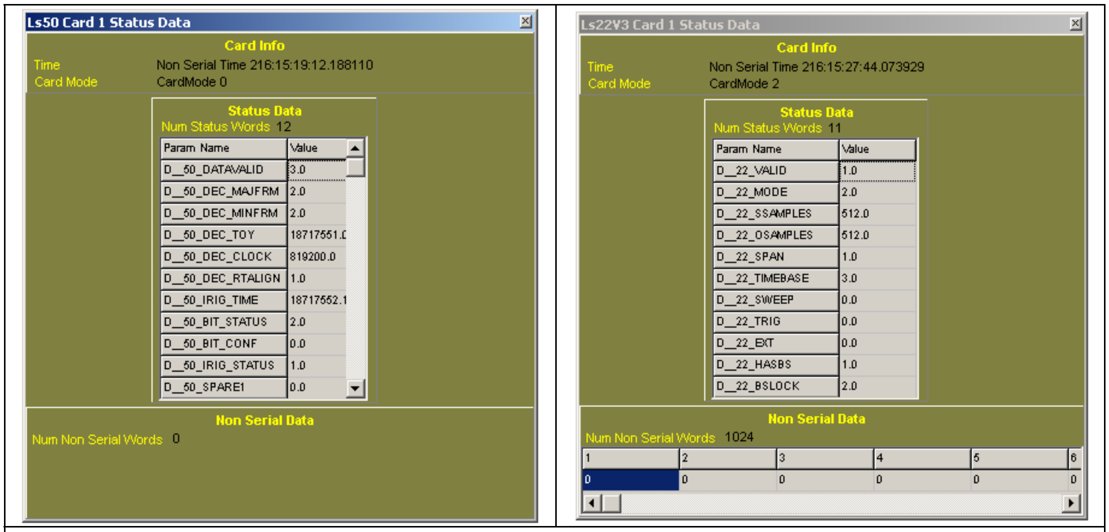
Data Valid	YES
Drdy Counter	0x00000073
Time	216:15:17:22.269977

FR0T1	1602
FR3T2	1615
FR5S1	7005
FR6T3	9626
FR7T4	7422
FR10W2	2000
FR8W3	0008
FR6W4	0004
FR3W2	2000
FR13T2	1615
FR13W1	FAF3
FR13W3	000D
FR12W5	0005
FR7W5	0005
FR13S1	700D
FR4W4	0004
FR5W2	2000

	T1	T2	T3	T4	S1	1	2	3	4	5	6
0	1602	1615	5725	2941	7000	FAF3	2000	0000	0004	0005	000
1	1602	1615	5825	2941	7001	FAF3	2000	0001	0004	0005	000
2	1602	1615	5925	2941	7002	FAF3	2000	0002	0004	0005	000
3	1602	1615	6025	2941	7003	FAF3	2000	0003	0004	0005	000
4	1602	1615	6125	2941	7004	FAF3	2000	0004	0004	0005	000
5	1602	1615	6225	2941	7005	FAF3	2000	0005	0004	0005	000
6	1602	1615	6325	2941	7006	FAF3	2000	0006	0004	0005	000
7	1602	1615	6425	2941	7007	FAF3	2000	0007	0004	0005	000
8	1602	1615	6525	2941	7008	FAF3	2000	0008	0004	0005	000
9	1602	1615	6625	2941	7009	FAF3	2000	0009	0004	0005	000
10	1602	1615	6725	2941	700A	FAF3	2000	000A	0004	0005	000
11	1602	1615	6825	2941	700B	FAF3	2000	000B	0004	0005	000
12	1602	1615	6925	2941	700C	FAF3	2000	000C	0004	0005	000
13	1602	1615	7025	2941	700D	FAF3	2000	000D	0004	0005	000
14	1602	1615	7125	2941	700E	FAF3	2000	000E	0004	0005	000
15	1602	1615	7225	2941	700F	FAF3	2000	000F	0004	0005	000
16	1602	1615	7325	2941	7010	FAF3	2000	0010	0004	0005	000

Figure 2-2 Real-Time Serial Device Data Display for the LS-50

Invoking the **Card Status Data** command will produce a list of status device types installed in the system (basically all cards supplied by Lumistar). For each device type, the user may then select a specific card number. The resulting window, an example of which is shown in Figure 2-3 below, provides a real-time updating view of the status data coming from the selected device. There will be one such display for each card in the system. The status data show in this display is the same data available to the clients for display as well.



Card Info

Time: Non Serial Time 216:15:19:12.188110
Card Mode: CardMode 0

Status Data

Num Status Words 12

Param Name	Value
D__50_DATAVALID	3.0
D__50_DEC_MAJFRM	2.0
D__50_DEC_MINFRM	2.0
D__50_DEC_TOY	18717551.[illegible]
D__50_DEC_CLOCK	819200.0
D__50_DEC_RTALIGN	1.0
D__50_IRIG_TIME	18717552.1
D__50_BIT_STATUS	2.0
D__50_BIT_CONF	0.0
D__50_IRIG_STATUS	1.0
D__50_SPARE1	0.0

Non Serial Data

Num Non Serial Words 0

Card Info

Time: Non Serial Time 216:15:27:44.073929
Card Mode: CardMode 2

Status Data

Num Status Words 11

Param Name	Value
D__22_VALID	1.0
D__22_MODE	2.0
D__22_SSAMPLES	512.0
D__22_OSAMPLES	512.0
D__22_SPAN	1.0
D__22_TIMEBASE	3.0
D__22_SWEEP	0.0
D__22_TRIG	0.0
D__22_EXT	0.0
D__22_HASBS	1.0
D__22_BSLOCK	2.0

Non Serial Data

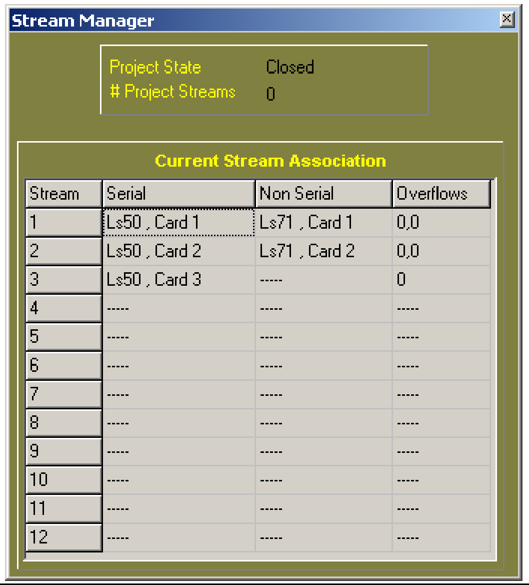
Num Non Serial Words 1024

1	2	3	4	5	6
0	0	0	0	0	0

Figure 2-3 Real-Time Device Status Display for the LS-50 & LS-22

Invoking the **Stream Manager** command will produce a list of all devices installed in the system, and what streams are associated with them. The resulting window, an example of which is shown right in Figure 2-4, provides a list of the twelve streams, and what serial and no-serial devices are assigned to them.

Invoking the **Error Logs** command allows the user to view all of the error logs generated by the various programs and device DLLs. The resulting display, an example of which is shown in Figure 2-5 on page 48, is not exclusively error logs, but contain status and other information as well.



Project State: Closed
Project Streams: 0

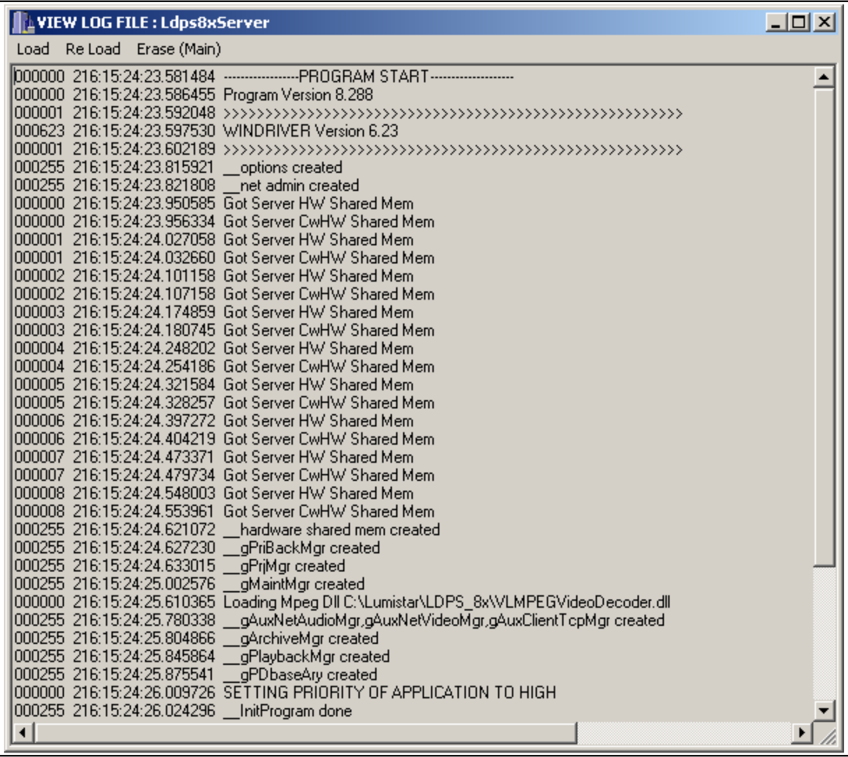
Current Stream Association

Stream	Serial	Non Serial	Overflows
1	Ls50 , Card 1	Ls71 , Card 1	0,0
2	Ls50 , Card 2	Ls71 , Card 2	0,0
3	Ls50 , Card 3	-----	0
4	-----	-----	-----
5	-----	-----	-----
6	-----	-----	-----
7	-----	-----	-----
8	-----	-----	-----
9	-----	-----	-----
10	-----	-----	-----
11	-----	-----	-----
12	-----	-----	-----

Figure 2-4 Stream Manager Display

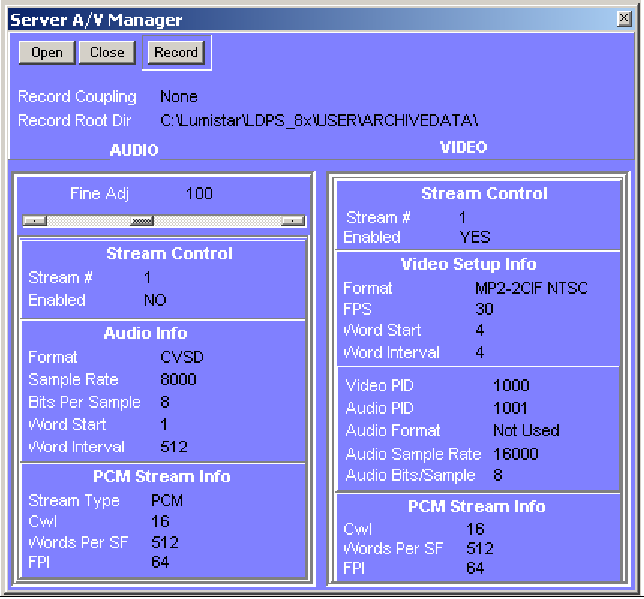
By invoking the **Embedded Audio/Video Manager**, shown in Figure 2-6 on page 48, the user can configure and control the extraction of embedded audio and/or video within a stream. Note: changes that override the project settings may be made using this command.

The **Aux TCP Client Manager**, shown in Figure 2-7 on page 48, allows the user to see who is connected to the auxiliary TCP output (clients it is talking to). This command is only enabled if networking is enabled (see paragraph 2.11.2 on page 55 for more information about network configuration).



```
000000 216:15:24:23.581484 ----------------PROGRAM START------------------
000000 216:15:24:23.586455 Program Version 8.288
000001 216:15:24:23.592048 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
000623 216:15:24:23.597530 WINDRIVER Version 6.23
000001 216:15:24:23.602189 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
000255 216:15:24:23.815921 __options created
000255 216:15:24:23.821808 __net admin created
000000 216:15:24:23.950585 Got Server HW Shared Mem
000000 216:15:24:23.956334 Got Server CwHW Shared Mem
000001 216:15:24:24.027058 Got Server HW Shared Mem
000001 216:15:24:24.032660 Got Server CwHW Shared Mem
000002 216:15:24:24.101158 Got Server HW Shared Mem
000002 216:15:24:24.107158 Got Server CwHW Shared Mem
000003 216:15:24:24.174859 Got Server HW Shared Mem
000003 216:15:24:24.180745 Got Server CwHW Shared Mem
000004 216:15:24:24.248202 Got Server HW Shared Mem
000004 216:15:24:24.254186 Got Server CwHW Shared Mem
000005 216:15:24:24.321584 Got Server HW Shared Mem
000005 216:15:24:24.328257 Got Server CwHW Shared Mem
000006 216:15:24:24.397272 Got Server HW Shared Mem
000006 216:15:24:24.404219 Got Server CwHW Shared Mem
000007 216:15:24:24.473371 Got Server HW Shared Mem
000007 216:15:24:24.479734 Got Server CwHW Shared Mem
000008 216:15:24:24.548003 Got Server HW Shared Mem
000008 216:15:24:24.553961 Got Server CwHW Shared Mem
000255 216:15:24:24.621072 __hardware shared mem created
000255 216:15:24:24.627230 __gPriBackMgr created
000255 216:15:24:24.633015 __gPrjMgr created
000255 216:15:24:25.002576 __gMaintMgr created
000000 216:15:24:25.610365 Loading Mpeg Dll C:\Lumistar\LDPS_8x\VLMPEGVideoDecoder.dll
000255 216:15:24:25.780338 __gAuxNetAudioMgr,gAuxNetVideoMgr,gAuxClientTcpMgr created
000255 216:15:24:25.804866 __gArchiveMgr created
000255 216:15:24:25.845864 __gPlaybackMgr created
000255 216:15:24:25.875541 __gPDbaseAry created
000000 216:15:24:26.009726 SETTING PRIORITY OF APPLICATION TO HIGH
000255 216:15:24:26.024296 __InitProgram done
```

Figure 2-5 Sample Error Log (*View→ Error Logs*)



Record Coupling None
Record Root Dir C:\Lumistar\LDPS_8x\USER\ARCHIVEDATA\

AUDIO

Fine Adj 100

Stream Control
Stream # 1
Enabled NO

Audio Info
Format CVSD
Sample Rate 8000
Bits Per Sample 8
Word Start 1
Word Interval 512

PCM Stream Info
Stream Type PCM
Cwl 16
Words Per SF 512
FPI 64

VIDEO

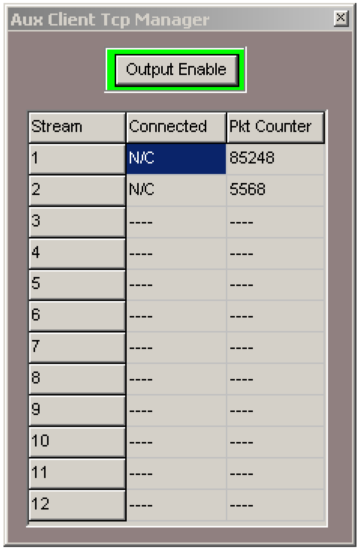
Stream Control
Stream # 1
Enabled YES

Video Setup Info
Format MP2-2CIF NTSC
FPS 30
Word Start 4
Word Interval 4

Video PID 1000
Audio PID 1001
Audio Format Not Used
Audio Sample Rate 16000
Audio Bits/Sample 8

PCM Stream Info
Cwl 16
Words Per SF 512
FPI 64

Figure 2-6 Embedded Audio & Video Manager



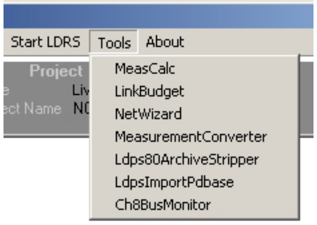
Stream	Connected	Pkt Counter
1	N/C	85248
2	N/C	5568
3	----	----
4	----	----
5	----	----
6	----	----
7	----	----
8	----	----
9	----	----
10	----	----
11	----	----
12	----	----

Figure 2-7 Auxiliary Client TCP Manager Window

2.8.5 Start Client Command

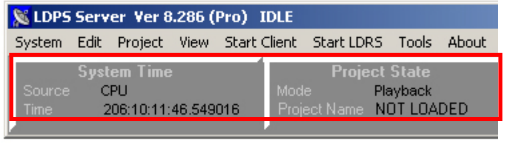
The user may launch the client application either from the desktop or by selecting this command from the main server menu. When the **Start Client** command is invoked from the server, and then subsequently the server application is then closed, the client application will automatically close also.

2.8.6 The Tools Menu



The **Tools** menu, shown right, offers a variety of handy utilities that are easily accessed via this menu. These tools include: a Link Budget calculator, Measurement Calculators, Measurement Converter, the Network Wizard, LDPS Archive Stripper, LDPS Parameter Database Import Utility, and an IRIG-106 Chapter 8 Bus Monitor. Note for Link Budget and Measurement Calculators, the user may add other tools simply by placing an exe file into the User Tools sub directory. For more information on the individual tools, please refer to the Appendix of this document beginning on page 253.

2.8.7 Server Window – Information Section

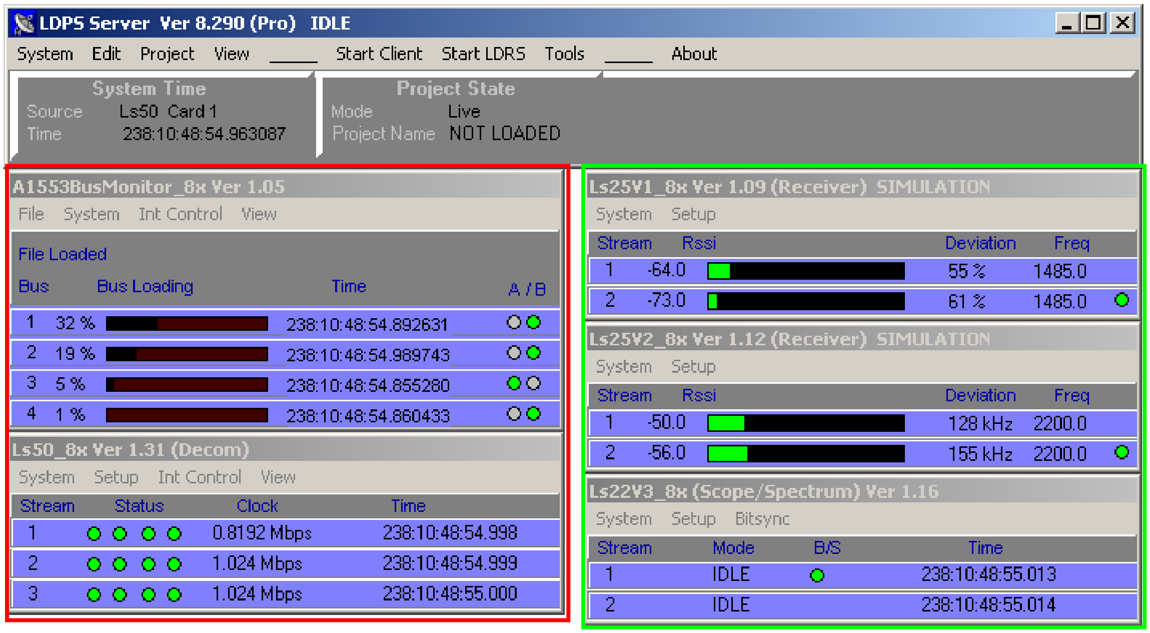


Source CPU
Time 206:10:11:46.549016
Mode Playback
Project Name NOT LOADED

The information section of the server window (shown right – red rectangle) is below the main menu and displays information such as the Project State (live or playback), the name of the project loaded (if any) and the system time. If the system time source is set to the CPU (which it is if no project is loaded), there will be an exclamation mark and the text CPU next to the time.

2.8.8 Server Window – Device Control Section

The Device Control section is located below the information section, and actually consists of individual device control windows that are linked to the main server window. The individual device control windows may not be moved independently but they do move in unison with the main server window. The device control windows are described in great detail in the numbered paragraphs of section 3 beginning on page 76. An example of several device control windows is shown in Figure 2-8 below. The device control section is where the operation of the various devices installed on the machine is controlled. Each device category is actually a stand-alone program linked to the server application. Each of these programs can have up to twelve (12) cards associated with them, but only a total of twenty-four cards for all the programs can be used. There are two fundamental types of cards, serial (red rectangle below) and non-serial (green rectangle below). The serial type cards (like an LS-50-P decommutator) are located on the left side below the server window. The non-serial type cards (like an LS-25 receiver) are located on the right side below the server window. Therefore, a total of twenty-four (24) cards can be in the system (assuming a chassis that can accept that many cards and with enough power).


LDPS Server Ver 8.290 (Pro) IDLE
System Edit Project View Start Client Start LDRS Tools About
A1553BusMonitor_8x Ver 1.05
File System Int Control View
Ls50_8x Ver 1.31 (Decom)
System Setup Int Control View
Ls25V1_8x Ver 1.09 (Receiver) SIMULATION
System Setup
Ls25V2_8x Ver 1.12 (Receiver) SIMULATION
System Setup
Ls22V3_8x (Scope/Spectrum) Ver 1.16
System Setup Bitsync

System Time
Source Ls50 Card 1
Time 238:10:48:54.963087

Project State
Mode Live
Project Name NOT LOADED

File Loaded

Bus	Bus Loading	Time	A / B
1	32 %	238:10:48:54.892631	○ ●
2	19 %	238:10:48:54.989743	○ ●
3	5 %	238:10:48:54.855280	● ○
4	1 %	238:10:48:54.860433	○ ●

Stream	Status	Clock	Time
1	● ● ● ●	0.8192 Mbps	238:10:48:54.998
2	● ● ● ●	1.024 Mbps	238:10:48:54.999
3	● ● ● ●	1.024 Mbps	238:10:48:55.000

Stream	Rssi	Deviation	Freq	
1	-64.0	55 %	1485.0	
2	-73.0	61 %	1485.0	●

Stream	Rssi	Deviation	Freq	
1	-50.0	128 kHz	2200.0	
2	-56.0	155 kHz	2200.0	●

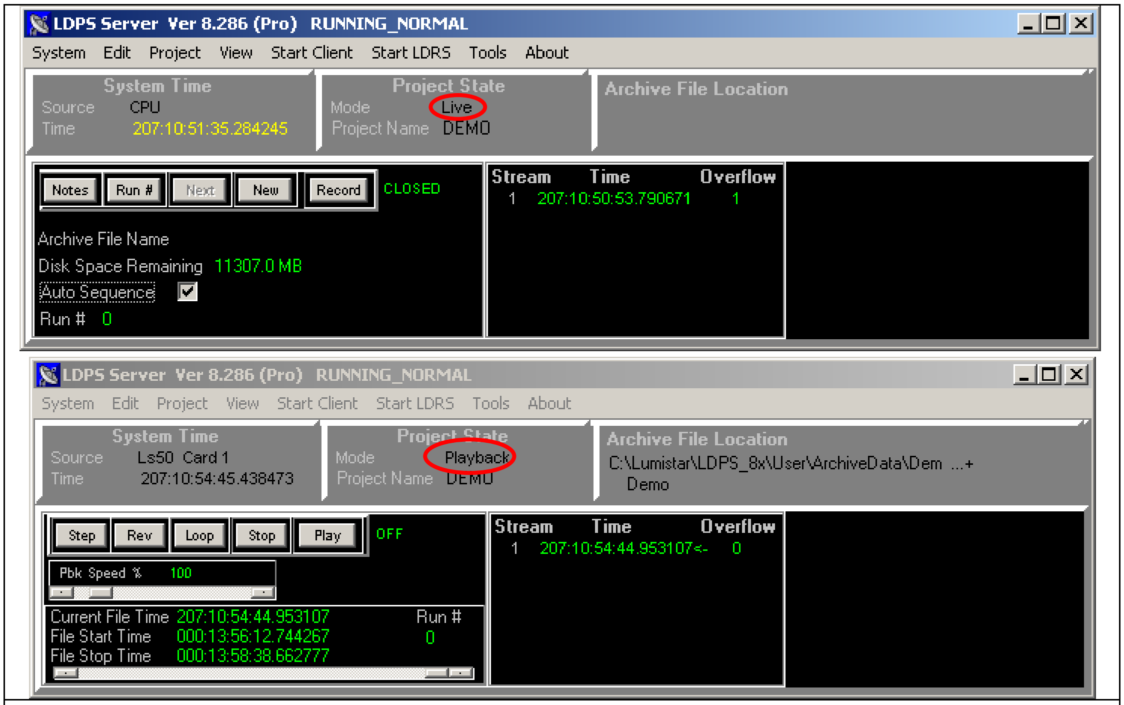
Stream	Mode	B/S	Time
1	IDLE	●	238:10:48:55.013
2	IDLE		238:10:48:55.014

Figure 2-8 Server Device Control Windows

2.8.9 Sever Window – Project State Section

The Project State section of the server window is only visible if a project is loaded (either in live or playback mode), and when visible is located directly below the Information section described in paragraph 2.8.7 on page 49. The two examples of the project state section are shown in Figure 2-9 on page 51. On the left side is either the archive control (if in live mode – upper part of figure below) or the playback control (if in playback mode – lower part of figure below).

To the right of the archive/playback controls is an information section showing the stream time. If the system time is set to one of the stream times, then an arrow will point to the stream time being used for system time. Next to the time display is an overflow counter. This counter will give an indication of how much data is being lost, if any, from the device. If this number is incrementing, then data is being lost during archive. This indicates that the interrupt rate may be too fast for the machine with the current setup of the decommutator. To address this scenario, unload the project and increase the frames-per-interrupt setting on the decommutator configuration. In general, the system should be tested BEFORE an actual test is conducted. Do this by setting the decommutator input source to the simulator and load and run the project to see if the overflow counter is incrementing. It is expected that the overflow counter will increment to 1, or perhaps even 2 when a project is first loaded and run. After that, the counter should not increment. In playback mode, the overflow counter should not increment no matter what the decommutator setup was.


LDPS Server Ver 8.286 (Pro) RUNNING_NORMAL
System Edit Project View Start Client Start LDRS Tools About
Notes Run # Next New Record
Step Rev Loop Stop Play

System Time
Source CPU
Time 207:10:51:35.284245

Project State
Mode Live
Project Name DEMO

Archive File Location

CLOSED

Archive File Name
Disk Space Remaining 11307.0 MB
Auto Sequence ☑
Run # 0

Stream Time Overflow
1 207:10:50:53.790671 1

System Time
Source Ls50 Card 1
Time 207:10:54:45.438473

Project State
Mode Playback
Project Name DEMO

Archive File Location
C:\Lumistar\LDPS_8x\User\ArchiveData\Dem ...+
Demo

OFF

Pbk Speed % 100

Current File Time 207:10:54:44.953107 Run #
File Start Time 000:13:56:12.744267 0
File Stop Time 000:13:58:38.662777

Stream Time Overflow
1 207:10:54:44.953107<- 0

Figure 2-9 LDPS Server Window Variants - Live & Playback Modes

2.9 The Backup Server

The LDPS server application can operate as a backup server as well as the primary server. The option to act as a backup server is set up by the user in the options menu. They are physically the same program, just running on different machines. The backup server only functions on network enabled systems. If no network is installed, then skip this section.

The purpose of the backup server is to allow a test to continue even if the primary server application stops running for some reason (blown power supply, computer crashes, etc.). If the primary server goes down (abnormal termination), the backup server will automatically take over. The clients are informed which server program is acting as the primary server and they will request tags and accept data accordingly. The switch from the primary to backup server is automatic, with no user intervention required. If the primary server can detect that it is shutting down, the transition to the backup server will take approximately 10 milliseconds. If the primary cannot detect the shutdown, the transition takes approximately 600 milliseconds. The user can manually force the backup server to take over by pressing the “CTRL-C” key on the primary server. This will abnormally terminate the primary server program and signal the backup server to take over.

If the backup server becomes the primary server, and the original primary server is repaired and brought back on line, it will then act as the backup server, in case the original backup server (now acting as the primary server) should fail.

Closing the project or normally terminating the server application will not trigger the backup server to take over as primary. Only failures or a system timeouts will cause the transition. In order for this automatic transition to work correctly, both servers must have access to the same project information. This can be achieved by having both servers pointing to the needed files on a separate machine, or ensuring that the project files are copied to the backup server prior to a test. Pointing to the required files is performed in the options menu/Directory Tab, on both servers. The Log On user password must also be the same on both servers, and each server must have the same hardware installed (decommutators, receivers, etc)

For normal operation, start the primary server first, followed by the backup server. From then on, load the project file only on the primary server. The backup server will automatically do the same thing as the primary (loads the same project). As clients request and release tags, the same requests goes to the backup server. The backup server mirrors everything the primary server does, except sending out the data. When a take over occurs, the actual sending of data goes out to the appropriate UDP ports and archiving (if it was in activated) begins again. The clients are then alerted to switch operational servers. If the backup server has become the operational server, then the client will display a message in its caption area alerting the user that the backup server has taken over.

2.10 Device Server Communication

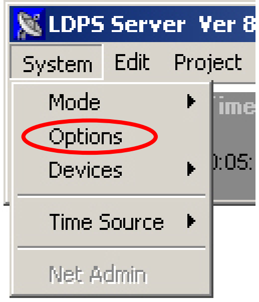
As mentioned in the introduction of this manual, the server program collects data from the hardware devices installed in the system. The hardware devices must in turn be attached to the server and configured to the machine (via the Device Manager on the server – see paragraph 2.12 on page 63). Hardware devices are things such as decommutators, telemetry receivers, etc. Currently, devices that are added to the list of available devices are hard-coded into LDPS. Lumistar must add any new hardware device that becomes available. In the future, however, a generic interface for new devices is planned and will allow such devices to be added by the user.

Each device has a corresponding stand-alone application associated with it. The GUI for the stand-alone version and the server version are identical, with the exception that the stand-alone version has the standard Windows terminate program control (X in upper right corner of the GUI), and the “About” version number command. On the server version, the user can hide the setup GUI, until needed, by selecting the **Hide** command from the devices menu *(System → Device → Hide).* The GUI must be running when a project is loaded, so the customary Windows controls on the top right of the GUI have been removed. The device GUI is still running while hidden, but the user has no control over it. When the device GUI is visible, it operates the same as the stand-alone version, except some controls may be disabled if a project is loaded (changing controls could affect the project). An example of this would be the words-per-minor frame control for a decommutator. If the project is loaded, this control will be disabled.

2.10.1 Device Event Logs

Each hardware device has an event log generated by the DLL controlling the device. This log is kept in the System\ErLogs directory. Each file contains events and errors associated with the program or communication with the device(s). The log files provide valuable information to help troubleshoot problems and are needed when calling Lumistar customer service for support.

2.11 Server Options

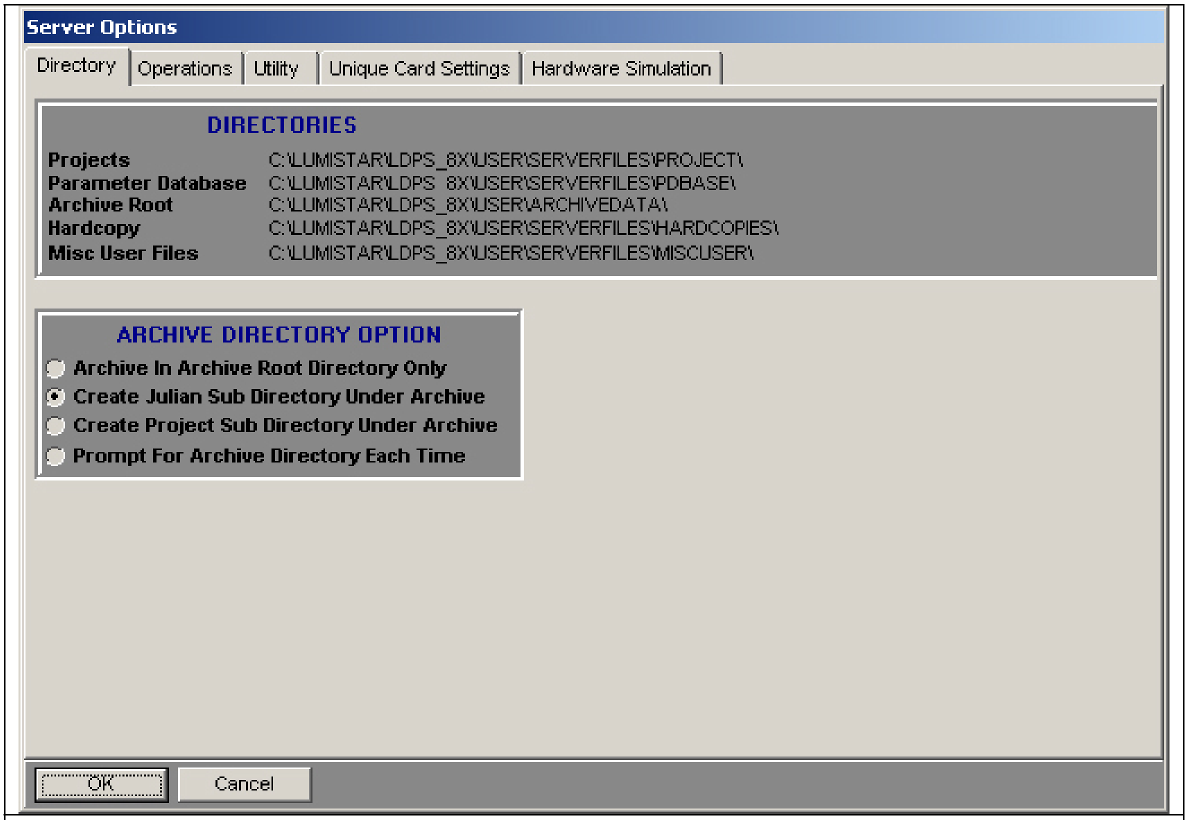


The server program has several options for customizing the system. The options include: categorization of directories, program operation, and startup. Selecting System → Options on the main menu as shown right will invoke the Server Options window shown in Figure 2-10 on page 54. The server options window has five (5) individual tabs to configure. The tabs include: Directory, Operations, Utility, Unique Card Settings, and Hardware Simulation. Each is described in detail in the numbered paragraphs that follow.

Some of the options described here are also used by the client (like color selections, and directories). The server options file is shared by the individual device applications.

2.11.1 Directory Tab

The directory tab shown in Figure 2-10 below allows the use to specify the location of certain LDPS files in directories chosen by the user. For the server, there are five directory locations that may be specified. These directories include project, parameter database, archive root, hardcopy, and miscellaneous user files. The directory tab also allows the user to set specific options for the creation of directory names associated with data archiving.



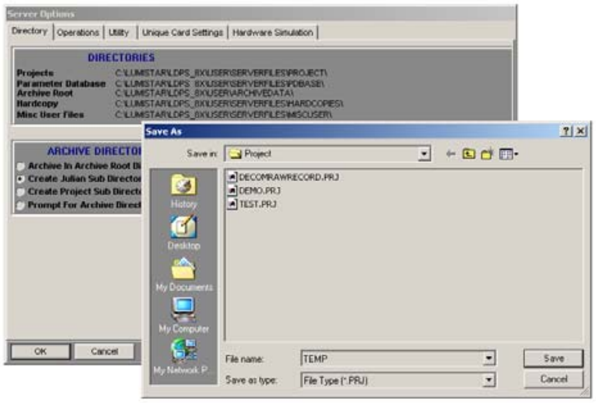
DIRECTORIES

Projects	C:\LUMISTAR\LDPS_8X\USER\SERVERFILES\PROJECT\
Parameter Database	C:\LUMISTAR\LDPS_8X\USER\SERVERFILES\PDBASE\
Archive Root	C:\LUMISTAR\LDPS_8X\USER\ARCHIVEDATA\
Hardcopy	C:\LUMISTAR\LDPS_8X\USER\SERVERFILES\HARDCOPIES\
Misc User Files	C:\LUMISTAR\LDPS_8X\USER\SERVERFILES\MISCUSER\

ARCHIVE DIRECTORY OPTION

○ Archive In Archive Root Directory Only
● Create Julian Sub Directory Under Archive
○ Create Project Sub Directory Under Archive
○ Prompt For Archive Directory Each Time

Figure 2-10 Server Options - Directory Tab



Save in: Project

DECOMRAWRECORD.PRJ
DEMO.PRJ
TEST.PRJ

File name: TEMP
Save as type: File Type (*.PRJ)

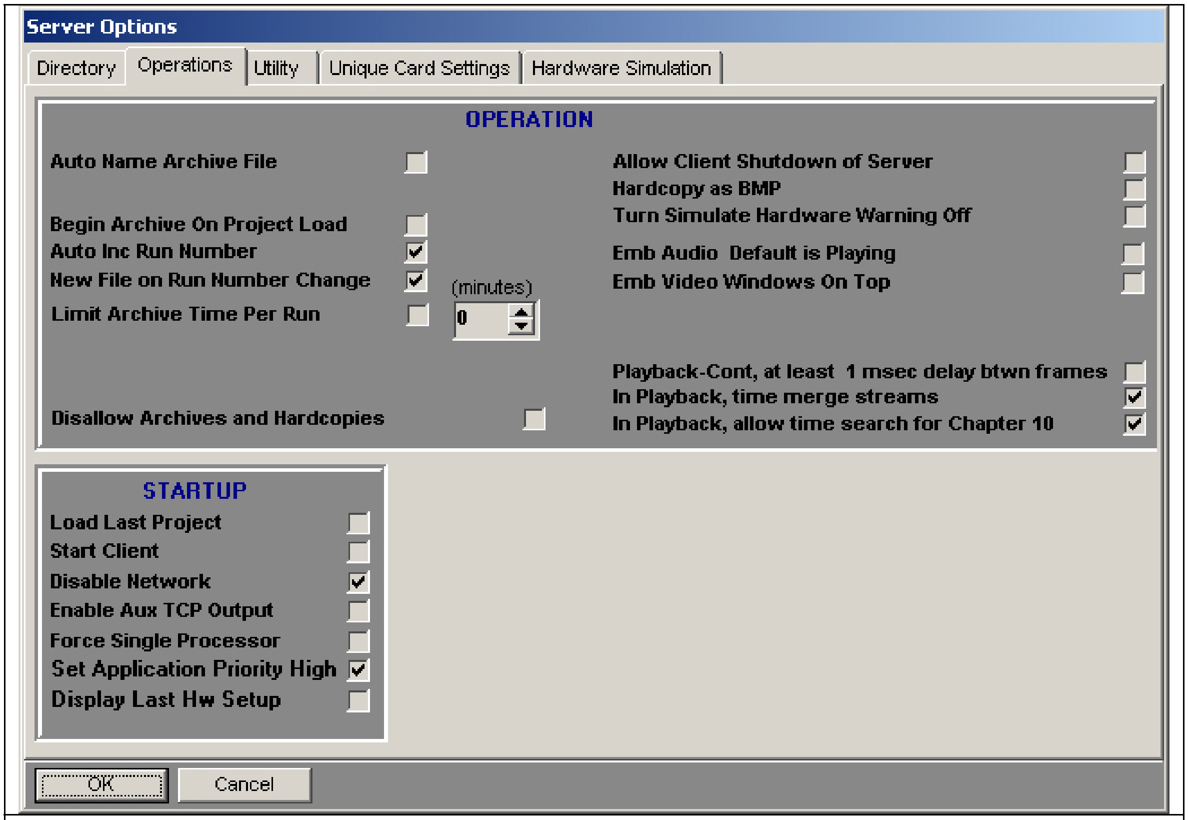
As mentioned, the Directories area includes: project, parameter database, archive root, hardcopy, and miscellaneous user files. To define the directory path, place the cursor on name of the directory, and single click. This will invoke the standard Windows file dialog box shown right. Navigate to the required directory via the Network Neighborhood and click O.K. For example, place the cursor on the word "**Projects**" and click once. This allows the user to store project files anywhere on the system. The remote clients must also have access to this location from the network. For more information on file and directory structure in LDPS, see paragraph 1.5 on page 30.

There are four archive directory options that the user may select. The user selects only one of these options via the radio buttons. The different directory options control how the individual directories that contain archive data are named and located. The **Archive Root Directory Only** option is not recommended for most users, but when selected, the archive data files are stored in the archive root directory. The **Create Julian Sub**

Directory Under Archive Root option will create a subdirectory under the archive root directory when an archive file is opened for recording. The subdirectory name will be the Julian date, followed by the time stamp, followed by the project name. The **Create Project Sub Directory Under Archive Root** option is the same as the previous option, but without the date and timestamp. The **Prompt for Archive Directory Each Time** option prompts the user to enter the subdirectory name when an archive file is opened for recording.

2.11.2 Operations Tab

The operations tab shown in Figure 2-11 below allows the user to setup and configure the individual peculiarities of how the LDPS application works. This includes those things that occur automatically upon the **Startup** of the LDPS application, as well as a variety of **Operation**al features.



OPERATION

- Auto Name Archive File □
- Begin Archive On Project Load □
- Auto Inc Run Number ☑
- New File on Run Number Change ☑
- Limit Archive Time Per Run □ (minutes) 0
- Disallow Archives and Hardcopies □
- Allow Client Shutdown of Server □
- Hardcopy as BMP □
- Turn Simulate Hardware Warning Off □
- Emb Audio Default is Playing □
- Emb Video Windows On Top □
- Playback-Cont, at least 1 msec delay btwn frames □
- In Playback, time merge streams ☑
- In Playback, allow time search for Chapter 10 ☑

STARTUP

- Load Last Project □
- Start Client □
- Disable Network ☑
- Enable Aux TCP Output □
- Force Single Processor □
- Set Application Priority High ☑
- Display Last Hw Setup □

Figure 2-11 Server Options - Operations Tab

The operational configuration for LDPS include the following features:

- **Auto Name Archive File** – By checking this box, the name of the archive data file is prefixed with any text that is entered by the user (see below).
- **Archive Filename Prefix** – If the Auto Name Archive File box is checked, then double click on this label and enter the prefix text to include on the archive file name.

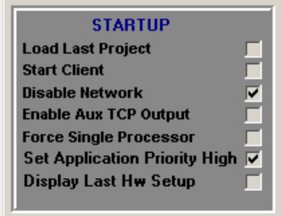
- **Begin Archive On Project Load** – If this option is chosen, archiving will automatically begin as soon as the project is loaded.
- **Auto Inc Run Number** – If this box is checked, when archiving is switched from On to Off, the run number will automatically increment by 1. The user may still manually increment the run number by pressing the “Run #” button.
- **New File On Run Number Change** – If this option is chosen, a new archive file will be created whenever the run number changes. Otherwise, all of the data from a given stream will be archived to a single file. The run number information is contained within the archive file.
- **Limit Archive Time Per Run** – This option is only available if Auto Inc Run Number and New File On Run Number Change are checked. If this box is checked, then select the number of minutes to archive data before the file is closed and a new file is created automatically for archive. Use this option to manage the file size of the individual archive files.
- **Hardcopy as BMP** – Check this box and all hardcopies made on the server will be saved as Windows BMP files in the hardcopy directory selected in the directory options. Otherwise, the hardcopies will be saved as JPG files.
- **Allow Client Shutdown** – The client program running on the same machine as the server has the ability to shut down the server if this box is checked. This is a dangerous thing, so the option must be set in both the server and client if this feature is to occur. This will not shut down the backup server unless the client is connected to the backup server.
- **Disallow Archives and Hardcopies** – If this option is checked, the user will not be allowed to archive data or make hardcopies.
- **In Playback, time merge streams** –If this option is selected, then during playback of multiple streams, the times for all streams will line up. This option does slow the operation of LDPS down somewhat. The time merge is based on the fastest of all the streams in the project. It has no affect on playback operations if only one stream was archived. For best results, archive the data with the decommutator setting at one-frame-per-interrupt. Otherwise, the time merging is based on the buffer of frames archived, not the minor frame. If time merging is active, the fastest data can be played back is the 1,000 Hertz minor frame rate. In the scenario of a fast stream with embedded audio (like a Chapter 8 stream), the audio will not sound right. Note: Playback in Reverse while this mode is active will result in garbage.
- **In Playback, allow time search for Chapter 10** – The IRIG-106 Chapter 10 archive format does not lend itself to playback very well. In order to do time searches, some preprocessing of the file needs to be performed. Depending on the length of time recorded, bit rate, etc, this process could take a couple of minutes before program control is returned. Once the preprocessing is performed, time searches will work as normal (as they do in the native archive format). It is recommended that the Chapter 10 file format be converted to the native format using the conversion tool so that full capability of searching can be maintained.
- **Playback-Cont, at least 1 msec delay btwn frames** – This option applies a 1 msec delay between minor frames. This ensures that there is at least 1 msec per

minor frame to allow the client to process all the data before the next buffer of data is sent to it.

- **Emb Audio Default is Playing** – If this option is selected, then the embedded audio will automatically begin playing. The default is not to automatically select Play.
- **Emb Video Windows On Top** – If this option is selected, then all embedded video displays will always be on top of other displays.
- **Turn Simulate Hardware Warning Off** – If this option is selected, the sometime annoying sound and message indicating the hardware is being simulated will be turned off. This option is not checked by default. The flashing SIMULATING above the hardware controls is not affected by this option.

2.11.2.1 Startup

Below the operations area shown in Figure 2-11 on page 55, are the startup controls for the LDPS application. The startup configuration for LDPS include the following features:



STARTUP

Load Last Project □
Start Client □
Disable Network ☑
Enable Aux TCP Output □
Force Single Processor □
Set Application Priority High ☑
Display Last Hw Setup □

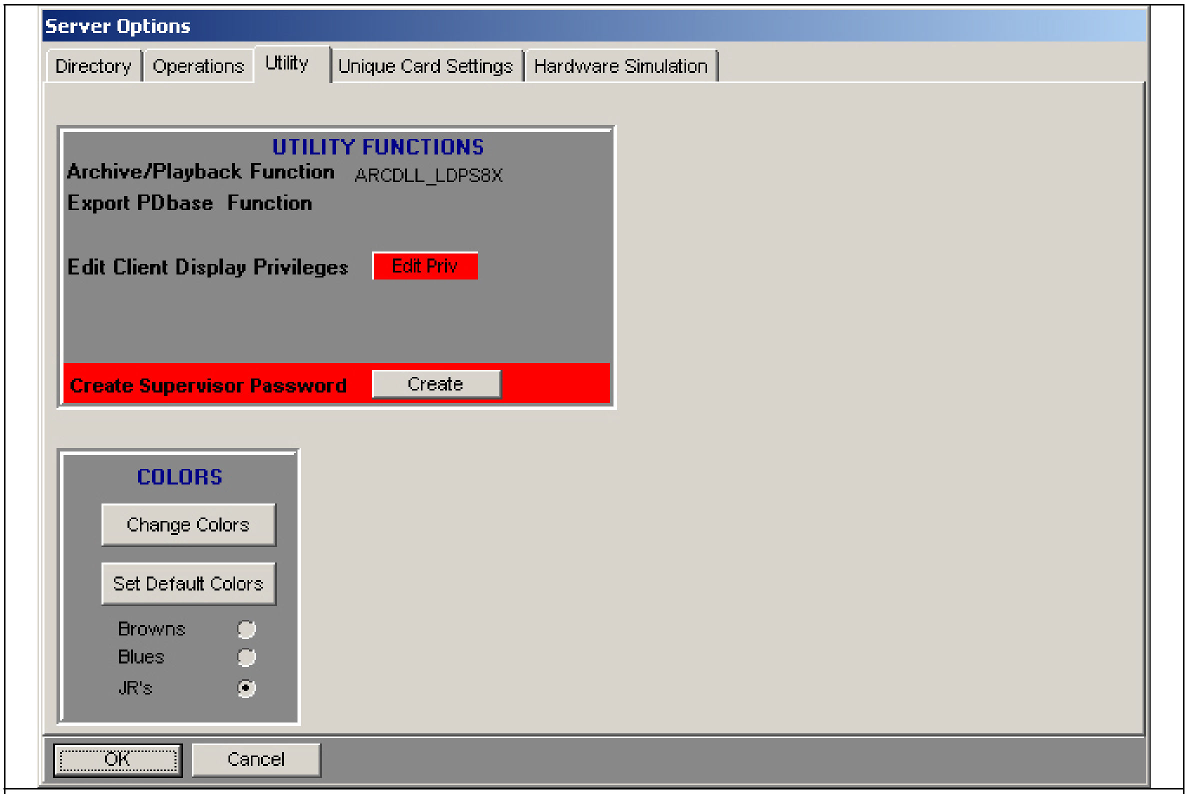
- **Load Last Project** – If this option is selected, then the last valid project loaded when LDPS was shut down will automatically load when LDPS is started up again.
- **Start Client** – If this option is selected, the client program will automatically start, provided that this is not the backup server. Note: If the client is already running before the server begins running, and this option is set, then an error message will result. In this scenario, simply accept the error message and continue (two clients are not to be running on the same machine!).
- **Disable Network** – If this option is selected, and the system is configured with the Pro version of LDPS and has a network card installed, the user may elect not to use the network for communications with the client with this option. The server must be restarted in order for this selection to take affect. Note: The backup server functions will not work with this option selected.
- **Force Single Processor** – For systems that have multiple processors, or support hyper-threading, the user may wish to force the server application to only use a single processor (or single virtual processor). This option is intended for systems that may have somewhat less than state-of-the-art motherboards. If this option is selected, then the affinity mask for the processor will be such that LDPS uses the first processor only. If multiple applications are running, then change the affinity mask for those applications to something different. For systems with single processor motherboards, this option has no affect. Note: The intent of this option can also be accomplished with the task manager, which sets the affinity to the desired state automatically.
- **Set Application Priority High** – The intent of this option can be accomplished with the task manager on the fly, however, selecting the option here sets the priority to high automatically.
- **Display Last Hw Setup** – By selecting this option, if there were hardware setup screens open when the server was shut down, then they will automatically reopen when the server is restarted. This option has a dual purpose when the server is in

Live mode. If selected, the setup displays will reopen automatically after the project is loaded.

- **Enable Aux TCP Output** – If this option is selected, and the network is enabled, then the Aux TCP outputs will automatically start up.

2.11.3 Utility Tab

The Utility Tab, shown in Figure 2-12 below, allows the user to setup archive and playback functions as well as specific parameter database export utilities. The utility function also allows the user to edit client display privileges and create the supervisor password. Finally, the colors section allows the user to completely customize the color schemes used for all of the windows and displays used in LDPS.

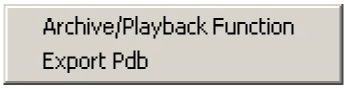


Archive/Playback Function ARCDLL_LDPS8X

○ Browns
○ Blues
● JR's

Figure 2-12 Server Options - Utility Tab

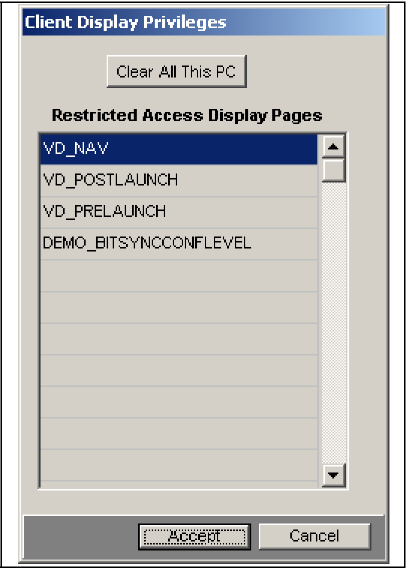
To access the menu in the **Utility Functions** section, place the mouse cursor in the area and right click to invoke the menu shown right



Archiving and playing back data files can now be performed on any file format required by invoking the **Archive/Playback Function** command and selecting the appropriate format from the list of supplied DLLs. The native LDPS file format is the default. To playback (or archive) in another vendor's file format, select the file format DLL from the list. If the selected format does not support archiving in the third party format, then the format used will be the native LDPS format (the user will be notified of such when a project is loaded).

For those needing to export data from the parameter database in whatever format that is required, the **Export Pdb** command may be used. Click the export format function and select the appropriate DLL from the list. If a particular export function does not exist,[9] the user may write their own function and place the DLL in this directory.

By invoking the **Edit Client Display Privileges** command, the supervisor can protect client displays so that they can't be edited by the user. The resulting setup window is shown in Figure 2-13 right. This command is a supervisor password protected entry.



Restricted Access Display Pages

VD_NAV
VD_POSTLAUNCH
VD_PRELAUNCH
DEMO_BITSYNCCONFLEVEL

Figure 2-13 Client Display Privileges Setup

The first person that clicks the **Create Supervisor Password** button and creates a non-blank password will be the supervisor for LDPS. The password cannot be changed once created except by contacting Lumistar. If the password is created, then all sensitive areas will require this password to gain access. This includes administering the network, setting the display page privileges, etc.

By using the Colors area, the user may change the LDPS color scheme to just about anything. There are a few default settings that can be used. Select the default colors button of interest (currently Browns, Blues, JRs) and click the **Set Default Colors** button and the respective pre selected color scheme will occur. Otherwise, click on the **Change Colors** button and the menu of various color items shown left in Figure 2-14 will be displayed Use this window to setup the color schemes for both server and client applications.

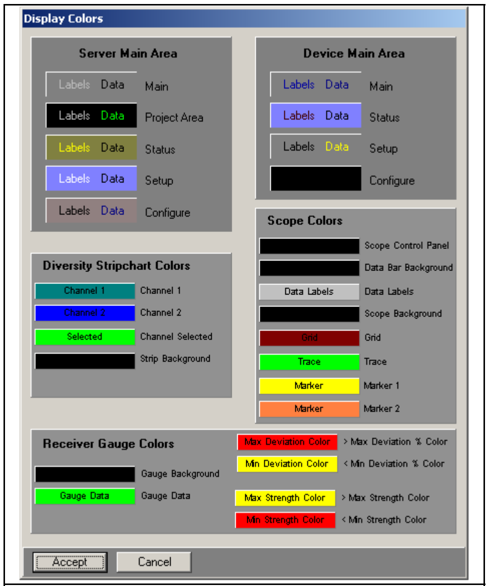


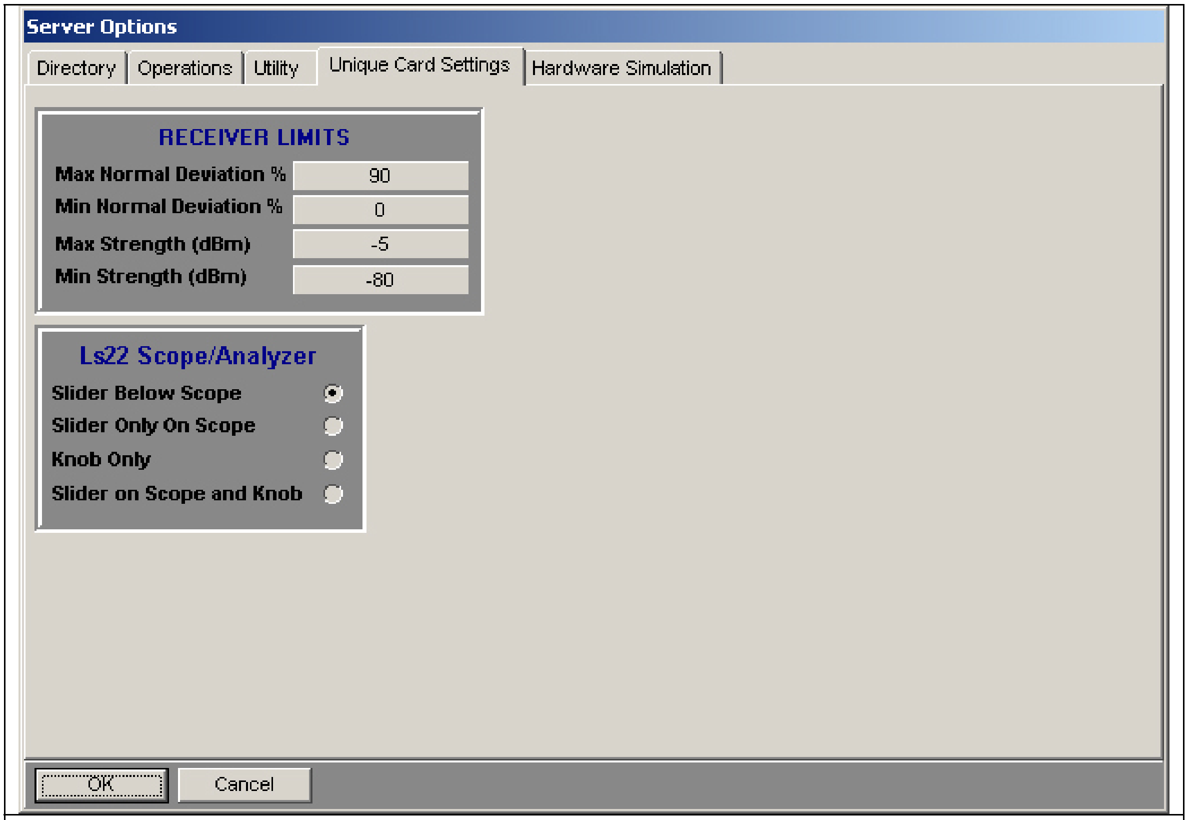
Figure 2-14 Display Colors Setup

[9] Lumistar can also write the export DLL per customer requirements. Contact custom support for more details about this process.

2.11.4 Unique Card Settings Tab

The Unique Card Settings Tab, shown in Figure 2-15 below, allows the user to setup certain hardware devices parameters that have unique settings. Currently, two Lumistar cards are supported on this tab. They include the LS-25 receiver, and the LS-22 Spectral/Time-Domain display.

The **Receiver Limits** section allows the user to set numerical values for the minimum and maximum signal strength and normal deviation displays for both server and client applications. To enter a value, click the respective button and enter the numeric value in the resulting dialog box.



RECEIVER LIMITS

Max Normal Deviation %	90
Min Normal Deviation %	0
Max Strength (dBm)	-5
Min Strength (dBm)	-80

Ls22 Scope/Analyzer

Slider Below Scope ●
Slider Only On Scope ○
Knob Only ○
Slider on Scope and Knob ○

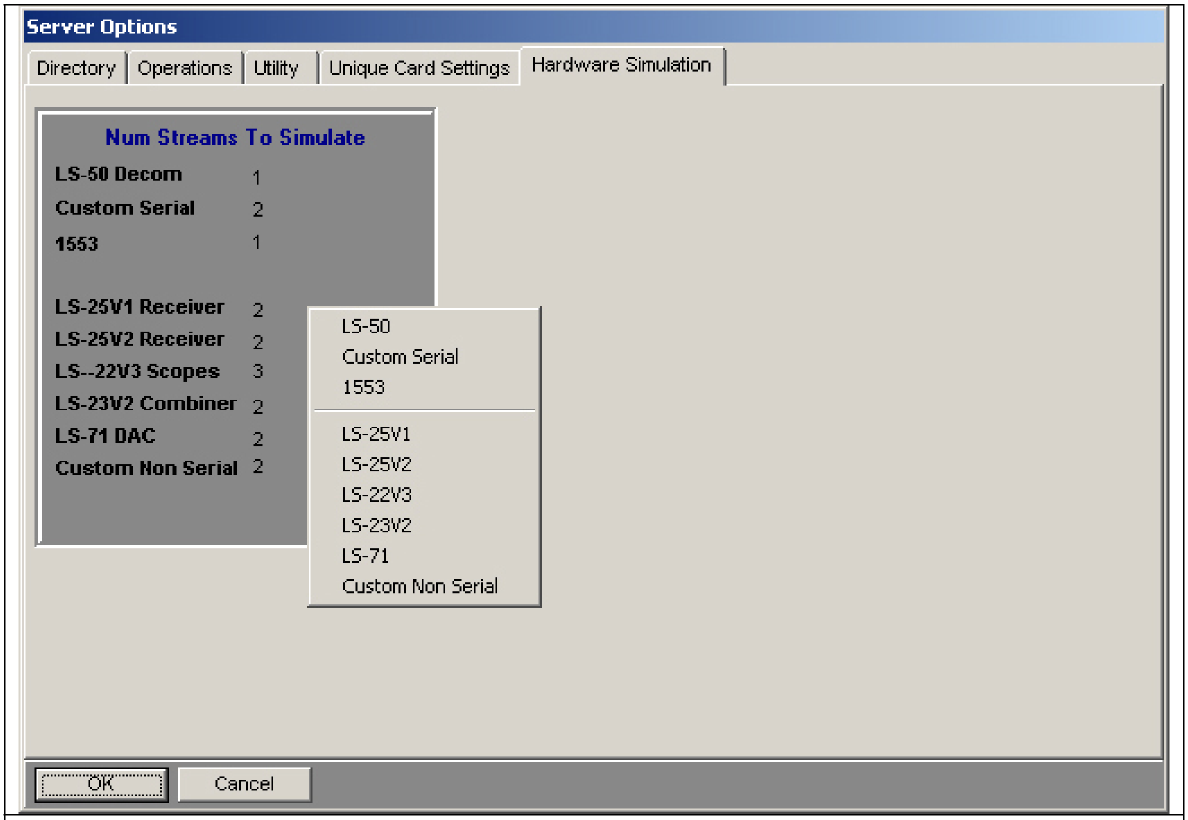
Figure 2-15 Server Options - Unique Card Settings Tab

The LS-22 Spectral/Time-Domain display card have controls that can either be manifest as a slider or a knob. For laptop users, the slider control works best. For systems that have a mouse pointing-device, the knob can make the display look more like a real oscilloscope. For more information on this feature, see paragraph 3.6.6 on page 160. The **Ls22 Scope/Analyzer** section allows the user to select from one of four possible control configurations.

2.11.5 Hardware Simulation Tab

The Hardware Simulation Tab, shown in Figure 2-16 below, allows the user to setup the number of streams that LDPS will simulate for each card type supported by the system. Note: The “Sim” check box for the particular card on the server device manager (see paragraph 2.12 on page 63) must also be checked to enable hardware simulation of the device. To set the number of streams for a device, place the cursor in the **Num Streams To Simulate** area and right click. Select the device from the list and enter the number of streams in the resulting dialog box. Note: The 1553 card is a special case. It can have only have one stream, but that stream has eight (8) busses.

If an archive file is played, and the current configuration doesn’t have enough streams assigned to the hardware application (live or simulated), LDPS will change the number of streams for the device to allow the playback to occur with the correct number of streams.



Num Streams To Simulate

LS-50 Decom	1
Custom Serial	2
1553	1
LS-25V1 Receiver	2
LS-25V2 Receiver	2
LS--22V3 Scopes	3
LS-23V2 Combiner	2
LS-71 DAC	2
Custom Non Serial	2

Figure 2-16 Server Options - Hardware Simulation Tab

Is should be noted that any change of these options will not take affect until the device application is restarted through the server device manager, or the sever program is restarted.

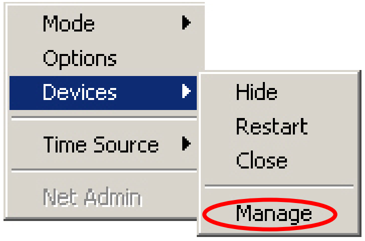
2.11.5.1 Card Simulation Notes

In an effort to avoid confusion in scenarios where cards are in simulation mode, several types of reminders are issued by LDPS. First, the caption in the GUI for the respective card will flash the word "SIMULATION" at a 2 Hertz rate. Also, an annoying sound is made with a message alerting that the card is in simulation mode in the follow conditions:

- The server starts up for the first time.
- A project is loaded that uses the card.
- A project is closed that uses the card.
- The server mode is changed from playback to live mode.

2.12 Server Device Manager GUI

The server Device Manager window, shown in Figure 2-17 below, controls which hardware devices will be used in the system at any give time by merely starting and stopping the device programs. The device manager is not available if a project is loaded or if LDPS is in playback mode. To invoke the device manager, select System → Devices → Manage from the main server menu as shown right.



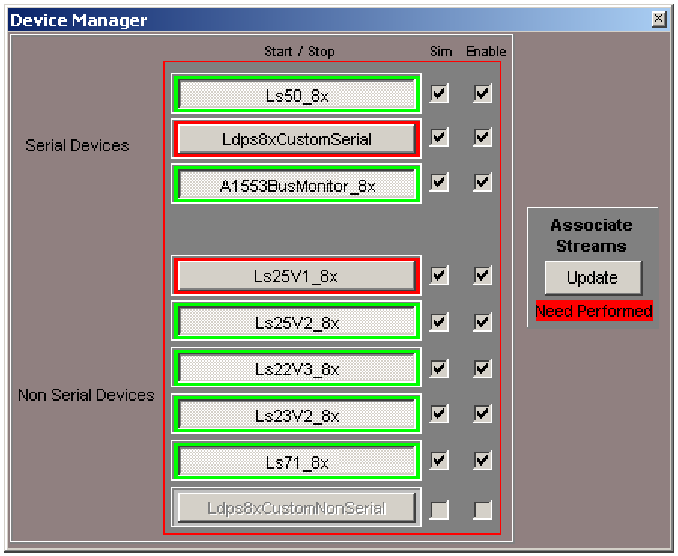


Figure 2-17 Device Manager Window

Access to the device manager is controlled by the supervisor password. If the supervisor password option is enabled, then the user must enter the correct password to have access to the controls.

When the server is first installed and started, the device manager window shown in the figure above will open automatically. If no hardware devices are assigned, this window will always be displayed. In general, the user needs to access this window only when hardware is installed or removed.

There are several hardware device programs associated with LDPS. As the need grows, additional programs can be added to LDPS with minimal effort or change to the server application. In LDPS there are two main categories of hardware device applications. Currently, the following hardware device applications available are:

The Serial Applications (DLL names) include:

- Ls50_8x – Ls50 multifunction decommutator card.
- Ldps8xCustomSerial – Allows ingest of serial data from any source.
- A1553BusMonitor_8x – Specialized custom application to ingest serial data from third party MIL-STD-1553 cards.

The Non Serial Applications (DLL names) include:

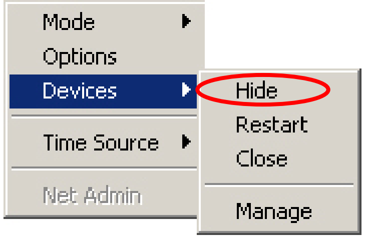
- Ls25V1_8x – Ls25 version-1 Receiver card.
- Ls25V2_8x – Ls25 version-2 Receiver card.
- Ls22V3_8x - Ls22 version-3 Oscilloscope/Spectrum Analyzer card.
- Ls23V2_8x – Ls23 version-2 Diversity combiner card.
- Ls71_8x – Ls71 DAC card.
- Ldps8xCustomNonSerial – Allows ingest of non-serial data from any source.

There are three controls for each application on the device manager window shown in Figure 2-17 on page 63. If **Enable** is checked, then two additional controls for the device become available. Also, if enable is checked the program for the device will startup automatically when the server program starts running. If **Sim** is checked, then the hardware device program will startup in simulate mode. Most device programs will simulate having at least two cards, with dynamic data. The simulation function is handled by the hardware device DLL. The server program doesn't care if the data is simulated or not, and the only indication of simulated data is on the device manager window, and in the caption area of the hardware device GUI.

In the **Start/Stop** area are individual buttons that allow the user to start and stop the device applications. This allows the user to reset an individual card if required. Once the device application is started, the button will turn green. If the device application is enabled, but not started, the button will be red. Also, if the device application is started, but there are no cards installed (real or simulated), the button will be red (See paragraph 2.12.2 on page 65 for more on no cards installed).

When the mode is changed from simulate to non simulate, the hardware device application will shut down and restart in the correct mode.

To the right of the Start/Stop & Sim/Enable controls is the **Associate Streams Update** button. If the state of a hardware device is modified, the user may or may not have to press this button. If an update is required, the red text "Needs Performed" will appear below the button, and the background color around the button will change from green to gray. As a matter of course, it doesn't hurt to press this button anyway if a device application is turned on or off.



The individual hardware device application GUIs are docked to the server application and can't be moved around. They can, however, be minimized (all of them or none of them) by selecting the **Hide** command from the device menu as shown right (red oval). To shut them all down, select the **Close** command from the menu, and to restart them all, select the **Restart** command from the menu. Note: the close command will not be available if a project is loaded.

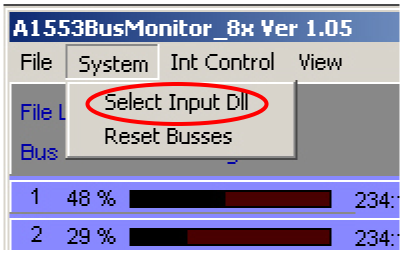
2.12.1 Playback Notes

One can play back archived data on a system different in configuration than the original the data was recorded one. The second system may have more or fewer hardware assets than the original system. If a project is playing back on a system that has more cards in it than the original system, the device programs will start up in simulate mode. This ensures having the correct number of cards for playback.

When going back into live mode on the alternate system, the cards will still be set to simulate mode because of the playback. To change the mode, the user must invoke the device manager and put them back into the correct state. Disabling the simulate mode in this scenario cannot be performed automatically.

2.12.2 Notes on "No Cards Installed"

A device application can be started with no cards detected. In this case, the start/stop button will be red. The No Cards Installed condition can occur because there are no physical cards found by the hardware device application (and not in simulate mode), or, no input DLL is assigned for the device, or the input DLL assigned is reporting no cards.



The latter (no input DLL is assigned) can happen on three of the hardware device applications because of the nature of the interface these applications have. These special applications allow input from any device. Therefore, a device DLL has to be written to collect data from the device. The user can choose which DLL to use from the system menu of each of the individual hardware device applications as shown right *(System → Select Input DLL)*. The three special applications include: Custom Serial, Custom Non Serial, and the 1553 application.

2.13 Server Playback Functions

LDPS has the ability to archive and/or playback data files from any vendor's system (not just the native LDPS format). This is achieved by having common calls to a DLL, which is responsible for reading and/or writing data in the required format. The DLL translates the vendor data to/from the LDPS data format. Most of the time, archiving is performed in the native LDPS format. If required, data may be played back in a different format.

Several formats are supplied with LDPS. Contact Lumistar for support of a specific vendor format. If the function is not currently available, the user may develop their own archive/playback DLL. Lumistar can also develop an archive/playback DLL for a given format. Contact customer support for details.

For LDPS to playback a different vendors' file format, there are several steps to go through. The easiest and most painless method is to perform the following steps:

1. Collect information on the vendor's decommutator setup, and create and save a setup in LDPS to match. Do this for each stream (if there are multiple streams). For 1553, there is only one stream available, with up to eight (8) busses. If there are no 1553 cards in the LDPS system, then there won't be a 1553 setup to access. In this case, from the device manager, configure the LDPS system to use the "Example1553Card" for its 1553 input type, which will give eight busses of SIMULATED data in live mode. This will allow a setup file to be saved for use in making a project. During playback, if the vendor archive doesn't have that many busses, then the data for those busses will be zero.
2. Collect information on the parameter data dictionary for each stream in the vendor's archive. Then convert the data dictionary from the vendor's format using the supplied LDPS tools and save the parameter database in LDPS format (see paragraph 1.3.8 on page 21 for more information). To get a parameter database generated quickly, without concern for contents, use the PDbase Editor described in paragraph 4.3.1.4 on page 199 to auto generate one based on the decommutator setup file. For 1553 stream, at least one parameter must be created.
3. Create a project to run a live mode. HINT: Indicate the archive format to use in the NOTES section of the project so as to keep track of what project requires what playback function.
4. Ensure the server options have the native Archive/Playback function selected from the utility functions tab described in paragraph 2.11.3 on page 59. Select the "ArkDll_Ldps.Dll" from the list.
5. Load the project and press the **Record** button for one second, then close the project. When this step is performed, all of the necessary files are copied to the archive directory (as defined in the server options), including the native ".ARK" file. The native file will not be used, but the rest of the files are.
6. Copy the vendors' archive file to the archive directory.
7. Switch the server options to the specified vendors' format.
8. Now the other vendors' data file can be played.

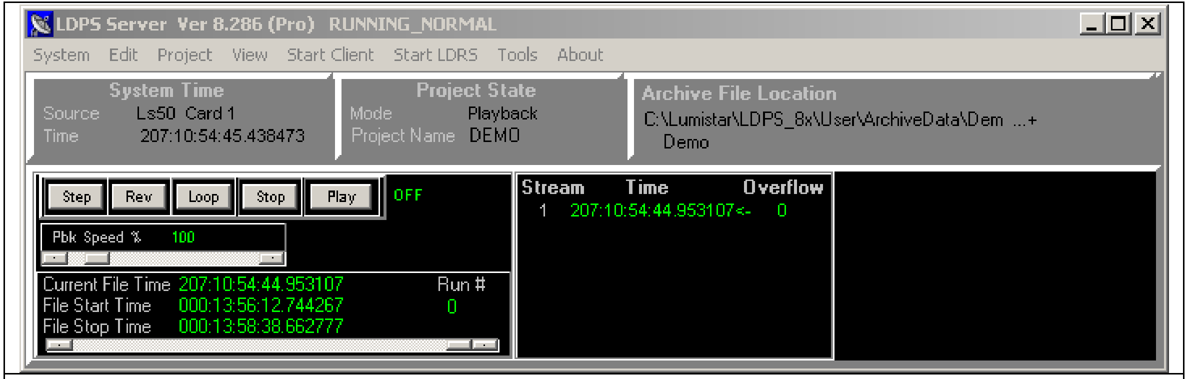
Actual hardware is not required in order to perform the eight steps described above. Simply set the hardware device applications into simulate mode and proceed.

Note: The archive function (DLLs) MUST reside in the subdirectory under the LDPS root directory called "BIN\ARCHIVEDLLS". This path is not an option.

2.14 Playback Control

LDPS has the ability to playback files archived on the server. It does not have to be the same server, as long as access to the required files is assured. The ability to playback data is crucially important for any kind of post-test analysis. In many cases, the data may be too fast for a human to ingest in real time, or the data may be too fast for the machine to process in real time (or both). In LDPS, the archiving process has the highest priority on the machine, and LDPS will attempt to archive 100% of the data. Assuming all of the data was archived correctly, the user now has the ability to play the data back at a slower rate, and view it over and over again for analysis and interpretation.

When the system is placed in playback mode and a project is loaded, the playback controls displayed in Figure 2-18 below will appear at the bottom of the server window. In addition to the controls, the display also contains the run number present during the archive. Once a playback project is loaded, the playback control comes to life and the user can control the data on the server (which in turn translates to control of the data on the clients).



LDPS Server Ver 8.286 (Pro) RUNNING_NORMAL

System Time
Source Ls50 Card 1
Time 207:10:54:45.438473

Project State
Mode Playback
Project Name DEMO

Archive File Location
C:\Lumistar\LDPS_8x\User\ArchiveData\Dem ...+
Demo

OFF

Pbk Speed % 100

Current File Time 207:10:54:44.953107 Run # 0
File Start Time 000:13:56:12.744267
File Stop Time 000:13:58:38.662777

Stream	Time	Overflow
1	207:10:54:44.953107<-	0

Figure 2-18 LDPS Archive Playback Controls

2.14.1 Control Options

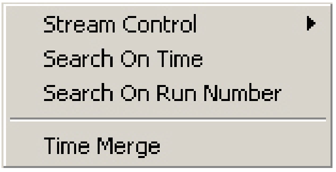
The archive playback function has seven (7) controls that are described in detail in the following paragraphs.

If the **Step** button is depressed, and then the **Play** button pressed, then one (and only one) buffer of data will be read into the system. If the step button is not depressed, then the buffers of data are continually being read in at the playback speed rate selected by the user. Depressing the **Reverse** button will play back the buffers of data in reverse order.

If the **Loop** button is depressed, then the file will start playing over again once the end of the file is reached. If playing in reverse, then the playback will start the end of the file when the beginning of the file is reached.

As might be expected, the **Stop** button will halt playing of the data (assuming the Step button is not depressed). Likewise, the **Start** button will begin playing the data.

Each stream archived will have its own set of controls and indicators. Right clicking in the playback control area will invoke the options menu (shown right) needed to change the stream and allow one to search for data based upon time or run number. The controls in this menu are as follows:



- **Search On Time**– This button searches the file for a particular time. Enter the time to search for when prompted. To go to the beginning of the file, enter zero (0.0). Otherwise, enter the time to search for in the format "ddd:hh:mm:ss.xxx" where there are 3 digits for the day of year, two digits for the hour, two digits for the minute, and two digits for the second. Milliseconds may be omitted, however, if it isn't then there must be three digits for the milliseconds. There must be a colon (:) between all fields except between seconds and thousandths of seconds, which must be a period (.).
- **Slide Bar** – The slide bar at the bottom gives an indication of how far along in the file the playback is. Click and drag the slide bar to position the file pointer to a specific time point in the file.
- **Search On Run** – Run numbers are archived along with the data. Searching on a run number will position the file pointer somewhere in the run being searched for. Enter the run number when prompted. If there are separate archived files for each run number, then the appropriate file will be recalled.
- **Time Merge** – For multiple streams, this command will merge the playback so the streams are aligned in time. Enter the time "hack" to search for when prompted in the format "ddd:hh:mm:ss.xxx" (see search on time above). Note: The streams must have the same relative time for this command to work.
- **Pbk Speed** – This slide bar regulates how fast the data is played back (while not in Step mode). The numeric value is a percentage of the real-time speed. If the slide bar is all the way to the left (0 %), then playback will go as fast as possible. The percent of playback speed is based on the clock rate the decommutator was receiving when the archive file was closed. If the lock state was lost during the archive, and the decommutator was drifting such that the clock rate was not accurate, then this slider is meaningless for the actual percent of real-time rate. In this scenario, the speed of playback will vary. Note that even though the slider may be set to 900%, the computer may not be able to keep up with this rate, and will not achieve it.

To the right of the playback controls are timestamps for the current time, the time the file was created and the time the file was closed.

During operation, LDPS archives at least two files-per-stream, sometimes three, depending on the processor power. However, these files are recorded at different rates, so on playback, they have to be merged in time. This is done automatically, but ONLY if the Loop button is not depressed. The algorithm for merging is (if playing in reverse then logic reversed)

```
Get time this stream
Get time latest stream
If (latest stream not this stream)
    If (this time <= latest time) then ok to output data
```

The data received by the client is not always guaranteed to be 100% of the data sent from the server, even in playback mode. This is because there are just too many factors that can cause data loss (like processing amount, minor frame rate, supercom data rates, PC CPU speed, etc). To determine if all of the data is being processed, look at the overflow counters for the stream(s) in question. In playback mode, one can control the playback speed on the server via the playback speed throttle described above. Slow the down the speed until the overflow counters stop incrementing. Typically, data update rates of 200 Hertz or less will not cause problems. For update rates faster than that, the numerous factors that can cause data loss begin coming into play. To help mitigate this problem of data loss, select the option, **Playback-Cont, at least 1 msec delay btwn frames** described in paragraph 2.11.2 on page 55. This option applies a 1 msec delay between minor frames. This ensures that there is at least 1 msec per minor frame delay to allow the client to process all the data before the next buffer of data is sent to it.

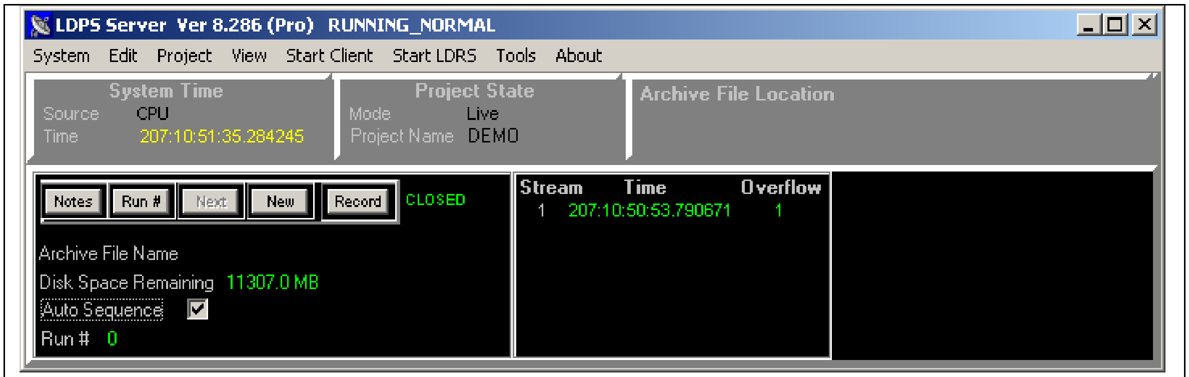
2.15 Server Archiving

The archiving of data on the server is performed at a very high priority in the system. Other tasks such as display or broadcasting data takes a back seat to the archive function. If data is arriving at a high rate, some sluggishness of the program response may be noticed during archive operations. The reason the archiving task is so high in priority stems from the fact that data from a live test can never be re-captured, nor is the data ever exactly alike from one test to the next.

The rate at which data can be archived (without losing data) varies, depending on the speed of the machine, the speed and status of the disk, the stream setup (affecting interrupts for new data), and the number of streams. If the archiving of high data rate streams is planed, it is recommended that the user spend the extra money to get a system with a high-speed disk and a fast processor.

When a project is loaded in live mode, an archive control window like the one seen in Figure 2-19 on page 70 will be created to control the archiving function. This control is located at the bottom left of the server window. If the options are set to allow automatic archive filenames (see paragraph 2.11.2 on page 55), then there is nothing to do except press the recording button. The archive control employs five (5) buttons, from left to right, that perform the following functions:

- **Notes** – The user may place notes in the archive file. Up to three lines of text can be placed in the header of the archive file at any time before the file is closed.
- **Run #** – Pressing this button will increment the run number by 1. It cannot decrement the run number. If the option to create a new archive file on each run number is selected, then pressing the run button will close the current archive file and start a new one. The new archive file will have the same primary name, except it will have a new run number appended.
- **Next** – The next button is disabled until some data has been recorded. Once enabled, this button will increment the run number, and then sequence to the next archive file based on the run number.
- **New File** – Press this button to start a new archive (closes any currently in use). The user is prompted for the name of the new archive file. Take care when using this button when the auto-create archive filename option is selected. A new directory will be created and the archives will go into the directory. This may or may not be desirable because of data management issues. This button DOES NOT have to be pressed for the FIRST archive file. Only press it to close the current or FIRST archive file and start a new one.
- **Record** – This button turns recording on and off. If depressed, it will flash the button text at a 1 Hertz rate indicating that recording is taking place.



LDPS Server Ver 8.286 (Pro) RUNNING_NORMAL

System Time
Source CPU
Time 207:10:51:35.284245

Project State
Mode Live
Project Name DEMO

Archive File Location

CLOSED

Stream	Time	Overflow
1	207:10:50:53.790671	1

Archive File Name
Disk Space Remaining 11307.0 MB
Auto Sequence ☑
Run # 0

Figure 2-19 LDPS Archive Recording Controls

Below the archive control buttons is the **Auto Sequence** check box. Selecting this overrides the option to start a new archive file when the run number is bumped. There is also status information on this window that indicates how much disk space is left and the name of the current archive file

When a file is opened for archiving, the location is defaulted to the archive file directory and a subdirectory under that (assuming one opted for any kind of subdirectory). The subdirectory is automatically created. See the archive directory options in Figure 2-10 on page 54 for subdirectory naming schemes. This mechanism helps to keep the location and names of data files straight. The user may change where the default directory is, or where a particular archive session directory is located.

NOTE. Many projects use the same device files. If one changes the device setup file and always archives in the same place, one risks overwriting a playback device setup file, which can cause the playback project load to fail.

Archiving can only take place if a project is loaded and LDPS is not in Playback mode. The project can have up to twelve (12) streams in it. A data file is generated for each stream, and has the file extension, ".ARK." The archive files' name is composed of several elements as follow: Immediately prior to the file extension is the stream number and the run number. Before the stream number is the project name. Before the project name is the prefix text selected by the user. All of these elements depend on the options selected for archiving. If one did not opt for prefix text in the filename, there won't be any. If one did not opt for sequential files on run number change, then there won't be a run number. There will always be a stream number, however. As an example, assume a project name called DEMO with the default options set for archiving. The resulting filename will be **DEMO_STREAM_1.ARK.** If there were two streams in the project, the second file would be named **DEMO_STREAM_2.ARK**. When run numbers are selected, the filename would be **DEMO_STREAM_1_RUN_7.ARK** for run number 7.

The exception to the ".ARK" file extension is when one selects a different archive/playback function DLL (instead of the native "ArkDll_Ldps.Dll"). In this scenario, the file extension is determined by the function DLL (i.e. Chapter-10 file extensions are ".c10" for some vendors, and ".ch10" for others. TTC vendor files have a ".BIN" file extension).

In addition to the .ARK file recorded for serial stream data, a ".STAT" file is archived as well. This file contains the status data of the devices used in the project. The status data is recorded at a 20 Hertz rate (this keeps the size of the serial data files down to a more reasonable size).

In scenarios where a specialized piece of hardware is used in the project (like an LS-22), another file is archived with an ".NSDAT" extension. This specialize hardware data contains the non-serial data for devices such as a spectrum analyzer, for example.

When the server opens a project for archiving, several files are copied to the archive directory as well as generating a playback project file. These files vary, depending on how the project is set up. If a file is used during live mode for archiving, that same file is subsequently copied to the archive directory for playback use. If one intend to playback the archived data on a different machine, then all of the files in the directory created during the archive will need to be on that system.

Archiving normally takes up a lot of hard disk space. It is also very difficult to know how each user will set up where the archived data is stored. For this reason, it is recommended that after the test is complete, the newly created files in the archive directory be moved to a different location for longer-term storage. This may be on a different machine, or disk.

The user can elect to open the archiving function in a directory other than the Archive directory. The required files will be copied to this location. The method of naming the location of the archived files is up to the user, but the default directory is:
LDPS_8X\User\ArchiveData.

2.15.1 Archive Format

The native LDPS format for the archive file is very simple. The first part of the file is ASCII text providing information about the file. Each line of text is followed by a carriage return and line feed. After the information about the file, a NULL character is generated. After the NULL character, the data starts. The header is in ASCII so as to make examining the file with any program possible. The ASCII part even instructs how to read the data. There are two basic parts to the file, an ASCII file header, and the data. Each is described in the following numbered paragraphs.

2.15.1.1 Archive Header

The file header structure used in the LDPS ASCII File Header is as follows:

```
//archive header info
typedef struct //stuff contained in the archive header
{
  int FileVersion;       //ark file version
  double StartTime;      //time (toy) archive file opened
  double StopTime;       //time (toy) archive file closed
  int BlockSizeTotal;    //number of bytes written/read per record = blocksizemisc +
blocksizeserial + blocksizedevice
  int BlockSizeMisc;     //number of bytes for misc data (run number, time source, etc)
  int BlockSizeSerial;   //number of bytes for the dbase data per record
  int BlockSizeDevice;   //number of bytes for the device tags per record
  int StreamId;          //stream id this was (0 based)
  int SerialWpf;         //words per frame in serial Device buffer
  int SerialNumSf; //num subframes in major frame
  int SerialSfStartCount; //sfid counter counts from 0 or 1
  int SerialSfIdWdNum;   //sfid word number (0 based) if used, 0 if not used
  bool SerialRightAligned; //data from device is right aligned or left aligned
  int SerialFpi;         //frames per inteerupt in serial Device buffer

  int HwProgram; //the type (see enum ehwprogramindex) hardware that created the serial
stream
  int CardIndex; //the card index of the program that created the serial stream (0 based)
  AnsiString HwSetupFn; //the hardware setup filename that was used

  AnsiString StatusTagNameBaseAry[MAXHWCARDSTATUSVALUES]; //the status tag names (base,
no stream id)

  double SerialFrameRateHz;  //frame rate in the serial Device buffer in hz, used for
playback
  AnsiString NotesAry[3];    //notes user has put into the ark file
  AnsiString Instructions; //how to read the ark file
}arkheaderstruct;
```

The LDPS program parses the ASCII text in the archive file to fill in the structures defined above. If one is writing a custom archive file reader, the structure may be different, but one can still get the information about the file from the ASCII header. A sample archive header is shown below.

```
LUMISTAR 8.0 ARCHIVE FILE FORMAT
Archive File Version :11
Start Time :233:08:10:50.693502  YEAR 2005
Stop Time  :233:08:11:15.833751  YEAR 2005
BlockSizeTotal (bytes) :1126
BlockSizeTimeStamp :8
BlockSizeMisc (bytes) :4
BlockSizeSerial (bytes) :1034
BlockSizeDevice (bytes) :80
Stream ID (0 based) :0
Serial Words Per Frame :512
Serial Num Subframes :64
Serial Subframe ID Start Count :0
Serial Subframe ID Word Number (0 based) :3
Serial Data Right Aligned :YES
Serial Frames Per Interrupt :64
Serial Data Application :1
Serial Data Card Index :0
Serial Data Setup Filename :F:\LUMISTAR\PROGRAMS\Ldps_8x\USER\SERVERFILES\DDBASE\DEMO.B50
Status Tag Name :D__50_DATAVALID
Status Tag Name :D__50_DEC_MAJFRM
Status Tag Name :D__50_DEC_MINFRM
Status Tag Name :D__50_DEC_TOY
Status Tag Name :D__50_DEC_CLOCK
Status Tag Name :D__50_DEC_RTALIGN
Status Tag Name :D__50_IRIG_TIME
Status Tag Name :D__50_BIT_STATUS
Status Tag Name :D__50_BIT_CONF
Status Tag Name :D__50_IRIG_STATUS
Status Tag Name :D__50_SPARE1
Status Tag Name :D__50_SPARE2
Num Serial Stream Header Words :5
Serial Frame Rate (Hz) :100.0
....PROJECT NOTES....

....END PROJECT NOTES....
```

After the header, Data will read System TimeStamp (a double[10] time of year in seconds), then Spare1 (signed byte), then Spare2 (signed byte), then Run number (an unsigned short (two bytes)), then 33,088 Tags for data stream of WORD (two bytes per word, which includes five header words per minor frame), then twelve Tags for the device values of DOUBLE (eight bytes per tag) and then repeat itself. After that, the rest of the header is fill data, with the fill character of “A”. The total header size is 5,120, not including the NULL character.

The first line " LUMISTAR 8.0 ARCHIVE FILE FORMAT" is a string used to identify the file as an archived file from the LDPS program. The next line is the archive format version number (not expected to change in the foreseeable future).

The next two lines are the start and stop times of the archived data. The data is represented as xxx:xx:xx:.xxxxxx for Julian day, hour of day, minute of hour, second of minute, number of microseconds. This is followed by two spaces, then "YEAR" followed by the year, i.e. "Start Time :024:05:13:27.552527 YEAR 2001"

The next five lines provide information on how to read a block of data, where:

- BlockSizeTotal (bytes) : - The total number of bytes to read for one block.

[10] Double, as in a double precision data format.

- BlockSizeTimeStamp : - The number of bytes used for the timestamp
- BlockSizeMisc (bytes) : - The number of bytes for run number, time source, spare
- BlockSizeSerial (bytes) : - The number of bytes for the serial data.
- BlockSizeDevice (bytes) : - The number of bytes for the device data.

The next line is the stream number of the archived file, zero based (i.e. the first stream number is 0, the second stream number is 1, etc.). The rest of the information in the header is fairly self-exclamatory (mostly decommutator setup information).

Following the stream number is information regarding the device status. This is the device category and card number, and the tag names for the values in that category. They are archived in the same order as listed in the header.

Following the device category data, there is a line of text "....PROJECT NOTES....". If one is using their own reader, look for this line of text to indicate the beginning of the production notes text. After this line there are three lines of PROJECT STREAM NOTES the user inserted when the archive was created. After the three lines of text is "....END PROJECT NOTES....", indicating no more user notes. (Use this method incase more notes will be added later).

Once all the device category information is written, a single line of text informs the user in plain English how to read a data block.

After the header, Data will read System TimeStamp (a double time of year in seconds), Spare1 signed byte, then Spare2 signed byte, then Run number (an unsigned short (two bytes)), then X Tags for data stream of WORD (two bytes per word, which includes five header words per minor frame), then Y Tags for the device values of DOUBLE (eight bytes per tag) and then repeat itself. . After that, the rest of the header is fill data, with the fill character of "A". The total header size is 5,120, not including the NULL character.

This text is always the same, with the exception of the single quoted 'X' and 'Y ' and the number of fill bytes. These values are filled in, depending on the format.

2.15.1.2 Archive Binary Data

The data portion of the archive file consists of six parts:

1. Time Stamp – A double precision number representing the time of year, in seconds, the data block was archived (using Stream Time for the time stamp).
2. Spare Byte 1 – For future use. Set to 0.
3. Spare Byte 2 – For future use. Set to 0.
4. Run Number - A two byte unsigned short for the run number the user had during archiving.
5. Serial Data Block – X 16 bit words collected from the serial device for this stream. X is defined as "Number Data Parameters" found in the ASCII header.
6. Device Data Block – X double precision values (eight bytes each) collected from each of the hardware devices associated with the stream. X is defined as

(NumDevices * NumTagsPerDevice for each device category) * NumDeviceCategories found in the ASCII header.

2.15.1.3 Notes About Time Sources

In the ASCII header, the start and stop times are the stream time. The block data time stamp is also the stream time. The stream time can sometimes come from embedded serial data in the stream and not from a reliable time source. If a reliable time source is not available, then one could put a counter in for time.

The user has the option to select the source for system time. It is up to the user to ensure the time source is valid. The time source comes from either the computer clock (poor resolution) or from a stream. If the user selects time to come from a stream, the user must ensure the time source is valid. (Some checks are made to ensure the time is valid by the LDPS, like time moving forward, but the accuracy is not checked. The accuracy will have to be gathered from the archived data.)

3 Machine Configuration

The process of machine configuration in LDPS involves setting up and configuring the individual hardware device applications associated with each card in the system. Currently, the hardware devices supported by LDPS include the following:

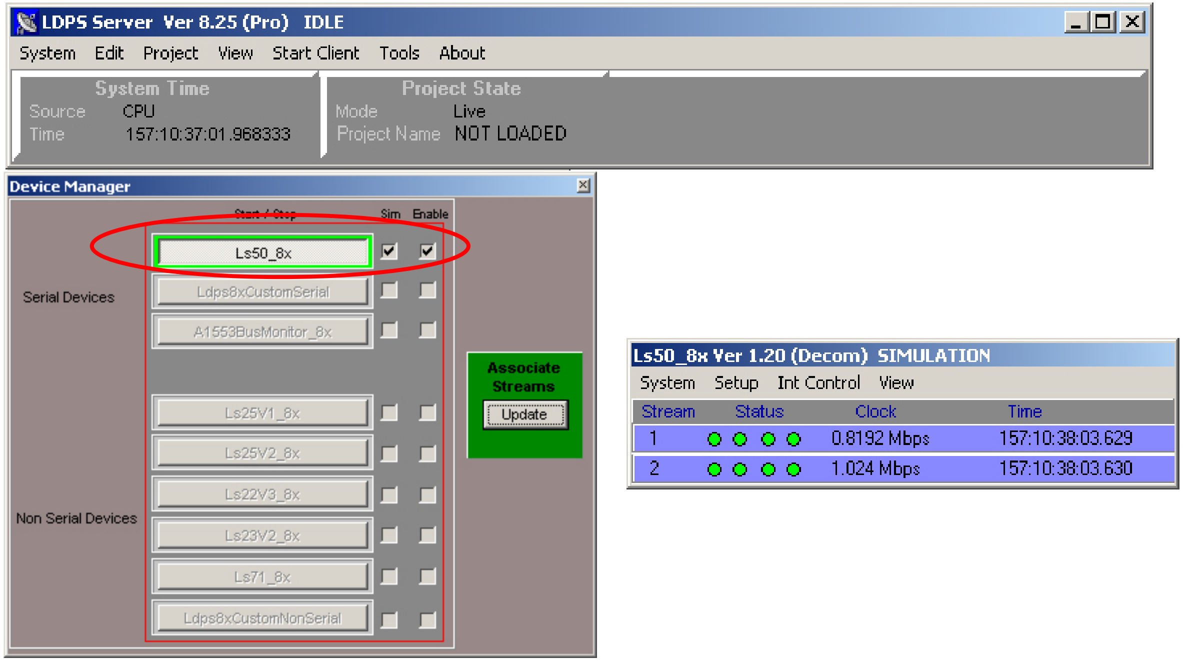
- LS-50-P/LS-55-DB Multifunction Decommutator
(see paragraph 3.1 on page 77)
- LS-25-D/LS-25 Telemetry Receiver (see paragraph 3.5on page 133)
- LS-22 Spectral/Time Domain Display (see paragraph 3.6on page 143)
- LS-23 Pre-Detect & Post Detect Diversity Combiner
(see paragraph 3.7 on page 163)
- LS-71 Multi-Channel Analog Output Card (see paragraph 3.8 on page 171)

LDPS also has support for third party serial and non-serial devices (see paragraph 3.4 on page 128), as well as third party MIL-STD-1553 cards (see paragraph 3.3 on page 120). All of the devices mentioned here are represented by individual control buttons on the device manager window. An example of this is shown in Figure 3-1 on the following page.

To configure the system, the user will follow the steps outlined below for each of the hardware devices installed in the system. For illustrative purposes, the initial configuration of the LS-50-P card will be presented in detail here, but is should be noted that the following four steps are the same for any hardware devices installed in the system.

To initially configure the LS-50-P, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select “Devices” and then “Manage” *(System→ Devices→ Manage)*
2. From the System Manager shown below left, select the “Enable” check box next to the Ls50 button. The “Ls50_8x” button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the “Sim” check box next to the “Enable” check box. Checking this box allows the LDPS application to operate when a LS-50-P board is not installed in the system.
3. From the System Manager, click the “Ls50_8x” button. This will launch the “Ls50_8x (Decom)” display shown below right. Note that the red rectangle around the button has changed to green indicating that the application is now running.
4. To setup and configure the LS-50-P card, follow the procedures outlined in paragraphs 3.2.

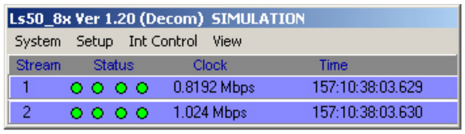


Source CPU
Time 157:10:37:01.968333
Mode Live
Project Name NOT LOADED

Stream	Status	Clock	Time
1	○ ○ ○ ○	0.8192 Mbps	157:10:38:03.629
2	○ ○ ○ ○	1.024 Mbps	157:10:38:03.630

Figure 3-1 LDPS Server Application Windows (LS-50 Enabled)

3.1 LS-50 Multi-Mode Decommutator Setup GUI


Ls50_8x Ver 1.20 (Decom) SIMULATION
System Setup Int Control View

Stream	Status	Clock	Time
1	○ ○ ○ ○	0.8192 Mbps	157:10:38:03.629
2	○ ○ ○ ○	1.024 Mbps	157:10:38:03.630

The decommutator setup GUI shown right is a very small window launched from the main application window on the Server. The user launches the GUI by selecting *Setup → Stream 1* on the main application window, or when a project is loaded. The decommutator setup GUI controls all multifunction cards installed in the system. The decommutator setup GUI has a multitude of other setup windows that are launched for setup, control, and status of the multifunction card. The decommutator setup GUI has two parts: a caption area and the menu area.

The caption area (top of the window) displays the name of the window, and if a simulation is running, the word "SIMULATION" will be flashing. Note there is no normal window closing mechanism in the caption area (the X normally found in the upper right corner of windows). This is because the window must be active in order to gather data from the multifunction card, and to control it.

Below the caption area and setup menu is the status display for all LS-50-P cards installed in the system. For each stream, the status display shows the system time, clock rate (Mbps), and major/minor frame lock condition, as well as the IRIG Time function status, and the Bit Sync lock status.

3.1.1 System Menu

The system menu has three commands including: Flash Board ID Leds, DMA Usage, and PLX Reset. To identify a specific LS-50-P card(s) in a system with multiple cards, the user may invoke the **Flash Board ID Leds** command, and select the desired stream number from the list of available streams. Remember that each LS-50-P card in the system has a unique and specific stream number assigned to it. For troubleshooting systems with DMA problems, the user may elect not to use DMA transfers of buffered data by invoking the **DMA Usage** command and again select the desired stream number from the list of available streams. Also for hardware troubleshooting is the **PLX Reset** command. This will do a soft reset of the PLX chip on the card. The result will be a loss of data for about 8 seconds.

3.1.2 Setup Menu

The setup menu allows the user to configure each stream in the system and the LS-50-P hardware associated with each stream. Selecting a specific stream from the list of possible streams will invoke the decommutator configuration setup screen shown in the lower portion of Figure 3-3 on page 79.

3.1.3 Interrupt Control Menu

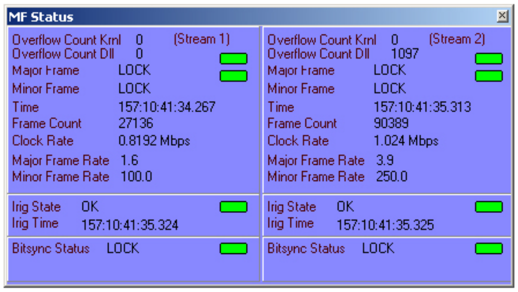
The interrupt control function allows the user to start or stop interrupt activity in unison for all LS-50-P cards installed in the system. The **Start All Decoms** command enables interrupts for all LS-50-P cards. When new data is ready from the LS-50s, they signal with an interrupt. The LDPS program picks up the data and stores/processes the data. This is also a convenient way to reload, or download, the setup data to the cards. When starting the interrupts in this manner, the cards are downloaded with their setup data first. The Stop All Decoms command stops interrupt activity on all LS-50s in the system.

3.1.4 View Menu

To display the status of all LS-50-P cards in the system, the user selects the **Status** command. This will launch a window similar to Figure 3-2 below that displays the status of all LS-50-P multifunction cards installed on the system. The example shown below corresponds to a system with *two* LS-50-P cards installed.

3.1.5 BERT Menu

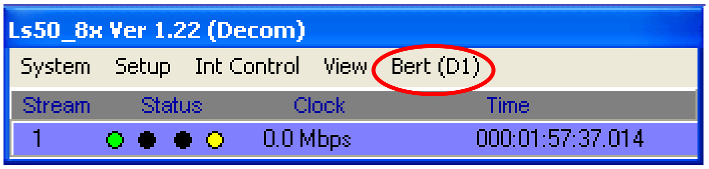
When actual LS-50-P hardware is installed in a system (presumably the default scenario), an additional item will appear in the decom server menu as indicated via the red oval shown below. Each LS-50-P card is equipped with it's own Bit Error Rate Tester (BERT) that may be configured by selecting the **Bert (Dn)** command (where "n" corresponds to the stream number). This will invoke the BERT setup configuration screen shown in Figure 3-14 on page 114



```
Overflow Count Krnl   0        (Stream 1)
Overflow Count Dll    0
Major Frame         LOCK
Minor Frame         LOCK
Time                157:10:41:34.267
Frame Count         27136
Clock Rate          0.8192 Mbps
Major Frame Rate    1.6
Minor Frame Rate    100.0

Irig State    OK
Irig Time     157:10:41:35.324

Bitsync Status    LOCK
```

```
Overflow Count Krnl   0        (Stream 2)
Overflow Count Dll    1097
Major Frame         LOCK
Minor Frame         LOCK
Time                157:10:41:35.313
Frame Count         90389
Clock Rate          1.024 Mbps
Major Frame Rate    3.9
Minor Frame Rate    250.0

Irig State    OK
Irig Time     157:10:41:35.325

Bitsync Status    LOCK
```

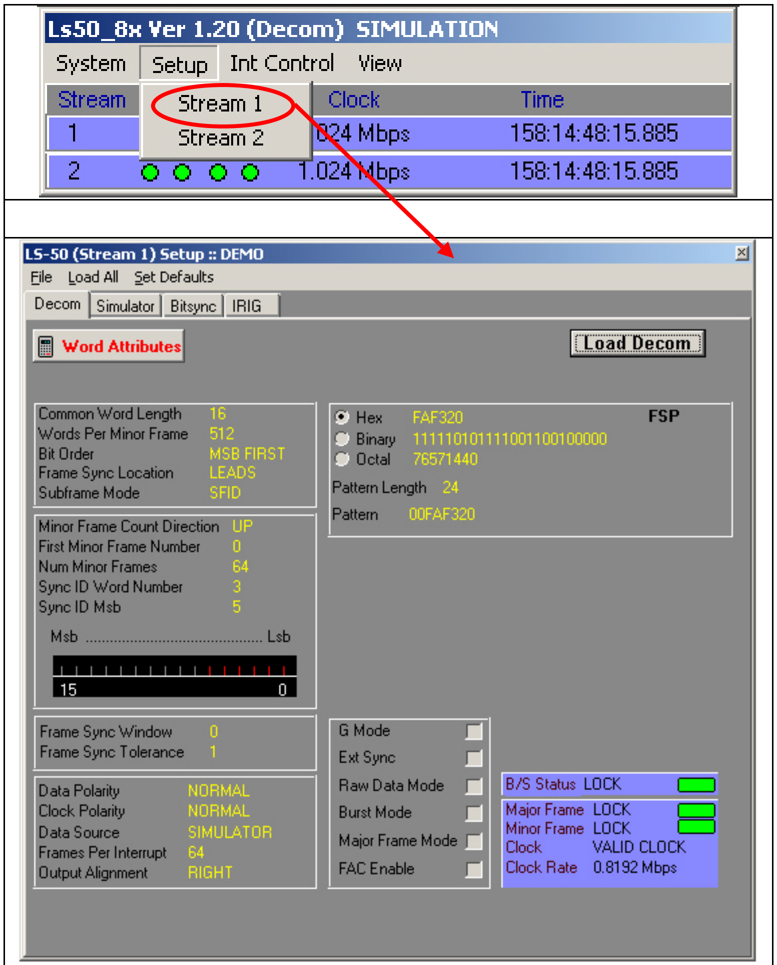
Figure 3-2 Status Display for the LS-50



Stream	Status	Clock	Time
1	● ● ● ●	0.0 Mbps	000:01:57:37.014

3.2 Configuring The LS-50-P Hardware

From the “Ls50_8x (Decom)” display[11] shown below in Figure 3-3, click *“Setup”* and then *“Stream 1” (Setup →Stream 1).*



Stream	Status	Clock	Time
1		024 Mbps	158:14:48:15.885
2	● ● ● ●	1.024 Mbps	158:14:48:15.885

Common Word Length 16
Words Per Minor Frame 512
Bit Order MSB FIRST
Frame Sync Location LEADS
Subframe Mode SFID

● Hex FAF320
○ Binary 111110101111001100100000
○ Octal 76571440
Pattern Length 24
Pattern 00FAF320

Minor Frame Count Direction UP
First Minor Frame Number 0
Num Minor Frames 64
Sync ID Word Number 3
Sync ID Msb 5

Frame Sync Window 0
Frame Sync Tolerance 1

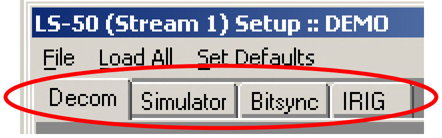
Data Polarity NORMAL
Clock Polarity NORMAL
Data Source SIMULATOR
Frames Per Interrupt 64
Output Alignment RIGHT

G Mode □
Ext Sync □
Raw Data Mode □
Burst Mode □
Major Frame Mode □
FAC Enable □

B/S Status LOCK 🟩
Major Frame LOCK 🟩
Minor Frame LOCK 🟩
Clock VALID CLOCK
Clock Rate 0.8192 Mbps

Figure 3-3 Configuration Menus/Controls for the LS-50-P

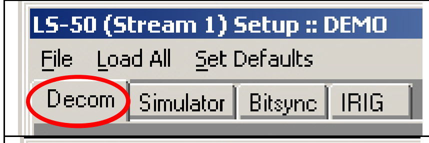
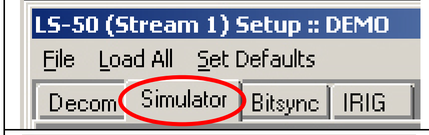
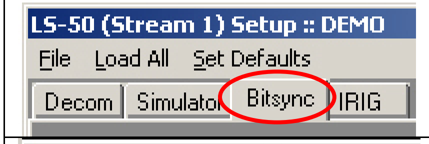
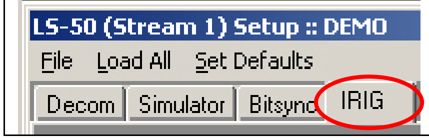
[11] This figure shows the server setup window in “Simulation” mode, where the LS-50-P hardware is not installed in the system. When actual LS-50-P hardware is installed, the server setup window appears as shown in Figure 3-14 on page 114

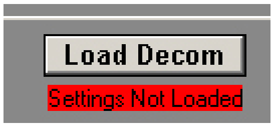


The "LS-50 (Stream 1) Setup" display shown above in Figure 3-3 is divided into several regions. Below the window header are the *"File," "Load All,"* and *"Set Defaults"* commands (more about these later). Each of the LS-50's four main functions have their own setup tab. To completely configure the LS-50-P, visit each tab in turn and configure the functions. After the setup configuration is complete, save the settings by invoking the *"File → SaveAs"* command. To download the configuration to the LS-50-P hardware, invoke the *"Load All"* command. To recall a previously defined LS-50-P setup configuration, invoke the *"File → Recall"* command and select the appropriate file from the file menu and then download the configuration to the LS-50-P hardware by invoking the *"Load All"* command.

3.2.1 Windows Launched from Decom Sseup

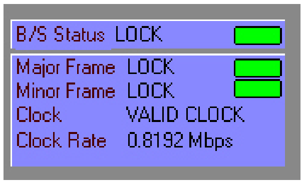
To invoke the controls for any of the tabs in the display, simply place the mouse curser in a region and right click. The resulting menus for the Decom tab are shown in Figure 3-4 on page 81 and are discussed in detail in the following paragraphs. The configuration setup for the Decommutator, Simulator, Bit Synchronizer, and IRIG Timecode functions are described in detail as indicated in the table below.

	See paragraph 3.2.1.1 on page 81 for more info on the Decommutator.
	See paragraph 3.2.1.2 on page 94 for more info on the PCM Simulator.
	See paragraph 3.2.1.3 on page 105 for more info on the Bit Synchronizer.
	See paragraph 3.2.1.4 on page 110 for more info on the IRIG Timecode Reader/Generator.



Settings Not Loaded

Each tab has a button control to load the setup information for the portion of the card displayed with the tab. Changes made with any of the controls will not take affect until this button is pressed.

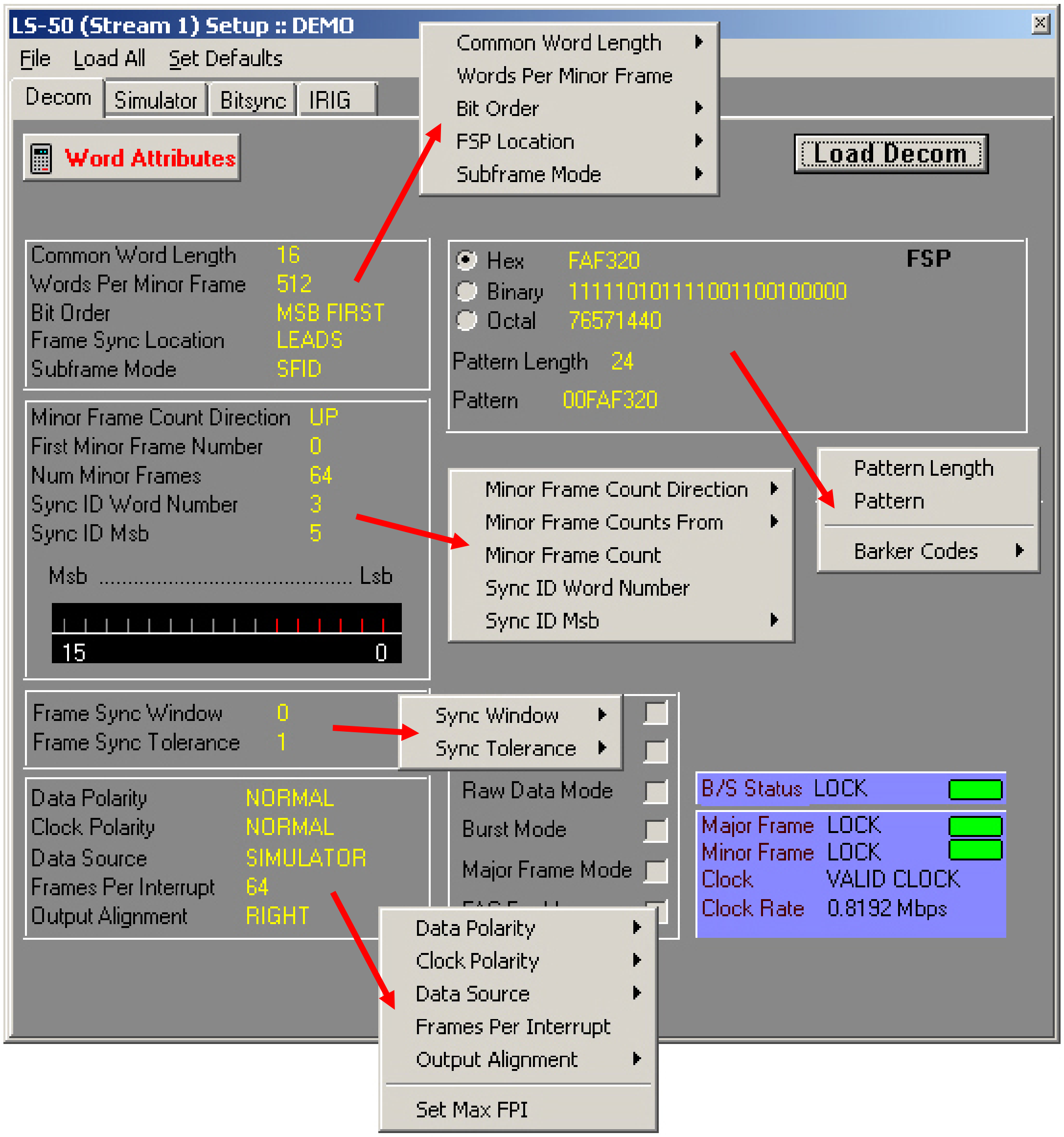


B/S Status LOCK 🟩
Major Frame LOCK 🟩
Minor Frame LOCK 🟩
Clock VALID CLOCK
Clock Rate 0.8192 Mbps

There is also a window displayed (shown left) showing the status of some of the LS-50's functional states (like frame lock). This status display is updated at a ten hertz rate. The user may load all four major functions (Decom, Simulator, Bitsync, and IRIG) from the *"Load All"* command on the menu next to the File menu). If any changes are made to an individual setup without loading, a red text will appear below the Load button (shown above right), indicating the displayed data does not match the cards' loaded data.

3.2.1.1 The LS-50-P Decommutator Tab

The LS-50-P decommutator setup tab and it's associated menus and controls are shown in Figure 3-4 below. There are up to seven groups of controls displayed for the decommutator, depending on the setting of other controls. If a project is loaded from the LDPS server (see Figure 3-1 on page 77), then some portions of the window will not be able to be controlled.



LS-50 (Stream 1) Setup :: DEMO

Common Word Length	16
Words Per Minor Frame	512
Bit Order	MSB FIRST
Frame Sync Location	LEADS
Subframe Mode	SFID

Minor Frame Count Direction	UP
First Minor Frame Number	0
Num Minor Frames	64
Sync ID Word Number	3
Sync ID Msb	5

Frame Sync Window	0
Frame Sync Tolerance	1

Data Polarity	NORMAL
Clock Polarity	NORMAL
Data Source	SIMULATOR
Frames Per Interrupt	64
Output Alignment	RIGHT

FSP

● Hex FAF320
○ Binary 111110101111001100100000
○ Octal 76571440

Pattern Length 24
Pattern 00FAF320

B/S Status	LOCK 🟩
Major Frame	LOCK 🟩
Minor Frame	LOCK 🟩
Clock	VALID CLOCK
Clock Rate	0.8192 Mbps

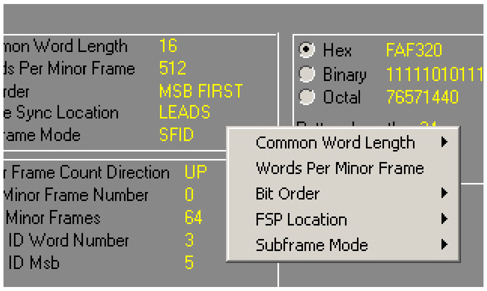
Figure 3-4 The LS-50-P Decom Tab Configuration Menus

The seven control groups of the LS-50-P decom tab include:

1. Major Frame Configuration
2. Minor Frame Configuration
3. Frame Synchronization Pattern (including optional URC)
4. Frame Sync Sensitivity Parameters
5. Data Source Configuration
6. Decommutator Modes
7. Word Attributes Control

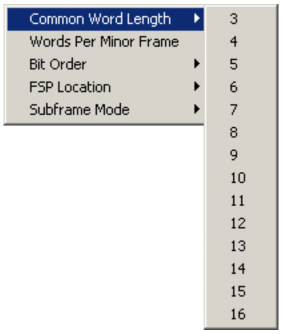
3.2.1.1.1 Major Frame Configuration

The major frame configuration consists of five controls/parameters that include: common word length, the number of words per minor frame, the bit order of the words in the frame, the frame synchronization patter location, and the subframe synchronization mode.



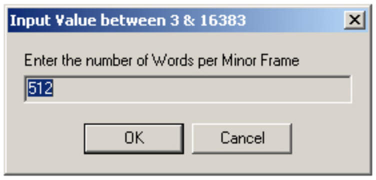
mon Word Length 16
ds Per Minor Frame 512
rder MSB FIRST
e Sync Location LEADS
ame Mode SFID
r Frame Count Direction UP
Minor Frame Number 0
Minor Frames 64
ID Word Number 3
ID Msb 5

● Hex FAF320
○ Binary 11111010111
○ Octal 76571440



The **Common Word Length** may be set from 3 to 16 bits in length. The common word length defines the length in bits of the *majority* of words that make up a minor frame. Note, not all words in a minor frame need be of the same length. For example, the majority of the words in a minor frame could be 8-bits in length, and thus the common word length would be 8. However, several of the words might be 14 or 16 bits in length and would be individually specified using the Decommutator Word Attributes command function described in paragraph 3.2.1.1.8 on page 91

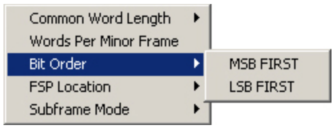
The user defines the minor frame length by invoking the **Words Per Minor Frame** command. Here, the user enters the number of words (of length specified by the word attributes) that make up a minor frame. The minor frame length on the LS-50 may be between 3 and 16,383 words.



Input Value between 3 & 16383

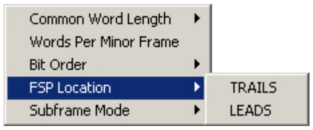
Enter the number of Words per Minor Frame

512



By invoking the **Bit Order** command, the user specifies for the common words of the minor frame whether the Most Significant Bit (MSB) is first, as read from left to right, or the Least Significant Bit (LSB) is first, again, read from left to right. Note, not all words in a minor frame need have the same bit order. For example, the majority of the words in a minor frame could have LSB first bit order. However, several of the words might be MSB first and would be individually specified using the

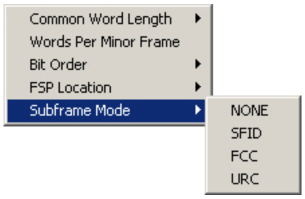
Decommutator Word Attributes command function described in paragraph 3.2.1.1.8 on page 91



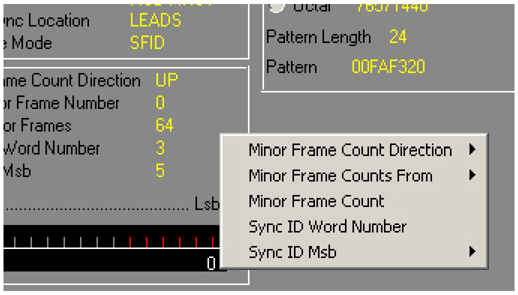
The user specifies the location of the FSP by invoking the **FSP Location** command, and selecting "TRAILS" or "LEADS."

To implement a subframe synchronization scheme, telemetry designers often add one or more "special" words to each minor frame. These special words are used by the frame synchronizer state machine to establish the location of the first minor frame in the major frame. The LS-50 supports three subframe synchronization modes: SFID, FCC, and URC.

The user specifies the method of subframe synchronization by invoking the **Subframe Mode** command, and selecting "None," "SFID," "FCC," or "URC." Note, if the user selects the "URC" subframe synchronization mode, then a second frame synchronization pattern setup area will appear on the Decom setup tab as shown in Figure 3-5 on page 85.

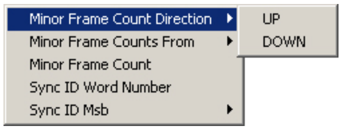


3.2.1.1.2 Minor Frame Configuration

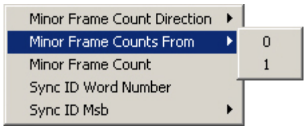

nc Location LEADS
Mode SFID
Pattern Length 24
Pattern 00FAF320
me Count Direction UP
r Frame Number 0
or Frames 64
Vord Number 3
Msb 5

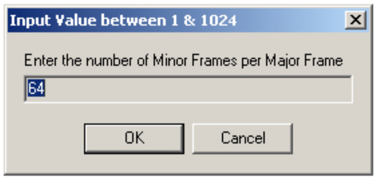
The minor frame configuration consists of five controls/parameters that include: Minor Frame Count Direction, Minor Frame Counts From, Minor Frame Count, Sync ID Word Number, and Sync ID MSB.

As mentioned previously, in the SFID mode, the synchronization pattern occupies one or more words in each minor frame and acts as a counter. The user may specify whether the pattern value increments or decrements from minor frame to minor frame by invoking the **Minor Frame Count Direction** command, and selecting "UP," or "DOWN."



In some telemetry frame designs, the subframe counter in minor frame-0 will initially begin counting from a starting value of zero (0), while in other frame designs, the subframe counter will begin counting from a starting value of one (1). The user specifies one or the other of these two conditions by invoking the **Minor Frame Counts From** command.

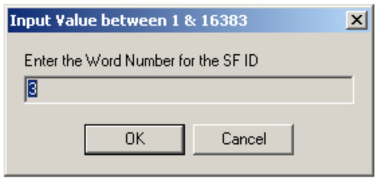



Input Value between 1 & 1024

Enter the number of Minor Frames per Major Frame

64

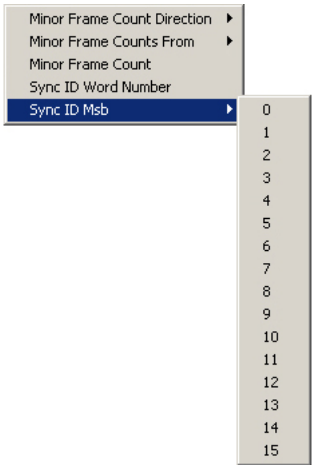
As mentioned previously, the major frame is composed of an integer number of minor frames, and the minor frame is a fixed length block of data sub-divided into an integer number of fixed-length words. By invoking the **Minor Frame Count** command, the user may specify the number of minor frames that make up the major frame. The LS-50 can support up to 1024 minor frames per major frame.


Input Value between 1 & 16383

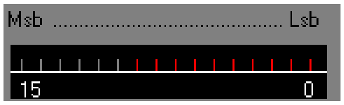
Enter the Word Number for the SF ID

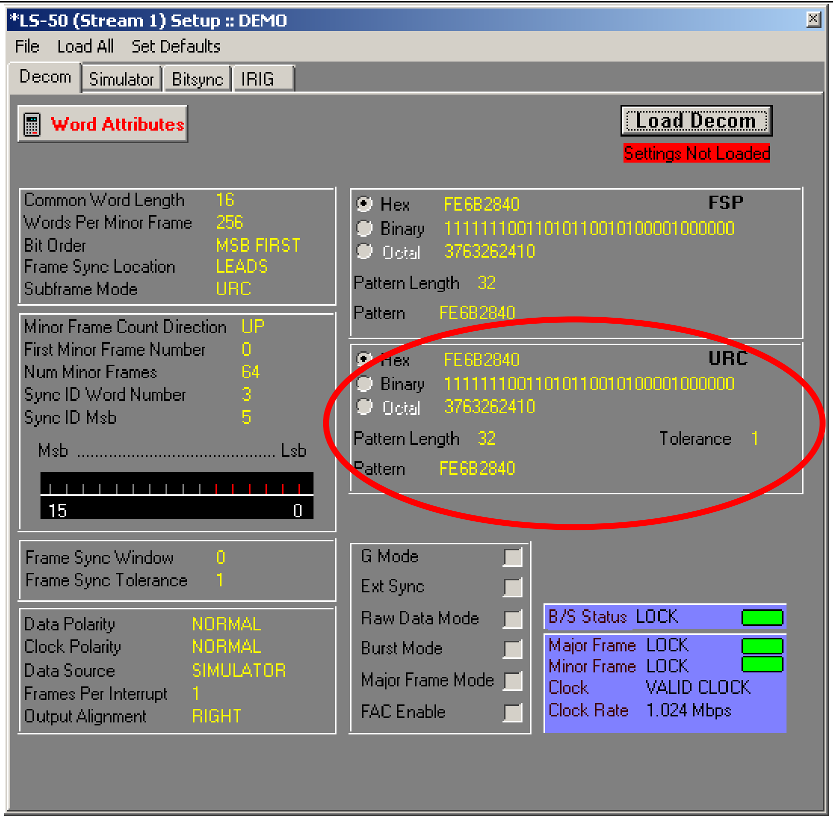
3

The location of the subframe identification (SFID) word(s) is arbitrary within the minor frame and may be specified by the user by invoking the **Sync ID Word Number** command. As the LS-50 can support up to 16,383 words per minor frame, the user may thus locate the SFID word anywhere within this range, provided it does not overlap or coincide with the frame synchronization pattern location.



As described previously, the SFID word is used as a counter, but it is not always the case that ALL of the bits in the SFID word are used for this purpose. For example, the SFID word might be 16-bits in length, but there might only be 512 minor frames in the major frame. In this scenario, a 9-bit counter ($2^9 = 512$) would be required and the user would specify the location of the counter within the larger 16-bit word by invoking the **Sync ID Msb** command and selecting the appropriate bit position for the most significant bit of the SFID counter. The Sync ID Msb is represented graphically in the minor frame configuration section as shown right.





*LS-50 (Stream 1) Setup :: DEMO

File Load All Set Defaults

Decom | Simulator | Bitsync | IRIG

Word Attributes

Load Decom

Settings Not Loaded

Common Word Length	16
Words Per Minor Frame	256
Bit Order	MSB FIRST
Frame Sync Location	LEADS
Subframe Mode	URC

Minor Frame Count Direction	UP
First Minor Frame Number	0
Num Minor Frames	64
Sync ID Word Number	3
Sync ID Msb	5

Msb .... Lsb

15 0

Frame Sync Window	0
Frame Sync Tolerance	1

Data Polarity	NORMAL
Clock Polarity	NORMAL
Data Source	SIMULATOR
Frames Per Interrupt	1
Output Alignment	RIGHT

FSP

● Hex FE6B2840
○ Binary 11111110011010110010100001000000
○ Octal 3763262410

Pattern Length 32

Pattern FE6B2840

URC

● Hex FE6B2840
○ Binary 11111110011010110010100001000000
○ Octal 3763262410

Pattern Length 32 Tolerance 1

Pattern FE6B2840

G Mode ☐
Ext Sync ☐
Raw Data Mode ☐
Burst Mode ☐
Major Frame Mode ☐
FAC Enable ☐

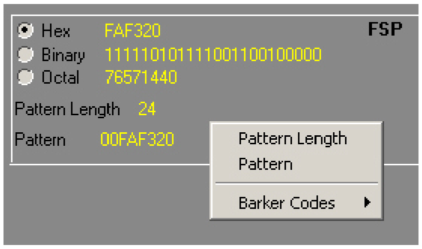
B/S Status LOCK 🟩

Major Frame LOCK 🟩
Minor Frame LOCK 🟩
Clock VALID CLOCK
Clock Rate 1.024 Mbps

Figure 3-5 Unique Recycle Code Variation of the Decom Setup Tab

3.2.1.1.3 Frame Synchronization Pattern

The frame synchronization pattern parameters include: the actual Pattern and the Pattern Length. The user may enter the actual pattern in a variety of different format representations including Hexadecimal (HEX), Binary and Octal. If the synchronization pattern is to contain "don't care bits," then the pattern must be entered in binary As mentioned previously, the frame synchronization pattern is a unique binary bit pattern used to indicate the beginning of a telemetry minor frame. To achieve this, a frame synchronizer is employed with correlate or & state machine circuitry that recognizes unique bit patterns indicating the beginning of minor frame data. The frame synchronizer typically "searches" for patterns, "checks" for the recurrence of the pattern in the same position for several frame periods, and then "locks" on the pattern.



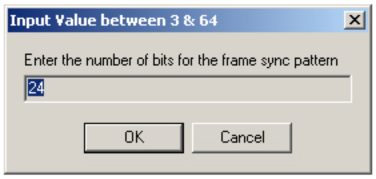
FSP

● Hex FAF320
○ Binary 111110101111001100100000
○ Octal 76571440

Pattern Length 24

Pattern 00FAF320

To enter the required frame synchronization pattern, the user must first invoke the **Pattern Length** command to specify the bit length of the frame sync pattern. For the LS-50, the length of the pattern may be up to 64-bits.

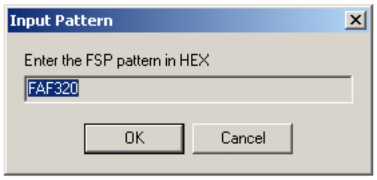


Input Value between 3 & 64

Enter the number of bits for the frame sync pattern

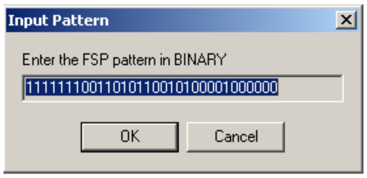
24

After entering the number of bits for the frame sync pattern, the appropriate Barker code pattern will automatically be filled in on the input pattern dialog box. This feature is based on the number of bits entered for the pattern length (only for lengths of bits 7 through 32 bits will this occur). Then the user must select one of the Hexadecimal (HEX), Binary or Octal format representation radio buttons. The selected radio button will determine the appearance of the input pattern dialog box when the **Pattern** command is invoked. Note that if the pattern length is NOT an even multiple of eight (8), then the “Octal” radio button will be grayed out. Also, if the pattern length is not an even multiple of four (4), then the “HEX” radio button will be grayed out.



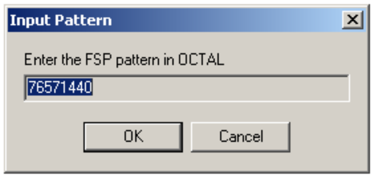
Enter the FSP pattern in HEX

FAF320



Enter the FSP pattern in BINARY

11111110011010110010100001000000



Enter the FSP pattern in OCTAL

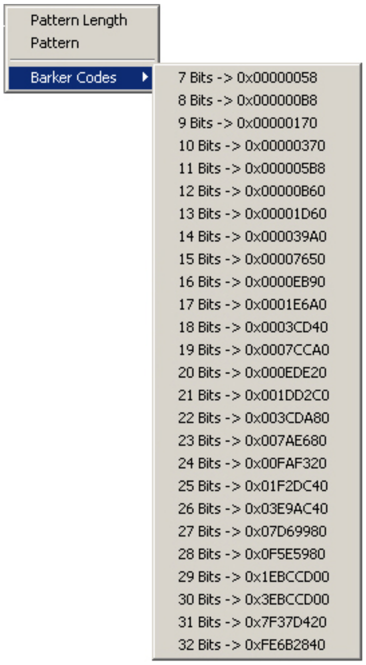
76571440

If the user wishes to use a pattern other than the one automatically selected based on the pattern length, then the pattern command should be invoked and a different pattern should be entered.

ℹ	Note – Per the IRIG-106, it is recommended that for optimal results, the frame synchronization pattern should be at least 16-bits in length[12]. *(24 or 32 bits would be much better).* In the LS-50, the pattern may be up to 64-bits in length

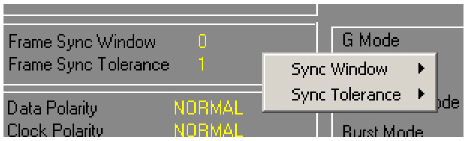
[12] J. L. Maury, Jr. and J. Styles, "Development of Optimum Frame Synchronization Codes for Goddard Space Flight Center PCM Telemetry Standards," in Proceedings of the National Telemetering Conference, June 1964.

As previously mentioned, optimal codes for the sync pattern should be chosen because they have low correlation properties unless the code pattern is exactly aligned with the desired pattern. To aid the user in selecting the appropriate pattern, invoke the **Barker Codes** command for a convenient list of some possible sync patterns. Note that choosing a pattern form the popup list does not "enter" the pattern – that still must be done via the **Pattern** command.

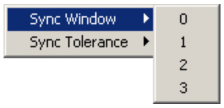


3.2.1.1.4 Frame Sync Sensitivity Parameters

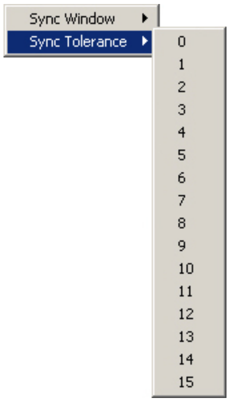
The frame synchronization sensitivity parameters include: the Sync Window and Sync Tolerance commands. Both of these commands relate to how well the frame synchronization process functions in a noisy, real world environment.



Frame Sync Window 0
Frame Sync Tolerance 1
Data Polarity NORMAL
Clock Polarity NORMAL

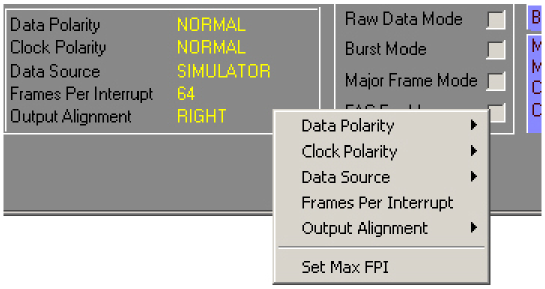


The frame synchronizer in the LS-50 typically "searches" for patterns, "checks" for the recurrence of the pattern in the *same position* for several frame periods, and then "locks" on the pattern. Because of certain peculiarities in the demodulation and bit synchronization processes for noisy channels, sometimes the recovered sync pattern may be shifted, or offset in time by one or more bit time periods. If these "bit-slips" in the recovered sync pattern are not allowed and accounted for, then the synchronization state machine will loose sync because the pattern is NOT in the exact *same position* as it was in the previous minor frame. The user specifies the number of bit-slips allowed by invoking the **Sync Window** command and entering a value of up to 3 bits. Note, in a noisy signal environment, setting the window to Zero (0) would likely result in the LS-50 NEVER acquiring or maintaining frame synchronization.



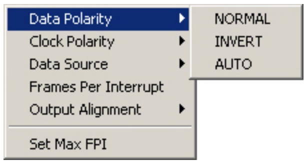
The user may specify the number of bits in the acquired sync pattern that may be different from the ideal pattern and still achieve & maintain synchronization by invoking the **Sync Tolerance** command. The user may specify that the received pattern must contain no bit errors, and would thus set the tolerance to Zero (0). In a noisy signal environment, such a setting would likely result in the LS-50 NEVER acquiring or maintaining frame synchronization. For the noisy, real world environment, the user may set the bit error tolerance from 1 to 16 bits. Some guidance on what to set the Sync Tolerance value to can be found below.

3.2.1.1.5 Data Source Configuration

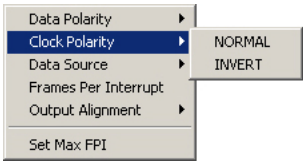


Data Polarity NORMAL
Clock Polarity NORMAL
Data Source SIMULATOR
Frames Per Interrupt 64
Output Alignment RIGHT

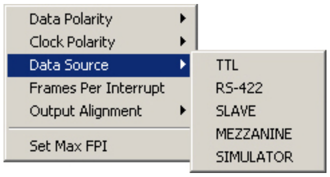
The Data Source Configuration parameters include: the Data Polarity, Clock Polarity, Data Source, Frames Per Interrupt, and Output Alignment.



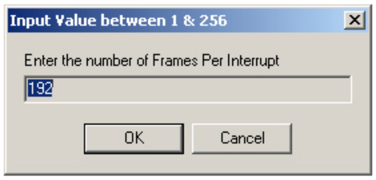
In the telemetry field, certain data transmission & demodulation schemes have inherent ambiguities that may result in the data at the decommutator input being inverted. By invoking the **Data Polarity** command, the LS-50 decommutator can be programmed by the user to accept patterns of either data polarity. The "AUTO" mode automatically inverts the incoming data if there is no frame lock and an inverted pattern is detected. This mode should probably be defaulted to unless the sync strategy is set to Frame Alternating Complement (FAC). To manually invert the incoming data, irrespective of the frame sync status, one selects the "INVERT" mode. The "NORMAL" mode leaves the polarity sense of the incoming data unchanged.



The LS-50 decommutator essentially has two basic signal input types; Clock, and Data. By using the **Clock Polarity** mode, the user may select either polarity sense of the input clock. In essence, the clock polarity mode allows the user to select either the rising or falling edge of the clock to latch incoming data into the decommutator. For the rising edge, select "NORMAL." For the falling edge, select "INVERT."



The LS-50-P decommutator has five sets of data and clock inputs, and the user may select from these by invoking the **Data Source** command. The inputs that may be selected include: TTL, RS-422, Slave, MEZZANINE, and SIMULATOR. For a single-ended clock/data input, select "TTL." For a differential clock/data input, select RS-422. For applications involving an onboard LS-40 bit synchronizer, select "MEZZANINE." For applications involving embedded asynchronous streams and an on-board LS-55-DB daughtercard decommutator, select “SLAVE.” For development and testing applications, select “SIMULATOR.” This will allow the decommutator to be driven by a known & controlled source of data.


Input Value between 1 & 256

Enter the number of Frames Per Interrupt

192

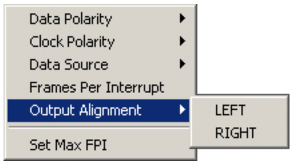
The LS-50-P decommutator can be used with extremely large frame formats (16,383 words per minor frame) and contains dual ping-pong data output buffers, each with 128K bytes of memory. The output of the decommutator is a stream of words from the input, with a header prefixed to the beginning of each minor frame. This data is grouped into “blocks” of one or more minor frames and written to the on-board buffer memory. Two such ping-pong buffers are provided. Normally while the decommutator writes to one ping-pong buffer, the other is accessible for use. When a block’s worth of data has been written, an interrupt is generated and the two buffers are logically switched so that fresh data becomes available. The user may control the number of minor frames that make up the ping-pong buffer by invoking the **Frames Per Interrupt** command. For optimal results, the user should set the frames per interrupt value to some multiple of the minor frames per major frame size. The LS-50 can support up to 256 frames per interrupt, depending on the frame size. Note, for fast streams, the user should maximize the number of frames per interrupt to reduce the load on the CPU. If the user is unsure what to set the frames per interrupt value to, the **Set Max FPI** (Frames Per Interrupt) command may be invoked to set the maximum number of frames per interrupt based on the minor frame size and the amount of memory on the card.

Note: The number of minor frames per interrupt cannot exceed:

a. 256

b. ((words-per-minor frame + 5) * 2 * frames per interrupt) cannot exceed 131,072 bytes.

c. (words-per-minor frame + 5) cannot exceed 16,383 words.



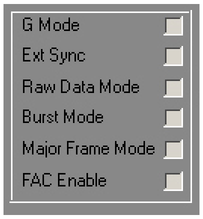
To select left-justified or right-justified output data from the decommutator, the user may invoke the **Output Alignment** command. Note: The output alignment should always be set to Right Aligned, with the possible exception of connecting the LS-50 to a LS-71 DAC[13]. In general, if left alignment is selected, then the processing overhead of LDPS will be increased, because part of the 'normalization' process involves the right alignment of all the data prior to sending it off to the client or processing tasks such as audio or video, etc.

[13] If the bits-per-word is greater than 14, then left alignment may come into play as a possible requirement, depending on what resolution the DAC output is using. If the data is right aligned and bits-per-word is 16, then the two LSBs on the DAC output will be lost. If the data is left aligned, then the two MSBs on the DAC output will be lost.

Recommendation:
A good rule of thumb - If the minor frame rate is 50 Hz or less, then set the FPI to 1. If it is more, then set it to the number of minor frames per major frame, if it will fit. Otherwise the user will have to experiment with FPI numbers between 1 and the minors-per-major (ideally a multiple of minors-per-major).

3.2.1.1.6 Decom Mode Check Boxes

The LS-50 decommutator setup tab has a number of mode selection check boxes that include: G Mode, External Sync, Raw Data Mode, Burst Mode, Major Frame Mode, and FAC Enable.



G Mode □
Ext Sync □
Raw Data Mode □
Burst Mode □
Major Frame Mode □
FAC Enable □

Normally the decommutator output stops when it loses minor frame lock. If **G Mode** is checked, the decommutator will continue to processes incoming bits into “frames” and output them. If it detects a sync pattern while in this state, it will abort the frame it is assembling, and start a new buffer. Basically, the G Mode tells the decom to try to lock onto the frame sync pattern, but even if it cannot, it collects the buffer of data and generates an interrupt even if there is no frame lock.

To support fixed length frames that arrive at irregular intervals, the user may check the **Burst Mode** box. Select this if the incoming data consists of fixed-length frames separated by zero or more fill bits. The data in the frames will be output and the fill bits will be discarded.

The **Ext Sync** mode instructs the decommutator to establish the “lock” condition based upon an external sync pulse signal only. This mode bypasses the internal frame synchronizer (correlator/state machine) in favor of an external signal provided by the user.

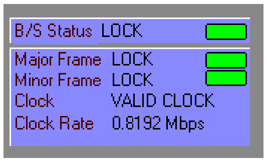
The **Raw Data Mode** instructs the decommutator to ignore the frame lock state (i.e., don’t look for a frame sync pattern) and just ingest the correct number of bits and generate an interrupt. This mode is used to record 100% of the input bits, regardless of lock state.

When selected, the **Major Frame Mode** will generate an interrupt only when a complete major frame of data has been gathered and the decom is in major and minor frame lock. Note, in this mode the frames per interrupt is fixed to the number of minor frames.

The **FAC Enable** mode is used to enable the Frame Alternating Complement subframe synchronization method. As discussed above, the FAC mode is a variant of the FCC subframe synchronization method.

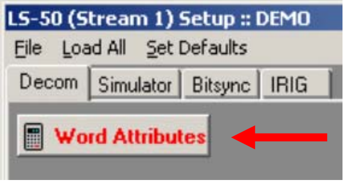
3.2.1.1.7 Decom Status Displays

The LS-50-P decom setup tab has a window display showing the status of some of the LS-50's functional states. These states include: bit synchronizer signal lock, major and minor frame lock, a valid clock indication, as well as the clock rate in Mbps. This status display is updated at a ten-hertz rate and is common to all LS-50-P function setup tabs.

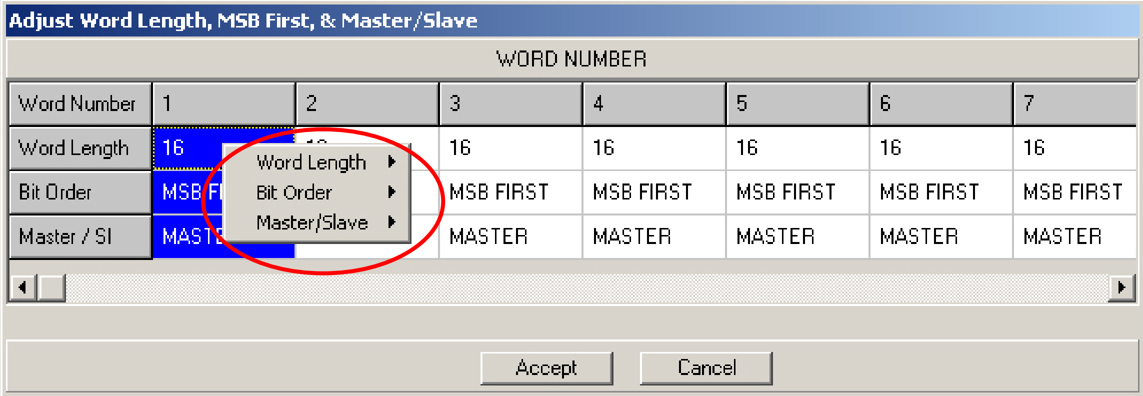


B/S Status	LOCK	🟩
Major Frame	LOCK	🟩
Minor Frame	LOCK	🟩
Clock	VALID CLOCK	
Clock Rate	0.8192 Mbps	

3.2.1.1.8 Decommutator Word Attributes



The **Word Attributes** button directly below the Decom tab allows the user to make individual exceptions to the definitions established in the Major Frame Configuration section of the Decom tab (see paragraph 3.2.1.1.1 on page 82). The word attributes include: word length, bit order, and master/slave status. The word attributes dialog box is shown in Figure 3-6 below. To modify the word attributes of a particular word in the minor frame, navigate using the scroll bar at the bottom of the window and select a word by clicking on the middle of the column. Right clicking will invoke the attributes menu as shown in the figure below (red oval). To select a contiguous group of words, select the first word, then shift-click on the last word to select the group. To select a noncontiguous set of words, select the first word, and then control-click on each subsequent word until all words are selected. After the words are selected, right click to invoke the attributes menu.

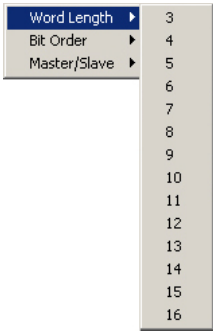


Adjust Word Length, MSB First, & Master/Slave

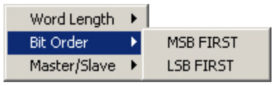
WORD NUMBER

Word Number	1	2	3	4	5	6	7
Word Length	16		16	16	16	16	16
Bit Order	MSB FI		MSB FIRST	MSB FIRST	MSB FIRST	MSB FIRST	MSB FIRST
Master / Sl	MASTE		MASTER	MASTER	MASTER	MASTER	MASTER

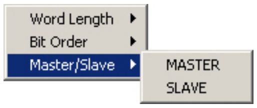
Figure 3-6 LS-50 Decom – Word Attributes Setup



The **Word Length** command may be used to set the length of selected words from 3 to 16 bits in length. The user might invoke this command because not all words in a minor frame need be of the same length. For example, the common words in a minor frame could be 8-bits in length. However, several of the words might be 14 or 16 bits in length and would be individually specified using this command.

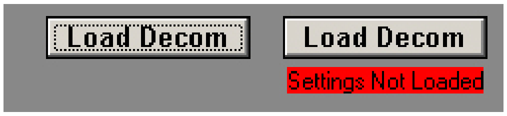


By invoking the **Bit Order** command, the user specifies for the selected words of the minor frame whether the Most Significant Bit (MSB) is first, as read from left to right, or the Least Significant Bit (LSB) is first, again, read from left to right. The user might invoke this command because not all words in a minor frame need have the same bit order. For example, the common words in a minor frame could have LSB first bit order. However, several of the words might be MSB first and would be individually specified using this command.



For telemetry formats that involve embedded asynchronous frames, and the use of a second hardware decommutator such as the Lumistar LS-55-DB, the user may specify the location of the embedded words by invoking the **Master/Slave** command and selecting the "SLAVE" mode. Thus selected, whenever any of the embedded words are encountered by the decom, they are serially redirected out of the decom via the "slave port." The slave port is a serial output (clock & data) that drives an LS-55-DB, or external decommutator. The embedded words may be prime commutated, or super-commutated within the minor frame. The default mode for all common words in the minor frame is "MASTER."

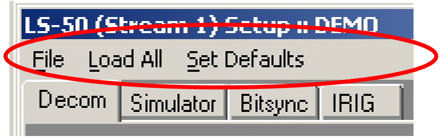
3.2.1.1.9 Load Decom Button



Settings Not Loaded

The Decom setup tab has a button control to load the setup information entered by the user. Changes made with any of the controls will not take affect until this button is pressed. The user may load all four major functions (Decom, Simulator, Bitsync, and IRIG) from the "Load All" command on the menu next to the File menu). If any changes are made to the decom setup without loading, a red text will appear below the Load button (shown above right), indicating the displayed data does not match the cards' loaded data.

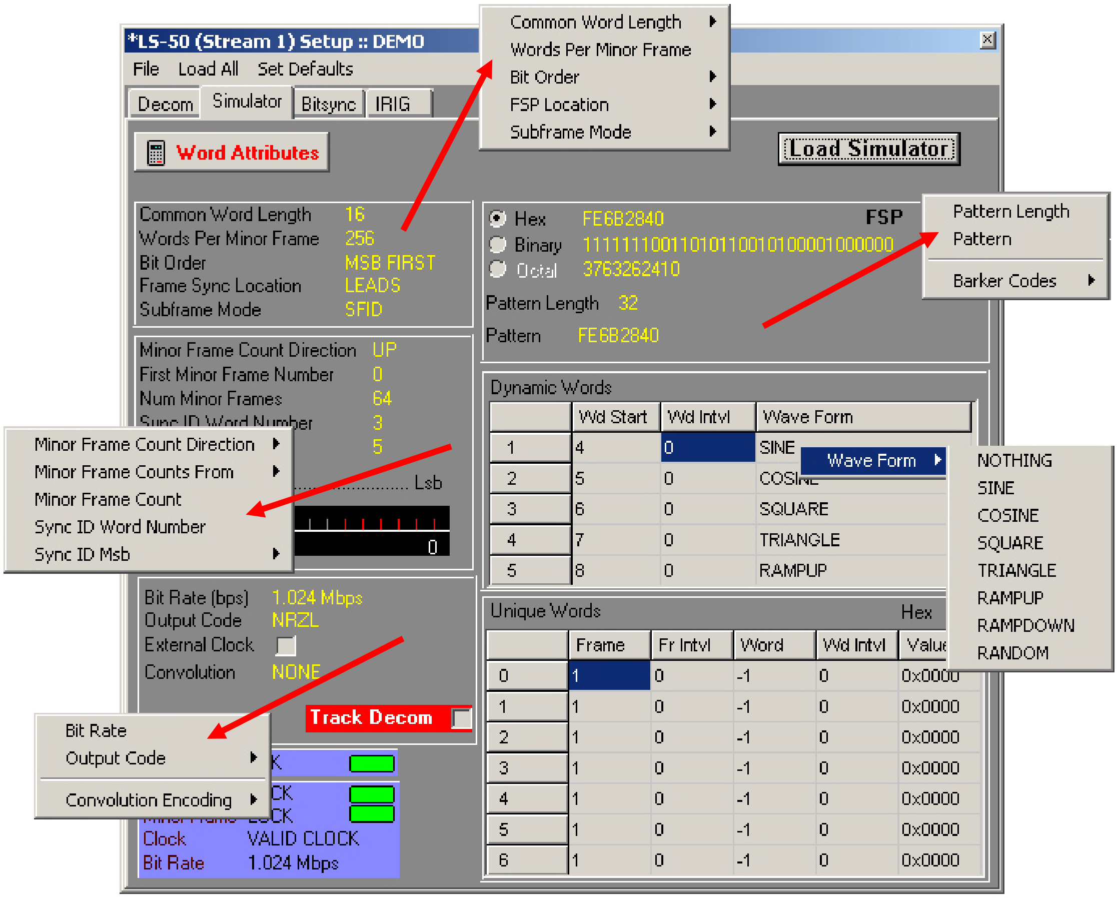
3.2.1.1.10 Saving the Decommutator Setup Configuration



Below the window header of the "LS-50 (Stream 1) Setup" display shown in Figure 3-3 on page 79 are the *"File," "Load All,"* and *"Set Defaults"* commands. After the decom setup configuration is complete, save the settings by invoking the *"File → SaveAs"* command. To download all of the configurations (decom, simulator, Bitsync, and IRIG) to the LS-50-P hardware, invoke the *"Load All"* command. To recall a previously defined LS-50-P setup configuration, invoke the *"File → Recall"* command and select the appropriate file from the file menu and then download the configuration to the LS-50-P hardware by invoking the *"Load All"* command. To set the LS-50-P hardware to its default state, invoke the *"Set Defaults"* command.

3.2.1.2 The LS-50-P Simulator Tab

The LS-50-P simulator setup tab and it's associated menus and controls are shown in Figure 3-7 below. The LS-50-P simulator may be used to drive the decommutator in a self-test or frame definition scenario, or it may be used independently to create PCM data streams not intended for the on-board decommutator.



Common Word Length 16
Words Per Minor Frame 256
Bit Order MSB FIRST
Frame Sync Location LEADS
Subframe Mode SFID

Minor Frame Count Direction UP
First Minor Frame Number 0
Num Minor Frames 64
Sync ID Word Number 3
5

Bit Rate (bps) 1.024 Mbps
Output Code NRZL
External Clock □
Convolution NONE

Track Decom □

Clock VALID CLOCK
Bit Rate 1.024 Mbps

● Hex FE6B2840 FSP
○ Binary 11111110011010110010100001000000
○ Octal 3763262410

Pattern Length 32
Pattern FE6B2840

Dynamic Words

	Wd Start	Wd Intvl	Wave Form
1	4	0	SINE
2	5	0	COSINE
3	6	0	SQUARE
4	7	0	TRIANGLE
5	8	0	RAMPUP

Unique Words Hex

	Frame	Fr Intvl	Word	Wd Intvl	Value
0	1	0	-1	0	0x0000
1	1	0	-1	0	0x0000
2	1	0	-1	0	0x0000
3	1	0	-1	0	0x0000
4	1	0	-1	0	0x0000
5	1	0	-1	0	0x0000
6	1	0	-1	0	0x0000

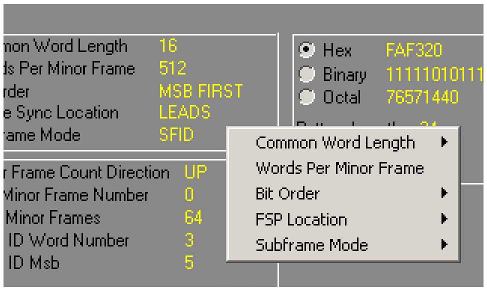
Figure 3-7 The LS-50-P Simulator Configuration Menus

There are up to eight groups of controls displayed for the simulator, depending on the setting of other controls.

The eight control groups of the LS-50-P simulator tab include:

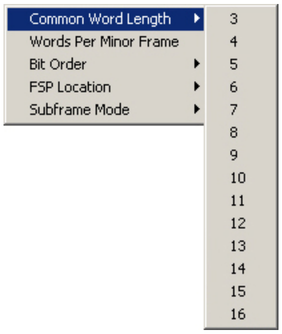
- Major Frame Configuration
- Minor Frame Configuration
- Frame Synchronization Pattern
- Clock & Output Coding Configuration
- Data Source Configuration
- Dynamic Word Configuration
- Unique Words Configuration
- Word Attributes Control

3.2.1.2.1 Major Frame Configuration

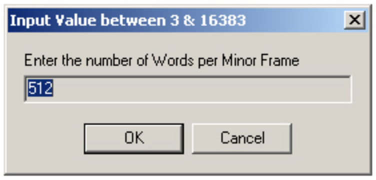


non Word Length 16
ds Per Minor Frame 512
rder MSB FIRST
e Sync Location LEADS
ame Mode SFID
r Frame Count Direction UP
Minor Frame Number 0
Minor Frames 64
ID Word Number 3
ID Msb 5
● Hex FAF320
○ Binary 11111010111
○ Octal 76571440

The major frame configuration consists of five controls/parameters that include: common word length, the number of words per minor frame, the bit order of the words in the frame, the frame synchronization patter location, and the subframe synchronization mode.



The **Common Word Length** may be set from 3 to 16 bits in length. The common word length defines the length in bits of the *majority* of words that make up a minor frame. Note, not all words in a minor frame need be of the same length. For example, the majority of the words in a minor frame could be 8-bits in length, and thus the common word length would be 8. However, several of the words might be 14 or 16 bits in length and would be individually specified using the Simulator Word Attributes command function described in paragraph 3.2.1.2.9 on page 102.

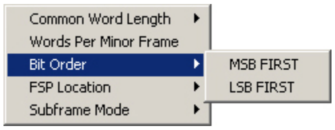


Input Value between 3 & 16383

Enter the number of Words per Minor Frame

512

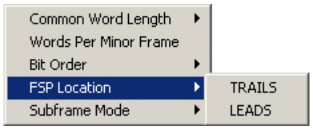
The user defines the minor frame length by invoking the **Words Per Minor Frame** command. Here, the user enters the number of words (of length specified by common word length) that make up a minor frame. The minor frame length on the LS-50 may be between 3 and 16,383 words.



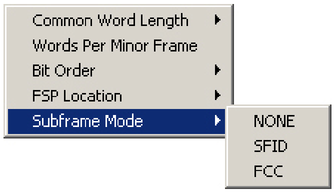
By invoking the **Bit Order** command, the user specifies for the common words of the minor frame whether the Most Significant Bit (MSB) is first, as read from left to right, or the Least Significant Bit (LSB) is first, again, read from left to right. Note, not all words in a minor frame need have the same bit order. For example, the majority of the words in a minor frame could have LSB first bit order. However, several of the words might be MSB first and would be individually specified using the

Simulator Word Attributes command function described in paragraph 3.2.1.2.9 on page 102.

The user specifies the location of the FSP by invoking the **FSP Location** command, and selecting “TRAILS” or “LEADS.”



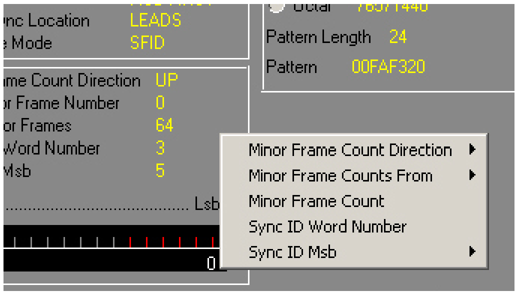
To implement a subframe synchronization scheme, telemetry designers often add one or more “special” words to each minor frame. The frame synchronizer state machine uses the special words to establish the location of the first minor frame in the major frame. The LS-50 supports three subframe synchronization modes: SFID, FCC, and URC.



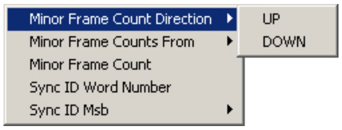
The user specifies the method of subframe synchronization by invoking the **Subframe Mode** command, and selecting “None,” “SFID,” or “FCC.”

3.2.1.2.2 Minor Frame Configuration

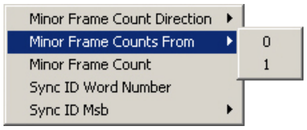
The minor frame configuration consists of five controls/parameters that include: Minor Frame Count Direction, Minor Frame Counts From, Minor Frame Count, Sync ID Word Number, and Sync ID MSB.

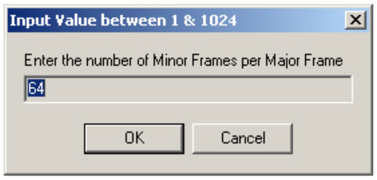


As mentioned previously, in the SFID mode, the synchronization pattern occupies one or more words in each minor frame and acts as a counter. The user may specify whether the pattern value increments or decrements from minor frame to minor frame by invoking the **Minor Frame Count Direction** command.



In some telemetry frame designs, the subframe counter in minor frame-0 will initially begin counting from a starting value of zero (0), while in other frame designs, the subframe counter will begin counting from a starting value of one (1). The user specifies one or the other of these two conditions by invoking the **Minor Frame Counts From** command.

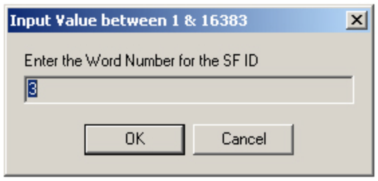



Input Value between 1 & 1024

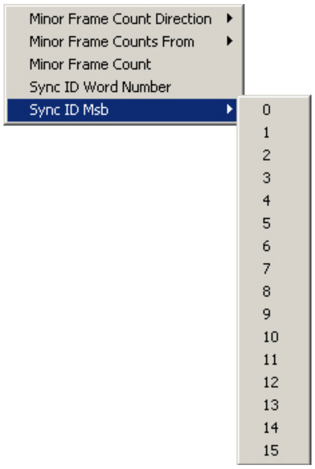
Enter the number of Minor Frames per Major Frame: 64

As mentioned previously, the major frame is composed of an integer number of minor frames, and the minor frame is a fixed length block of data sub-divided into an integer number of fixed-length words. By invoking the **Minor Frame Count** command, the user may specify the number of minor frames that make up the major frame. The LS-50 can support up to 1024 minor frames per major frame.

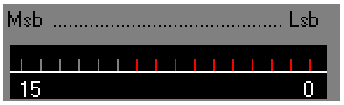
The location of the subframe identification (SFID) word(s) is arbitrary within the minor frame and may be specified by the user by invoking the **Sync ID Word Number** command. As the LS-50 can support up to 16,383 words per minor frame, the user may thus locate the SFID word anywhere within this range, provided it does not overlap or coincide with the frame synchronization pattern location.


Input Value between 1 & 16383

Enter the Word Number for the SF ID: 3

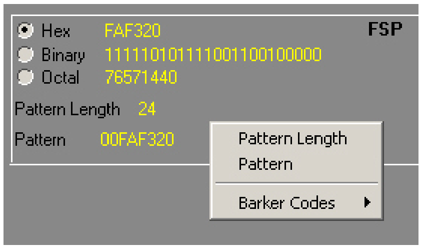


As described previously, the SFID word is used as a counter, but it is not always the case that ALL of the bits in the SFID word are used for this purpose. For example, the SFID word might be 16-bits in length, but there might only be 512 minor frames in the major frame. In this scenario, a 9-bit counter ($2^9 = 512$) would be required and the user would specify the location of the counter within the larger 16-bit word by invoking the **Sync ID Msb** command and selecting the appropriate bit position for the most significant bit of the SFID counter. The Sync ID Msb is represented graphically in the minor frame configuration section as shown right.



3.2.1.2.3 Frame Synchronization Pattern

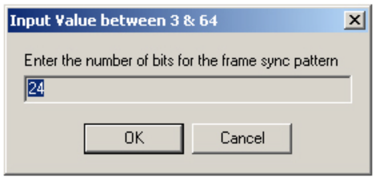
The frame synchronization pattern parameters include: the actual Pattern and the Pattern Length. The user may enter the actual pattern in a variety of different format representations including Hexadecimal (HEX), Binary and Octal. If the synchronization pattern is to contain "don't care bits," then the pattern must be entered in binary As mentioned previously, the frame synchronization pattern is a unique binary bit pattern used to indicate the beginning of a telemetry minor frame. To achieve this, a frame synchronizer is employed with correlator & state machine circuitry that recognizes unique bit patterns indicating the beginning of minor frame data. The frame synchronizer typically "searches" for patterns, "checks" for the recurrence of the pattern in the same position for several frame periods, and then "locks" on the pattern.


FSP

● Hex FAF320
○ Binary 111110101111001100100000
○ Octal 76571440

Pattern Length 24

Pattern 00FAF320

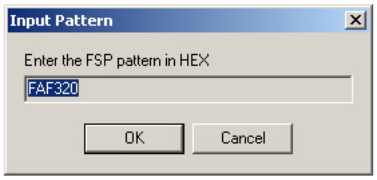


Input Value between 3 & 64

Enter the number of bits for the frame sync pattern

24

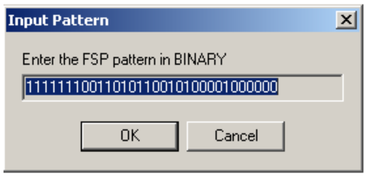
To enter the required frame synchronization pattern, the user must first invoke the **Pattern Length** command to specify the bit length of the frame sync pattern. For the LS-50, the length of the pattern may be up to 64-bits. After entering the number of bits for the frame sync pattern, the appropriate Barker code pattern will automatically be filled in on the input pattern dialog box, based on the number of bits (only for number of bits 7 through 32 will this occur). Then the user must select one of the Hexadecimal (HEX), Binary or Octal format representation radio buttons. The selected radio button will determine the appearance of the input pattern dialog box when the **Pattern** command is invoked. Note that if the pattern length is NOT an even multiple of eight (8), then the "Octal" radio button will be grayed out.



Input Pattern

Enter the FSP pattern in HEX

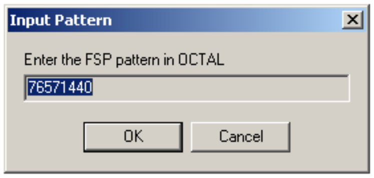
FAF320



Input Pattern

Enter the FSP pattern in BINARY

11111110011010110010100001000000



Input Pattern

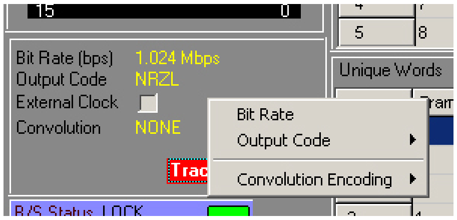
Enter the FSP pattern in OCTAL

76571440

If the user wishes to use a pattern other than the one automatically selected based on the pattern length, then the Pattern command should be invoked and a different pattern should be entered.

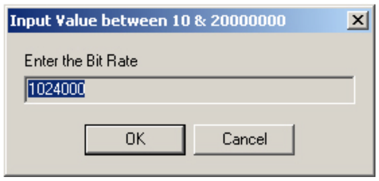
As previously mentioned, optimal codes for the sync pattern should be chosen because they have low correlation properties unless the code pattern is exactly aligned with the desired pattern. To aid the user in selecting the appropriate pattern, invoke the **Barker Codes** command for a convenient list of some possible sync patterns. Note that choosing a pattern form the popup list does not "enter" the pattern – that still must be done via the **Pattern** command.

3.2.1.2.4 Clock & Data Output Mode Configuration



Bit Rate (bps) 1.024 Mbps
Output Code NRZL
External Clock □
Convolution NONE

The Clock and Data Output Mode controls include: the output bit rate (bits/second), the output encoding format, and the Forward Error Correction (FEC) coding mode. Invoking **Bit Rate** allows the user to specify the output bit rate (bits/second) of the PCM encoder on the simulator. The user may enter a value between 10 bps to 20 Mbps for NRZ codes, and 10 bps to 10 Mbps for all other codes. By invoking the **Output Code** command, the user may select from a variety of possible PCM output codes, some of which are shown graphically in Figure 3-8 on page 99. The PCM output codes fall into several general classes including: Non-Return to Zero (NRZ) codes, self-clocking codes such as Bi-Phase, and Miller, and Randomized codes. NRZ codes are the most commonly used but are occasionally



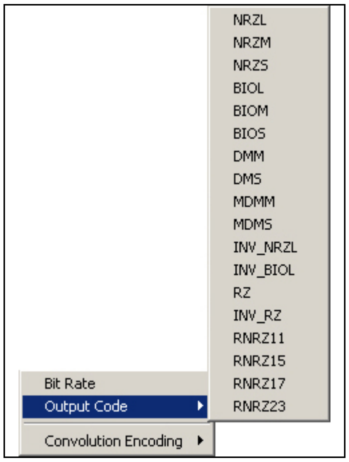
Input Value between 10 & 20000000

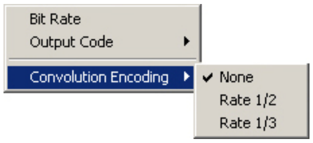
Enter the Bit Rate

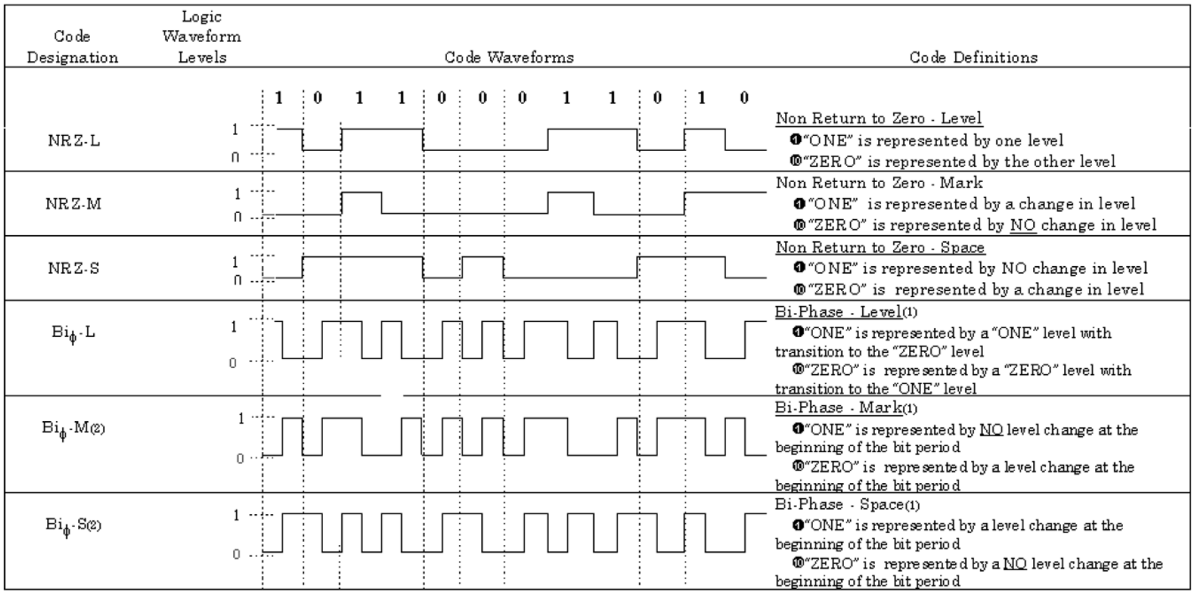
1024000

problematic if they are not well behaved[14]. Ill-behaved data streams may be mitigated by using a self-clocking code such as Bi-Phase or and Miller, but with the added penalty of doubling the required channel bandwidth. Randomized codes do not require twice the bandwidth to transmit, but in a worst-case scenario, their use can triple the received bit-error-rate.

Using Forward Error Correction schemes such as **Convolution Encoding** of the data often alleviates bit-error-rate issues in telemetry systems. A convolutional code is a type of error-correcting code often used to improve the performance of a radio or satellite link. The LS-50-P can support rate ½, and rate ⅓ convolutional codes as well as non-FEC encoded data. In general, if a convolutional code is said to be rate ½, this means that for every input data bit, the encoder will produce two output code symbol bits. For rate ⅓, every input data bit will produce three output code symbol bits. Thus, employing this type of FEC scheme in a telemetry system will double or triple the transmitted channel data rate. (*There is no free lunch in telemetry engineering!*)







Non Return to Zero - Level
❶"ONE" is represented by one level
⓪"ZERO" is represented by the other level

Non Return to Zero - Mark
❶"ONE" is represented by a change in level
⓪"ZERO" is represented by NO change in level

Non Return to Zero - Space
❶"ONE" is represented by NO change in level
⓪"ZERO" is represented by a change in level

Bi-Phase - Level(1)
❶"ONE" is represented by a "ONE" level with transition to the "ZERO" level
⓪"ZERO" is represented by a "ZERO" level with transition to the "ONE" level

Bi-Phase - Mark(1)
❶"ONE" is represented by NO level change at the beginning of the bit period
⓪"ZERO" is represented by a level change at the beginning of the bit period

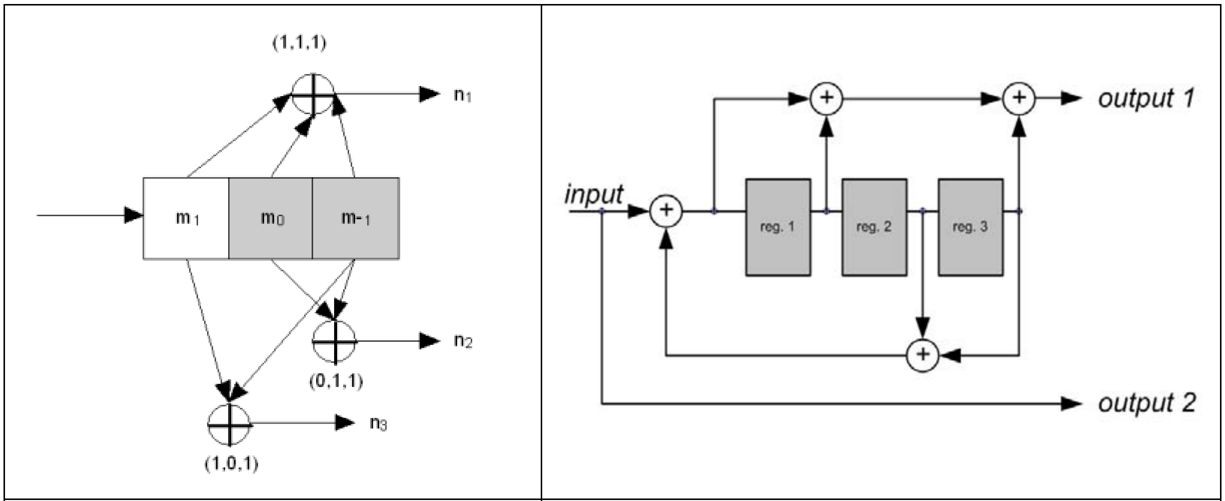
Bi-Phase - Space(1)
❶"ONE" is represented by a level change at the beginning of the bit period
⓪"ZERO" is represented by a NO level change at the beginning of the bit period

Figure 3-8 PCM Code Definitions

Several algorithms exist for decoding convolutional codes. For relatively small constraint length values, the Viterbi algorithm is universally used as it provides maximum likelihood performance and is highly parallelizable. Viterbi decoders are thus easy to implement in VLSI or FPGA hardware. An especially popular Viterbi-decoded

[14] An NRZ data stream is said to be "ill-behaved" if its spectrum has strong DC components caused by long strings of ones or zeros. Bit synchronizers have great difficulty locking onto ill-behaved signals.

convolutional code, used on the Voyager program has a constraint length of 7 and a rate of ½.

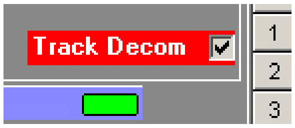


Rate 1/3 non-recursive, non-systematic convolutional encoder with constraint length 3

Rate 1/2 recursive, systematic convolutional encoder with constraint length 4

Figure 3-9 Some Examples of Convolutional Encoder Circuits

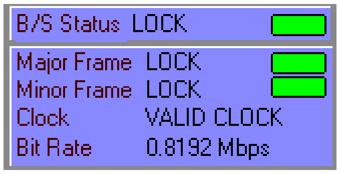
3.2.1.2.5 Linking the Simulator and Decommutator Configurations



The LS-50-P simulator may be used to drive the decommutator in a self-test or frame definition scenario, or it may be used independently to create PCM data streams not intended for the on-board decommutator. When they are used together, the user may click the **Track Decom** checkbox. This convenience will link the major and minor frame configurations entered for the decommutator with the simulator. When unchecked, the major and minor frame configurations of the simulator may be entered independently of the decommutator.

3.2.1.2.6 Status Displays

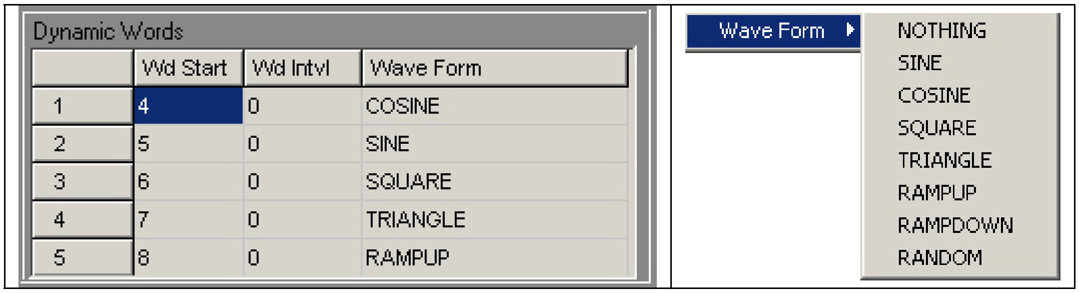
The LS-50-P Simulator setup tab has a window display showing the status of some of the LS-50's functional states. These states include: bit synchronizer signal lock, major and minor frame lock, a valid clock indication, as well as the clock rate in Mbps. This status display is updated at a ten-hertz rate and is common to all LS-50-P function setup tabs.



B/S Status	LOCK	🟩
Major Frame	LOCK	🟩
Minor Frame	LOCK	🟩
Clock	VALID CLOCK	
Bit Rate	0.8192 Mbps	

3.2.1.2.7 Dynamic Words Setup

The PCM simulator in the LS-50-P may be programmed to generate dynamic data for up to five (5) words in every minor frame at the same location. For each **Dynamic Word**, the user may select from one of seven mathematical functions as shown below right.



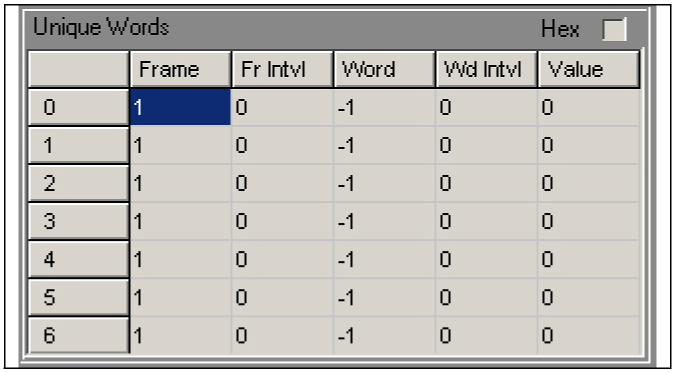
Dynamic Words

	Wd Start	Wd Intvl	Wave Form
1	4	0	COSINE
2	5	0	SINE
3	6	0	SQUARE
4	7	0	TRIANGLE
5	8	0	RAMPUP

To configure a Dynamic Word, highlight the value in the “Wd Start” cell and enter the word number. To disable a dynamic word, set the “Wd Start” cell value to “–1.” Commutation of the dynamic word is set via the “Wd Intvl” cell value. If the dynamic word is to be Prime commutated, then set the “Wd Intvl” cell value to Zero. If the dynamic word is to be supercommutated, then set the “Wd Intvl” cell value to the required increment value. To define the mathematical function that will determine the value of the dynamic word, place the cursor in the “Wave Form” cell and right-click to review the menu of functions shown above right. Select the function from the list.

3.2.1.2.8 Unique Words Setup

The PCM simulator in the LS-50-P may be programmed to generate static data for up to seven (7) words in every minor frame at the same location(s). For each **Unique Word**, the user may select the minor frame number, the frame interval, the word number within the minor frame, the word interval, and finally the word value. To disable a unique word, set the “Frame” cell value to “–1” and the “Word” cell value to “–1.” To display the word value in Hexadecimal, click the “Hex” checkbox in the upper right of the Unique Words display.

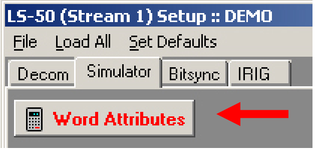


Unique Words Hex □

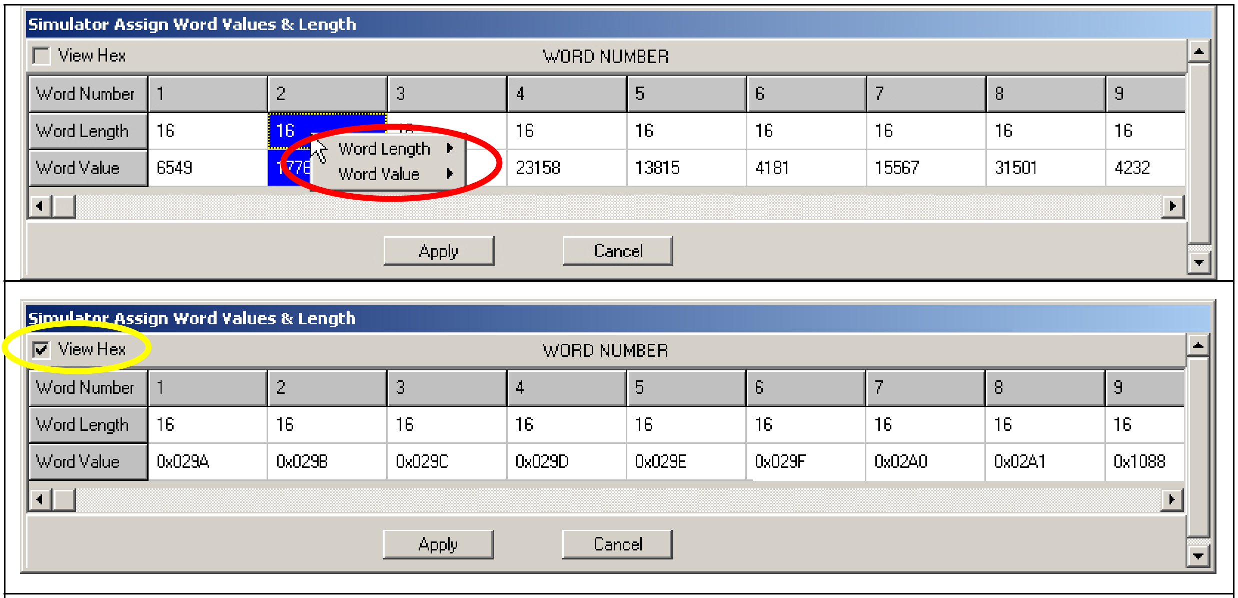
	Frame	Fr Intvl	Word	Wd Intvl	Value
0	1	0	-1	0	0
1	1	0	-1	0	0
2	1	0	-1	0	0
3	1	0	-1	0	0
4	1	0	-1	0	0
5	1	0	-1	0	0
6	1	0	-1	0	0

A wide assortment of word commutation is possible using the minor frame, frame interval, word number, and word interval values. Prime, super-commutated, subcommutated, super-subcommutated, etc. are possible. Note: in general for both Dynamic and Unique words, they cannot be the same as the frame sync pattern or the SFID word.

3.2.1.2.9 Simulator Word Attributes



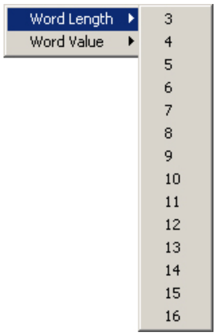
The **Word Attributes** button directly below the Simulator tab allows the user to make individual exceptions to the definitions established in the Major Frame Configuration section of the Simulator tab (see paragraph 3.2.1.2.1 on page 95). The word attributes include: word length, and word value. The word attributes dialog box is shown in Figure 3-10 below. To modify the word attributes of a particular word in the minor frame, navigate using the scroll bar at the bottom of the window and select a word by clicking on the middle of the column. Right clicking will invoke the attributes menu as shown in the figure below (red oval). To display the word values in Hex format, click the "View Hex" check box as shown below (yellow oval). To select a contiguous group of words, select the first word, then shift-click on the last word to select the group. To select a noncontiguous set of words, select the first word, and then control-click on each subsequent word until all words are selected. After the words are selected, right click to invoke the attributes menu.



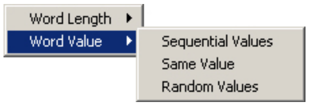
Word Number	1	2	3	4	5	6	7	8	9
Word Length	16	16		16	16	16	16	16	16
Word Value	6549	177		23158	13815	4181	15567	31501	4232

Word Number	1	2	3	4	5	6	7	8	9
Word Length	16	16	16	16	16	16	16	16	16
Word Value	0x029A	0x029B	0x029C	0x029D	0x029E	0x029F	0x02A0	0x02A1	0x1088

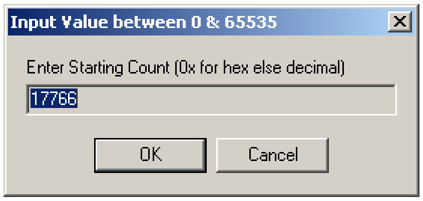
Figure 3-10 LS-50 Simulator – Word Attributes Setup



The **Word Length** command may be used to set the length of selected words from 3 to 16 bits in length. The user might invoke this command because not all words in a minor frame need be of the same length. For example, the common words in a minor frame could be 8-bits in length. However, several of the words might be 14 or 16 bits in length and would be individually specified using this command.

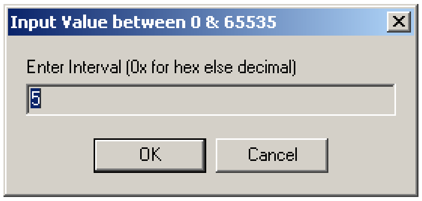


The **Word Value** command may be used to set the numerical value of individual words, or groups of words in either decimal or hexadecimal format. By invoking *"Sequential Values,"* the user may specify an initial value and in increment value for a sequences of words. The word sequence may be contiguous or irregular. To select a contiguous group of words, select the first word, then shift-click on the last word to select the group. To select a noncontiguous set of words, select the first word, and then control-click on each subsequent word until all words are selected.


Input Value between 0 & 65535

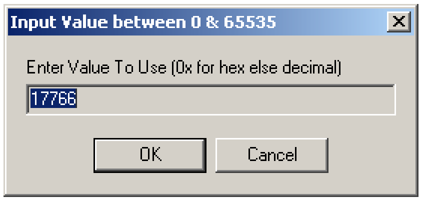
Enter Starting Count (0x for hex else decimal)

17766


Input Value between 0 & 65535

Enter Interval (0x for hex else decimal)

5


Input Value between 0 & 65535

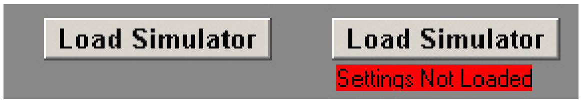
Enter Value To Use (0x for hex else decimal)

17766

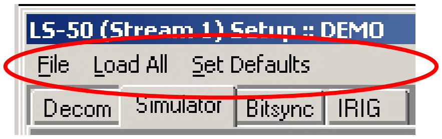
By invoking *"Same Value,"* the user may specify a common value for an individual word, or for a sequence of words. The word sequence may be contiguous or irregular. By invoking *"Random Value,"* the user may populate an individual word, or a sequences of words with random numerical values. As with the other two word value modes, the word sequence for the random values may be contiguous or irregular.

3.2.1.2.10 Load Simulator Button

The Simulator setup tab has a button control to load the setup information entered by the user. Changes made with any of the controls will not take affect until this button is pressed. The user may load all four major functions (Decom, Simulator, Bitsync, and IRIG) from the "Load All" command on the menu next to the File menu). If any changes are made to the Simulator setup without loading, a red text will appear below the Load button (shown below right), indicating the displayed data does not match the cards' loaded data.



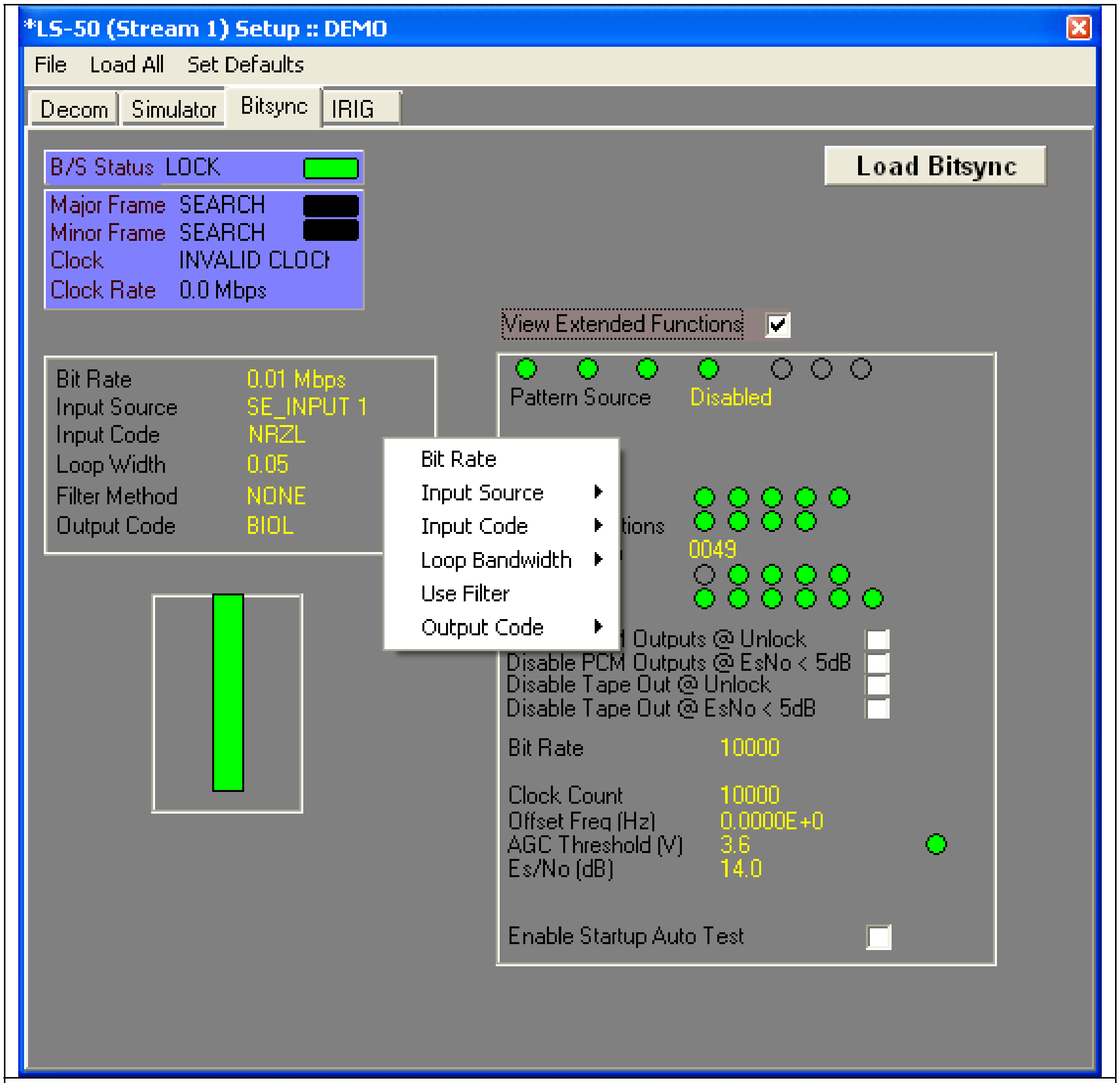
3.2.1.2.11 Saving the Simulator Setup Configuration



Below the window header of the "LS-50 (Stream 1) Setup" display shown in Figure 3-3 on page 79 are the *"File," "Load All,"* and *"Set Defaults"* commands. After the simulator setup configuration is complete, save the settings by invoking the *"File → SaveAs"* command. To download all of the configurations (decom, simulator, Bitsync, and IRIG) to the LS-50-P hardware, invoke the *"Load All"* command. To recall a previously defined LS-50-P setup configuration, invoke the *"File → Recall"* command and select the appropriate file from the file menu and then download the configuration to the LS-50-P hardware by invoking the *"Load All"* command. To set the LS-50-P hardware to its default state, invoke the *"Set Defaults"* command.

3.2.1.3 The LS-50 Bit Synchronizer Tab

The LS-40-DB daughtercard Bit Synchronizer setup tab and it's associated menus and controls are shown in Figure 3-11 below. The View Extended Functions check box is described in detail in paragraph 3.2.1.3.9 on page 108. The Lumistar LS-40-DB Bit Synchronizer daughterboard provides optimal reconstruction of a serial PCM data stream that has been corrupted by noise, phase jitter, amplitude modulation, or base line variations.



B/S Status LOCK 🟩
Major Frame SEARCH
Minor Frame SEARCH
Clock INVALID CLOCI
Clock Rate 0.0 Mbps

Bit Rate	0.01 Mbps
Input Source	SE_INPUT 1
Input Code	NRZL
Loop Width	0.05
Filter Method	NONE
Output Code	BIOL

View Extended Functions ☑

● ● ● ● ○ ○ ○
Pattern Source Disabled

tions 0049

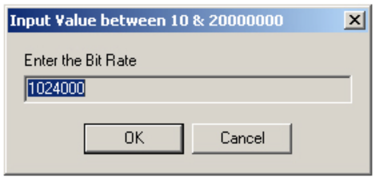
l Outputs @ Unlock ☐
Disable PCM Outputs @ EsNo < 5dB ☐
Disable Tape Out @ Unlock ☐
Disable Tape Out @ EsNo < 5dB ☐

Bit Rate	10000
Clock Count	10000
Offset Freq (Hz)	0.0000E+0
AGC Threshold (V)	3.6
Es/No (dB)	14.0

Enable Startup Auto Test ☐

Figure 3-11 The LS-50 Bit Synchronizer Configuration Menus

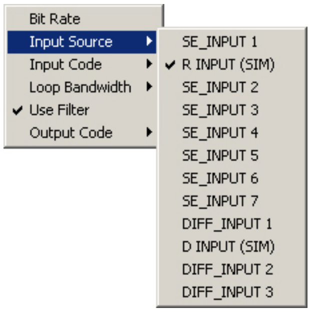
3.2.1.3.1 Input Bit Rate


Input Value between 10 & 20000000

Enter the Bit Rate

1024000

The LS-40-DB20 Bit Synchronizer can operate over an input range of 100 bits per second to 20 Mbps for all NRZ codes, or from 100 bits per second to 10 Mbps for the Bi-Phase and Miller codes. The LS-40-DB10 is limited to 10 Mbps for NRZ codes and 5 Mbps for the Bi-Phase and Miller codes. By invoking the **Input Bit Rate** command, the user may enter the required input data rate in bits per second.



3.2.1.3.2 Input Source

The LS-40-DB Bit Synchronizer can support up to twelve (12) separate input signals. The inputs include both single-ended (SE) and differential (D/Diff) with 50Ω, 75Ω, or 1KΩ (Jumper Select) input impedance. The input signal amplitude supported ranges from 0.1 V pp to 10 V pp. To select the appropriate input, invoke the **Input Source** command and select the specific input from the drop-down list.

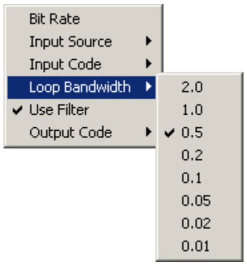
3.2.1.3.3 Input Code

The LS-40-DB Bit Synchronizer supports the PCM input code types specified in Table 3-1 below. Both normal and inverted variants are available. To select the appropriate input code, invoke the **Input Code** command and select the specific input code from the drop-down list.

Table 3-1 LS-40-DB Supported PCM Input Codes (normal or inverted)

NRZ codes	NRZ-L, NRZ-M, NRZ-S
RZ codes	RZ
Split phase codes	BiPhase-L, BiPhase-M, BiPhase-S
Miller codes	DM-M, DM-S, M^2-M, M^2-S
Randomized codes	RNRZ-L, RNRZ-M, RNRZ-S
Randomization sequence	2^{11}-1, 2^{15}-1, 2^{17}-1, 2^{23}-1 (normal or inverted)

3.2.1.3.4 Loop Bandwidth



The Loop-Bandwidth of the PLL circuit in the LS-40-DB may be programmed by the user from 0.01% to 2% depending on the bit rate of the input signal. As described in the "Technical Tidbit" above, The Acquisition Range (0.04% to 8%, depending on the Loop-Bandwidth selected) and the Tracking Range (0.1% to 20%, again depending on the Loop-Bandwidth selected) are both heavily dependent on the loop bandwidth of the PLL. To select the appropriate loop bandwidth, invoke the **Loop Bandwidth** command and select the specific value from the drop-down list.

3.2.1.3.5 Use Filter

The user may enable additional data filtering, prior to the actual phase lock loop of the bit synchronizer by invoking the **Use Filter** command. The additional filter uses a “Raised-Root Cosine” topology and is used to improve the performance metric of the bit synchronizer.

3.2.1.3.6 Output Code

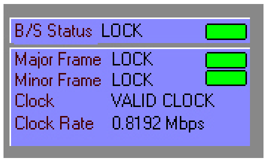
The LS-40-DB Bit Synchronizer supports the PCM output code types specified in Table 3-2 below. Both normal and inverted variants are available. To select the appropriate output code, invoke the **Output Code** command and select the specific output from the drop-down list.

Table 3-2 LS-40-DB Supported PCM Output Codes

NRZ codes	NRZ-L, NRZ-M, NRZ-S, INV_NRZL
RZ codes	RZ, INV_RZ
Split phase codes	BiPhase-L, BiPhase-M, BiPhase-S, INV_BIOL
Miller codes	DM-M, DM-S, M^2-M, M^2-S
Randomized codes	RNRZ-L, RNRZ-M, RNRZ-S
Randomization sequence	2^{11}-1, 2^{15}-1, 2^{17}-1, 2^{23}-1

3.2.1.3.7 Bit Sync Status Display

The LS-50-P Bit Sync setup tab has a window display showing the status of some of the LS-50’s functional states. These states include: bit synchronizer signal lock, major and minor frame lock, a valid clock indication, as well as the clock rate in Mbps. This status display is updated at a ten-hertz rate and is common to all LS-50-P function setup tabs.

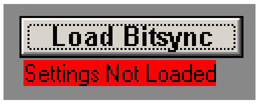


B/S Status LOCK 🟩

Major Frame LOCK 🟩
Minor Frame LOCK 🟩
Clock VALID CLOCK
Clock Rate 0.8192 Mbps

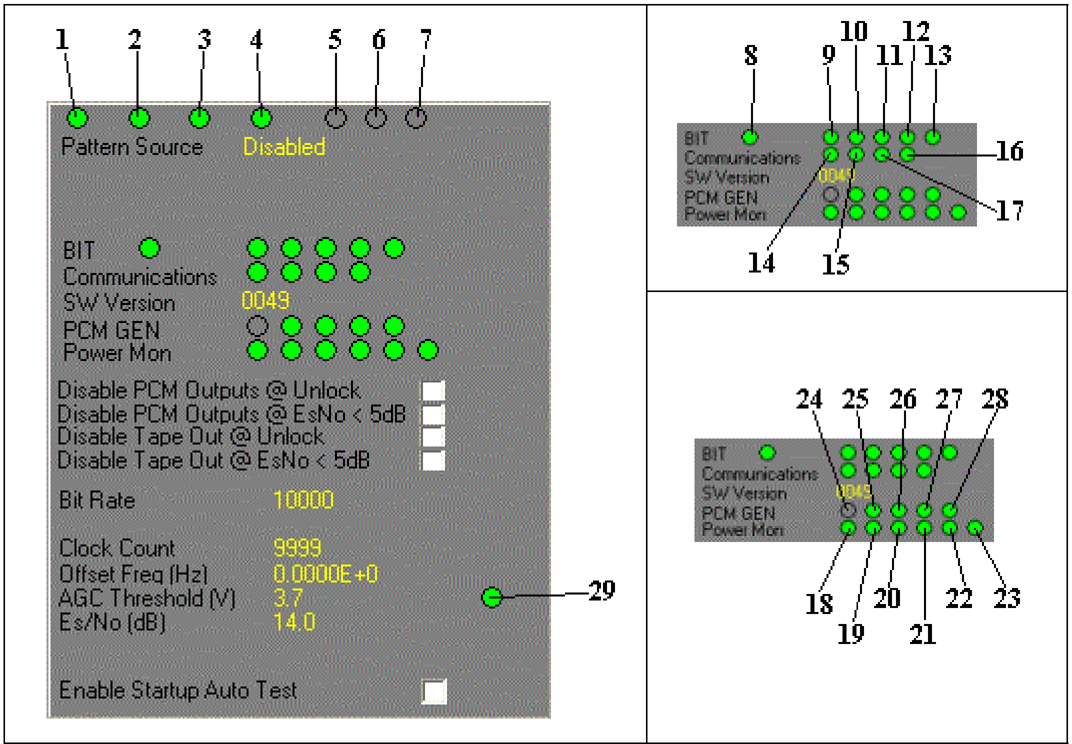
3.2.1.3.8 Load Bit Sync Button

The Bit Synchronizer setup tab has a button control to load the setup information entered by the user. Changes made with any of the controls will not take affect until this button is pressed. The user may load all four major functions (Decom, Simulator, Bitsync, and IRIG) from the “Load All” command on the menu next to the File menu). If any changes are made to the Bit Synchronizer setup without loading, a red text will appear below the Load button (shown below right), indicating the displayed data does not match the cards’ loaded data.



Settings Not Loaded

3.2.1.3.9 View Extended Functions



Pattern Source Disabled

BIT
Communications
SW Version 0049
PCM GEN
Power Mon

Disable PCM Outputs @ Unlock □
Disable PCM Outputs @ EsNo < 5dB □
Disable Tape Out @ Unlock □
Disable Tape Out @ EsNo < 5dB □

Bit Rate 10000

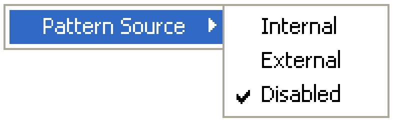
Clock Count 9999
Offset Freq (Hz) 0.0000E+0
AGC Threshold (V) 3.7
Es/No (dB) 14.0

Enable Startup Auto Test □

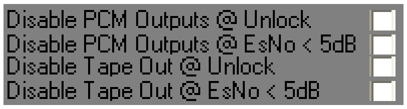
1	Bit Sync Status	16	Protocol Error
2	Input Signal Above Threshold	17	Syntax Error
3	PLL Lock	18	+3.3 VDC
4	Input Quality Above Threshold	19	-5 VDC
5	PRN Correlator Locked	20	+5 VDC
6	PRN Correlator History	21	-12 VDC
7	Error Count Overflow	22	+12 VDC
8	Overall Health Flag	23	VCC (+5 VDC)
9	Synchronization Flag	24	Link Analysis Active
10	Power Source Error	25	Signal Within Range
11	Serial Error	26	Es/No Within Range
12	Hardware Error	27	Offset Frequency
13	Power Up BIT Error	28	Symbol Tracker
14	Overflow	29	Overall Health Flag
15	Semantic Error		

Figure 3-12 Bit Synchronizer Extended Functions Display

3.2.1.3.10 Pattern Source



When the BERT function is enabled (see paragraph 3.2.1.5 on page 114), the user may select the source of the PN pattern by invoking the **Pattern Source** command. Place the cursor in the extended functions display (see Figure 3-12 upper left on page 108) and right click, then select Internal, External, or Disabled.

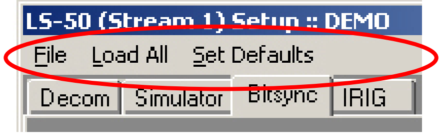


Disable PCM Outputs @ Unlock ☐
Disable PCM Outputs @ EsNo < 5dB ☐
Disable Tape Out @ Unlock ☐
Disable Tape Out @ EsNo < 5dB ☐

3.2.1.3.11 Disable Output Checkboxes

The extended functions feature allows the user to automatically disable the PCM and/or Tape outputs of the bit synchronizer during certain signal conditions. The user may select to disable the PCM output whenever the bit synchronizer is out of lock, and/or when the system E_s/N_0 level drops below 5 dB. The tape output of the bit synchronizer may be similarly controlled.

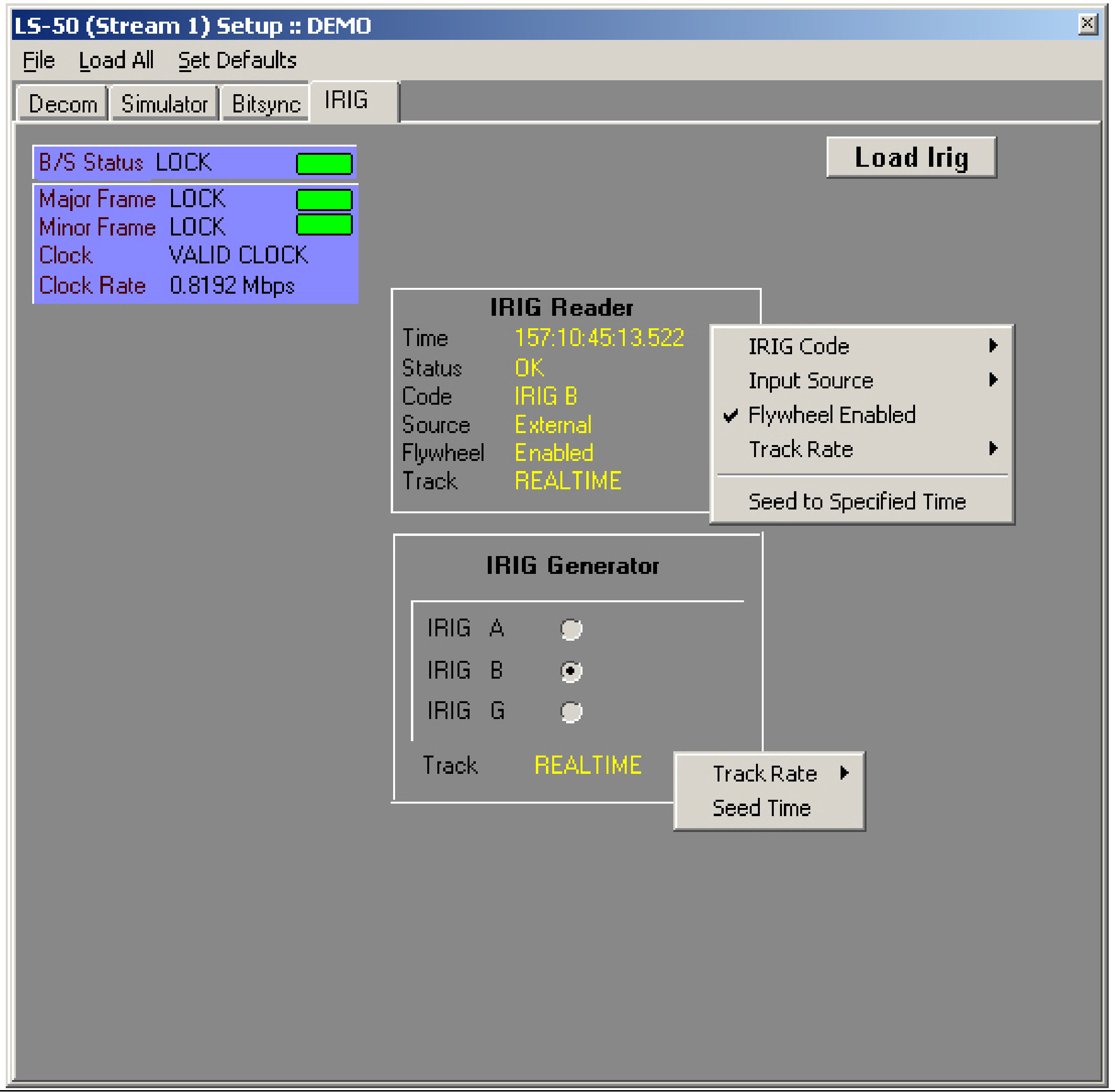
3.2.1.3.12 Saving the Bit Synchronizer Setup Configuration



Below the window header of the "LS-50 (Stream 1) Setup" display shown in Figure 3-3 on page 79 are the *"File," "Load All,"* and *"Set Defaults"* commands. After the bit synchronizer setup configuration is complete, save the settings by invoking the *"File → SaveAs"* command. To download all of the configurations (decom, simulator, Bitsync, and IRIG) to the LS-50-P hardware, invoke the *"Load All"* command. To recall a previously defined LS-50-P setup configuration, invoke the *"File → Recall"* command and select the appropriate file from the file menu and then download the configuration to the LS-50-P hardware by invoking the *"Load All"* command. To set the LS-50-P hardware to its default state, invoke the *"Set Defaults"* command.

3.2.1.4 The LS-50 IRIG Time Code Tab

The LS-50-P IRIG Time Code configuration setup tab and it's associated menus and controls are shown in Figure 3-13 below. The IRIG time code functions include both a reader and generator that can operate with IRIG A, B, or G time code formats. The time code generator creates and outputs time information in accordance with the IRIG 200 time code standards. The time code reader is typically used to insert time information into the PCM minor frame block of data.



B/S Status	LOCK	🟩
Major Frame	LOCK	🟩
Minor Frame	LOCK	🟩
Clock	VALID CLOCK	
Clock Rate	0.8192 Mbps	

IRIG Reader

Time	157:10:45:13.522
Status	OK
Code	IRIG B
Source	External
Flywheel	Enabled
Track	REALTIME

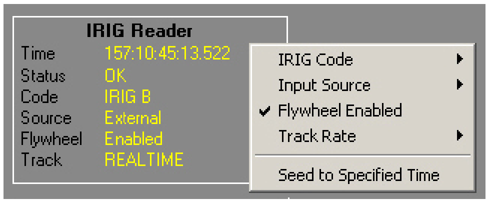
IRIG Generator

IRIG A	○
IRIG B	●
IRIG G	○
Track	REALTIME

Figure 3-13 The LS-50 IRIG Time Code Reader/Generator Configuration Menus

3.2.1.4.1 IRIG Time Code Reader Menu

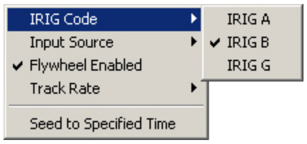
The IRIG time code reader configuration consists of five controls/parameters that include: IRIG Code, Input Source, Flywheel Mode, Tracking Rate, and Seed to Specific Time. Each is discussed in the following paragraphs.


IRIG Reader

Time	157:10:45:13.522
Status	OK
Code	IRIG B
Source	External
Flywheel	Enabled
Track	REALTIME

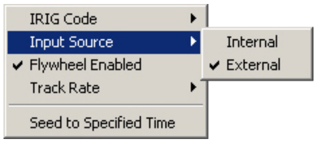
3.2.1.4.2 IRIG Code

The IRIG functionality in the LS-50-P supports three Class-I IRIG frame formats including “A,” “B,” and “G.” To select the appropriate code format, the user invokes the **IRIG Code** command and selects from the drop-down list.



3.2.1.4.3 Input Source

The time source for the IRIG time code reader may be either internal, or external. The user selects the input source by invoking the **Input Source** command. The “Internal” mode derives time information from the LDPS application (see paragraph 3.2.1.4.6 below). The “External” mode connects the reader input to an external time source signal.

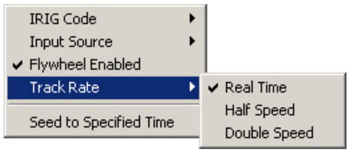


3.2.1.4.4 “Flywheel” Mode

To enable the time code reader to continue to operate, or “flywheel” during dropout periods of the carrier signal, the user must select the **Flywheel Enabled** mode. While in this mode, the IRIG time reader will flywheel if the time carrier was lost for at least one cycle in the last time frame. If the carrier is lost altogether, the reader will continue to flywheel indefinitely, with an accompanying loss of timing accuracy.

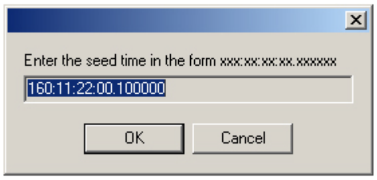
3.2.1.4.5 Track Rate

The IRIG time code reader can operate at several different input carrier frequencies. These include the standard carrier frequency (see the table in the IRIG-200 “Factoid” in the appendix on page 250) and frequencies that are half the standard frequency (half speed) and twice the standard frequency (double speed). The **Track Rate** feature is useful when the source of the incoming time code is coming from a tape recorder playing at either half speed, or double speed. Playing the tape at a different speed will change the carrier frequency of the time code signals recorded on the tape.



3.2.1.4.6 Seed to Specific Time Value

The time code reader can function in the absence of an input carrier. If no carrier is present, the system time from the CPU is used instead. In this scenario, the user may specify an arbitrary initial time, or “Seed” value by invoking the **Set Seed to Specific Time** command. Here

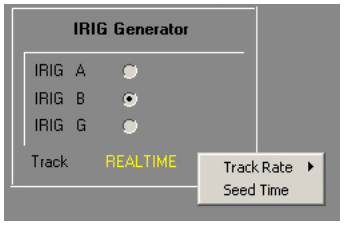

Enter the seed time in the form xxx:xx:xx:xx.xxxxxx

160:11:22:00.100000

the user enters the time in days, hrs, minutes and seconds format as shown right (days:hrs:min:sec).

3.2.1.4.7 IRIG Time Code Generator Menus

The IRIG time code generator configuration consists of two controls/parameters that include: Tracking Rate, and Seed Time. Each is discussed in the following paragraphs.



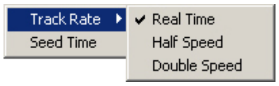
Track REALTIME

3.2.1.4.8 IRIG Code

The IRIG functionality in the LS-50-P supports three class-I IRIG frame formats including “A,” “B,” and “G.” To select the appropriate code format, the user clicks one of the **IRIG Code** radio buttons to make the selection.

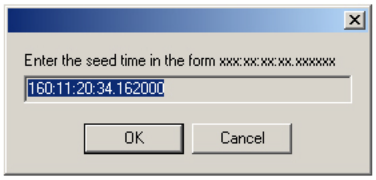
3.2.1.4.9 Track Rate

The IRIG time code generator can operate at several different output carrier frequencies. These include the standard carrier frequency (see the table in the IRIG-200 “Factoid”), and frequencies that are half the standard frequency (half speed) and twice the standard frequency (double speed). The **Track Rate** feature is useful when simulating the outgoing time code coming from a tape recorder playing at either half speed, or double speed.



3.2.1.4.10 Set Seed to Specific Time Value

The user can set the initial time, or “Seed” information within the IRIG frame by invoking the **Seed Time** command. Here the user enters the time in days, hrs, minutes and seconds format as shown left (days:hrs:min:sec).

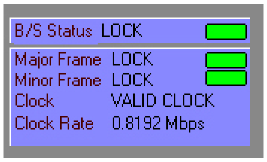


Enter the seed time in the form xxx:xx:xx:xx.xxxxxx

160:11:20:34.162000

3.2.1.4.11 Bit Sync Status Display

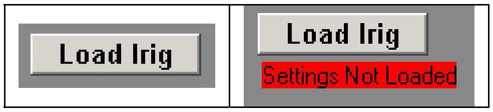
The LS-50-P Bit IRIG setup tab has a window display showing the status of some of the LS-50’s functional states. These states include: bit synchronizer signal lock, major and minor frame lock, a valid clock indication, as well as the clock rate in Mbps. This status display is updated at a ten-hertz rate and is common to all LS-50-P function setup tabs.



B/S Status LOCK 🟩
Major Frame LOCK 🟩
Minor Frame LOCK 🟩
Clock VALID CLOCK
Clock Rate 0.8192 Mbps

3.2.1.4.12 Load IRIG Button

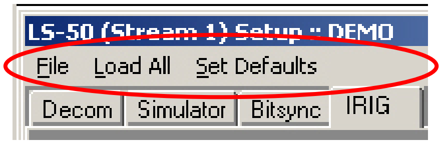
The IRIG setup tab has a button control to load the setup information entered by the user. Changes made with any of the controls will not take affect until this button is pressed. The user may load all four major functions (Decom, Simulator, Bitsync, and IRIG) from the “Load All” command on the menu next to the File menu). If any changes are made to the IRIG setup



Settings Not Loaded

without loading, a red text will appear below the Load button (shown below right), indicating the displayed data does not match the cards' loaded data.

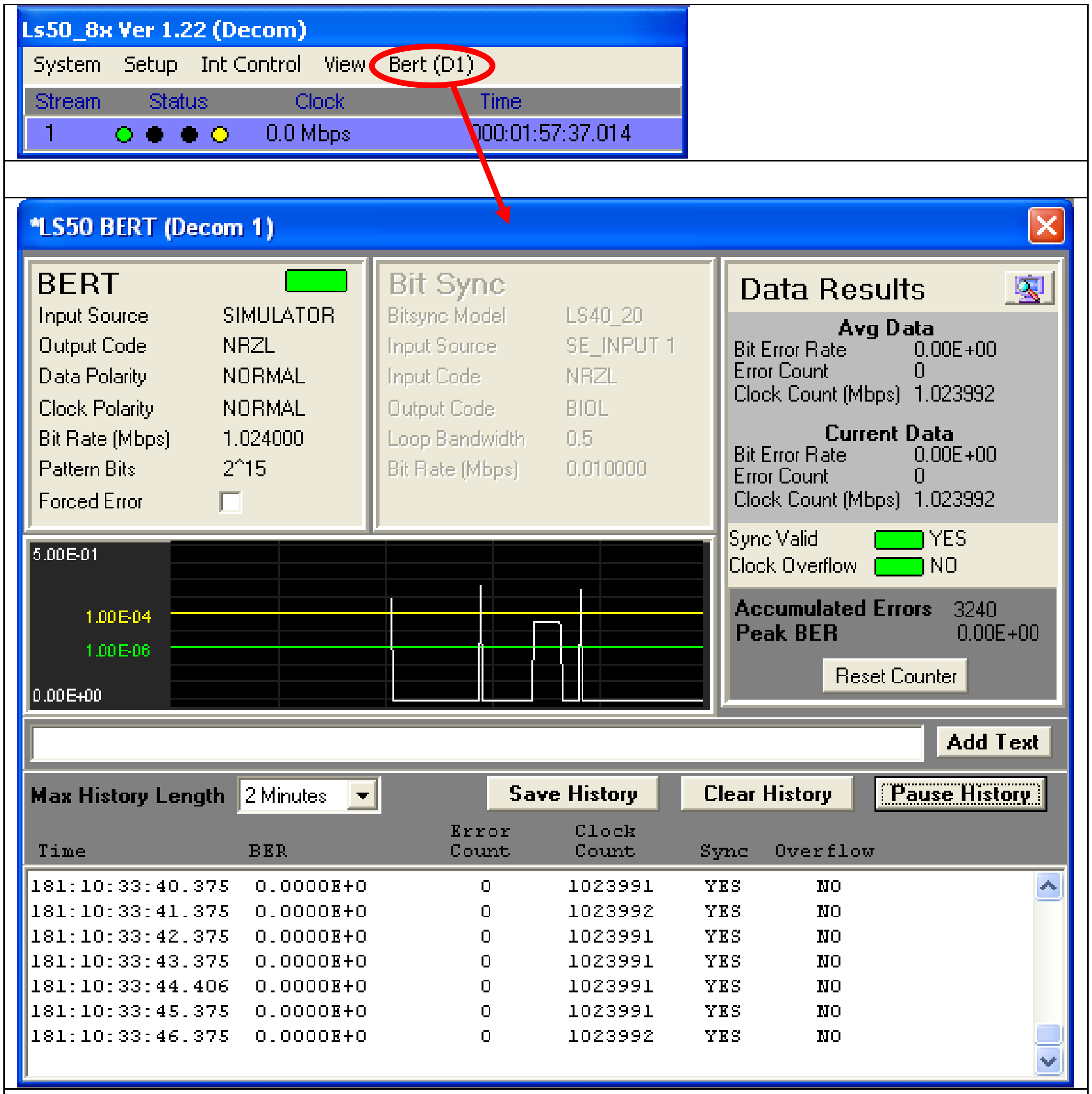
3.2.1.4.13 Saving the IRIG Time Code Setup Configuration



Below the window header of the "LS-50 (Stream 1) Setup" display shown in Figure 3-3 on page 79 are the *"File," "Load All,"* and *"Set Defaults"* commands. After the IRIG time code setup configuration is complete, save the settings by invoking the *"File → SaveAs"* command. To download all of the configurations (decom, simulator, Bitsync, and IRIG) to the LS-50-P hardware, invoke the *"Load All"* command. To recall a previously defined LS-50-P setup configuration, invoke the *"File → Recall"* command and select the appropriate file from the file menu and then download the configuration to the LS-50-P hardware by invoking the *"Load All"* command. To set the LS-50-P hardware to its default state, invoke the *"Set Defaults"* command.

3.2.1.5 LS-50 Bit Error Rate Test (BERT) Function

From the “Ls50_8x (Decom)” display shown below in Figure 3-14, click *“Bert”* to invoke the BERT functionality for the LS-50-P. Note, this feature can only be accessed if the actual LS-50-P hardware is installed in the system. It cannot be simulated, and thus the BERT menu item will not appear if no board is installed.



Stream	Status	Clock	Time
1	○ ● ● ○	0.0 Mbps	000:01:57:37.014

BERT 🟩

Input Source	SIMULATOR
Output Code	NRZL
Data Polarity	NORMAL
Clock Polarity	NORMAL
Bit Rate (Mbps)	1.024000
Pattern Bits	2^15
Forced Error	□

Bit Sync

Bitsync Model	LS40_20
Input Source	SE_INPUT 1
Input Code	NRZL
Output Code	BIOL
Loop Bandwidth	0.5
Bit Rate (Mbps)	0.010000

Data Results

Avg Data

Bit Error Rate	0.00E+00
Error Count	0
Clock Count (Mbps)	1.023992

Current Data

Bit Error Rate	0.00E+00
Error Count	0
Clock Count (Mbps)	1.023992
Sync Valid	🟩 YES
Clock Overflow	🟩 NO
Accumulated Errors	3240
Peak BER	0.00E+00

Max History Length: 2 Minutes

```
                                   Error    Clock
Time              BER              Count    Count     Sync   Overflow
181:10:33:40.375  0.0000E+0        0        1023991   YES    NO
181:10:33:41.375  0.0000E+0        0        1023992   YES    NO
181:10:33:42.375  0.0000E+0        0        1023991   YES    NO
181:10:33:43.375  0.0000E+0        0        1023991   YES    NO
181:10:33:44.406  0.0000E+0        0        1023991   YES    NO
181:10:33:45.375  0.0000E+0        0        1023991   YES    NO
181:10:33:46.375  0.0000E+0        0        1023992   YES    NO
```

Figure 3-14 Configuration Menus/Controls for the LS-50-P BERT Functionality

The BERT configuration display shown in the figure above has several distinct regions that include: the BERT configuration, an optional bit sync configuration (if the selected BERT input source is Mezzanine), a Data Results Display, and a History Display. Each of these regions will be discussed starting in paragraph 3.2.1.5.1 on page 115. Before that however, some background information on the BERT functionality is presented.

The BERT is an instrument that generates a special digital test signal. This signal is sent through the system and the BERT counts the number of bit errors in the recovered signal and provides the user with a Bit Error Rate, or BER. The BER measurement is one of the fundamental parameters that characterize the overall performance of the telemetry system and of many of its components.

As mentioned earlier, errors introduced into the transmission of a digital signal are often random in nature and are strongly affected by system parameters such as signal level, noise level and noise bandwidth, timing jitter, and data rate. The BER is actually a probability and is related to another system parameter - E_b/N_0 (pronounced ebbno). E_b/N_0 is the ratio of the energy-per-bit and the noise-power-per-unit-bandwidth of the digital transmission. The E_b/N_0 as a quantity is a theoretical convenience rather than the direct output of a test measurement device. The parameters that do in effect define the E_b/N_0, and that can be directly measured by the user are the received carrier power (C), and the received noise power (N). These measured parameters, in addition to the noise bandwidth (W) of the system component being tested and the data rate (R_b) of the signal define the system E_b/N_0 in the following relationship:

$$\boxed{\frac{Eb}{No}=\left(\frac{C}{N}\right)\left(\frac{W}{Rb}\right)}$$

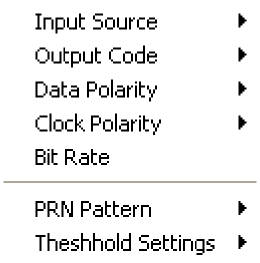
With the system E_b/N_0 defined in terms of measurable quantities, we can now define the BER probability. For example, the BER probability of a digital signal employing bipolar signaling expressed in terms of E_b/N_0 has the following relationship:

$$\boxed{\mathrm{Pe}=\mathrm{Q}\left(\sqrt{\frac{2\mathrm{Eb}}{\mathrm{No}}}\right)}$$

Where E_b is the average energy of a modulated bit, and N_0 is the noise power spectral density (noise in 1-Hz bandwidth). The value Q(X) is called the Gaussian Integral Function and is usually calculated numerically. Note, the quantity “X” will vary mathematically for each type of modulation and signal encoding used in the system.

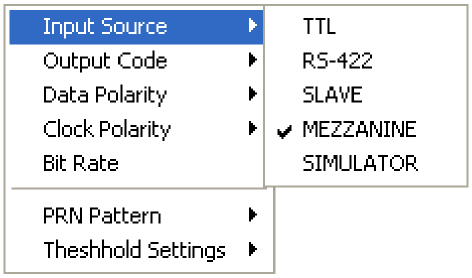
3.2.1.5.1 BERT Configuration Setup Menu

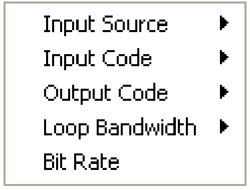
The BERT configuration pane consists of seven (7) controls/parameters that include: Input Source, Output Code, Data Polarity, Clock Polarity, Bit Rate, PRN Pattern, and Threshold Settings.



3.2.1.5.2 Input Source

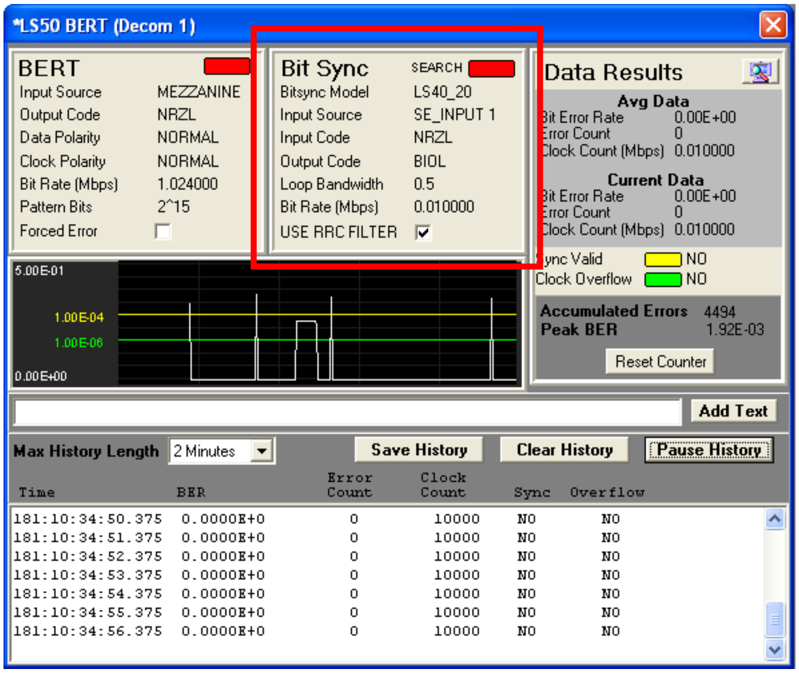
The user may select from one of five input sources by invoking the **Input Source** command and selecting the appropriate input type. The input source may include: TTL, or RS-422 differential inputs, the input from the Slave Port on the decommutator, the Mezzanine bit sync daughtercard (LS-40-DB), or the LS-50's onboard PCM simulator.





Bit Sync Config.

If the selected input source is the Mezzanine (LS-40-DB), then the "Bit Sync" configuration pane will appear next to the BERT configuration pain as shown below (red rectangle). Setup of the LS-40-DB is identical to that described in paragraph 3.2.1.3 on page 105, with the caveat that the configuration established here only applies when the BERT mode is invoked. In other words, the bit sync configuration in BERT mode can be different from the configuration during normal operation.



BERT

Input Source	MEZZANINE
Output Code	NRZL
Data Polarity	NORMAL
Clock Polarity	NORMAL
Bit Rate (Mbps)	1.024000
Pattern Bits	2^15
Forced Error	□

Bit Sync SEARCH

Bitsync Model	LS40_20
Input Source	SE_INPUT 1
Input Code	NRZL
Output Code	BIOL
Loop Bandwidth	0.5
Bit Rate (Mbps)	0.010000
USE RRC FILTER	☑

Data Results

Avg Data

Bit Error Rate	0.00E+00
Error Count	0
Clock Count (Mbps)	0.010000

Current Data

Bit Error Rate	0.00E+00
Error Count	0
Clock Count (Mbps)	0.010000
Sync Valid	NO
Clock Overflow	NO
Accumulated Errors	4494
Peak BER	1.92E-03

Max History Length 2 Minutes

```
                                     Error      Clock
Time                  BER            Count      Count      Sync    Overflow
181:10:34:50.375  0.0000E+0            0        10000      NO        NO
181:10:34:51.375  0.0000E+0            0        10000      NO        NO
181:10:34:52.375  0.0000E+0            0        10000      NO        NO
181:10:34:53.375  0.0000E+0            0        10000      NO        NO
181:10:34:54.375  0.0000E+0            0        10000      NO        NO
181:10:34:55.375  0.0000E+0            0        10000      NO        NO
181:10:34:56.375  0.0000E+0            0        10000      NO        NO
```

3.2.1.5.3 BERT Output Code

The BERT supports the PCM output code types specified in Table 3-2 on page 107. Both normal and inverted variants are available. To select the appropriate output code, invoke the **Output Code** command and select the specific output from the drop-down list.

3.2.1.5.4 BERT Data Polarity

In the telemetry field, certain data transmission & demodulation schemes have inherent ambiguities that may result in the data at the decommutator input being inverted. By invoking the **Data Polarity** command and selecting either "NORMAL" or "INV" (inverted), the ambiguity may be simulated by the BERT.

3.2.1.5.5 BERT Clock Polarity

The BERT essentially has two basic output signals: Clock, and Data. By using the **Clock Polarity** mode, the user may select either polarity sense of the output clock. In essence, the clock polarity mode allows the user to select either the rising or falling edge of the clock to coincide with the output data. For the rising edge, select "NORMAL." For the falling edge, select "INV."

3.2.1.5.6 BERT Bit Rate

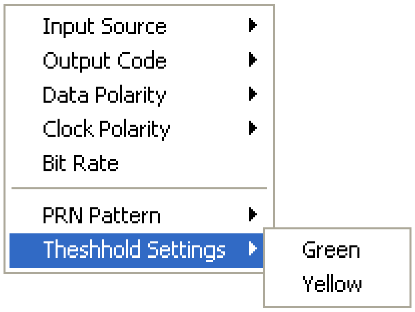
Invoking **Bit Rate** allows the user to specify the output bit rate (bits/second) of the BERT. The user may enter a value between 10 bps to 20 Mbps for NRZ codes, and 10 bps to 10 Mbps for all other codes.

3.2.1.5.7 BERT PRN Pattern

Selecting the proper PN sequence that will be appropriate for the particular system being tested is important. Some of the key properties of the selected PN sequence that are of importance include: the length of the PN Sequence, the type of Linear Feedback Shift Register configuration used to implement the PN generator (this defines the binary run properties of the sequence), and the spectral line spacing of the sequence (which depends on the bit rate of the sequence). The user may select from one of seven (7) PN sequences by invoking the **PRN Pattern** command. Available pattern lengths include: 2^{11}-1, 2^{15}-1, 2^{17}-1, 2^{19}-1, 2^{21}-1, 2^{23}-1, and 2^{25}-1.

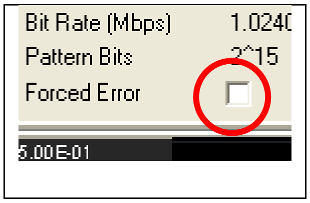
3.2.1.5.8 BERT Threshold Settings

The strip cart recorder pane shown in Figure 3-15 on page 118 has two error threshold lines that may be manipulated by the user. Invoke the **Threshold Settings** command, and select either Green or Yellow. Enter the threshold value in scientific notation (X.xxE±Y) in the resulting dialog box.



3.2.1.5.9 Forced Error Checkbox

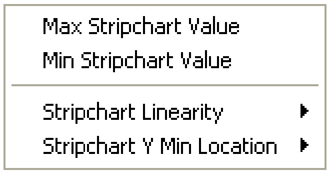
To introduce bit errors at a know rate, the user may click the **Forced Error** checkbox. This will inject a single bit error that will repeat once every $2^{\mathbf{n}}$-1 bits, where "**n**" is the length of the PN pattern selected by the user (see paragraph 3.2.1.5.7). Use this feature to calibrate a test scenario that is in an unknown and un-quantified state.



Bit Rate (Mbps) 1.024(
Pattern Bits 2^15
Forced Error □
5.00E-01

3.2.1.5.10 BER Strip Chart Configuration

The BERT Strip Chart configuration pane consists of four (4) controls/parameters that include: Max Stripchart Value, Min Stripchart Value, Stripchart Linearity, and Stripchart Y Min Location. To invoke the configuration menu (shown right), place the cursor in the display shown below, and right click.



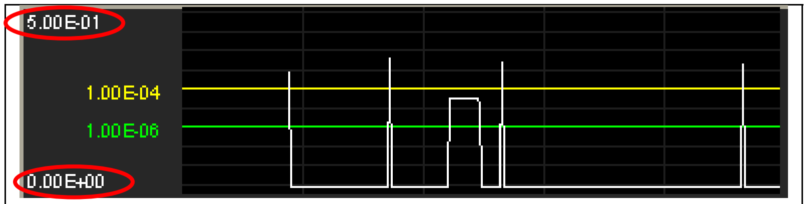
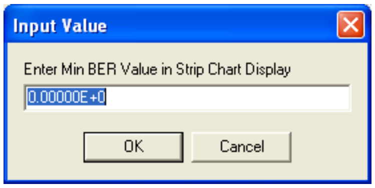


Figure 3-15 BER Strip Chart Recorder Display

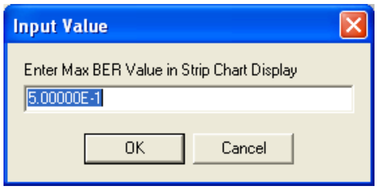
3.2.1.5.11 Min and Max Strip Chart Values

To specify the extreme values for the strip chart (red ovals in Figure 3-15) the user must invoke both the **Max Stripchart Value** and **Min Stripchart Value** command and enter the value in scientific notation (X.xxE ± Y) in the resulting dialog boxes.


Input Value

Enter Min BER Value in Strip Chart Display

0.00000E+0

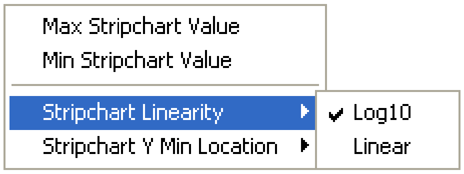

Input Value

Enter Max BER Value in Strip Chart Display

5.00000E-1

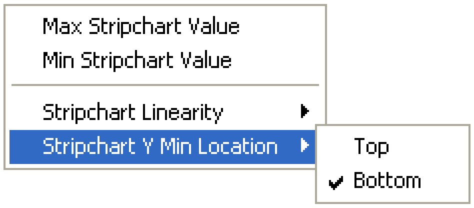
3.2.1.5.12 Strip Chart Linearity

The BER strip chart can display data in either linear, or logarithmic (Logbase10) format. Invoke the **Stripchart Linearity** command and select either “Linear” or “Log10.”



3.2.1.5.13 Strip Chart Y Min Location

The vertical location of the minimum value specified in paragraph 3.2.1.5.11 may be placed either at the top or bottom of the strip chart display by invoking the **Stripchart Y Min Location** command.

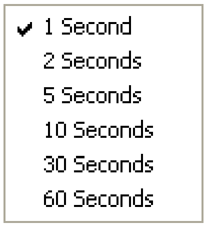


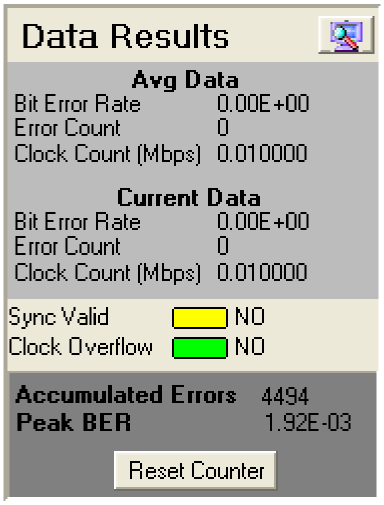
3.2.1.5.14 Data Results Display

The BER data results are displayed as shown in Figure 3-16 below. Both long-term and instantaneous values for bit error rate, error count, and clock are displayed. Status indicators for "Sync Valid" and "Clock Overflow" are also provided. Total errors counted and the peak BER value encountered are displayed. Both of these values continue to update until the **Reset Counter** button is clicked, at which time both values will return to zero.

3.2.1.5.15 BER Average Period

The average values for bit error rate, error count, and clock are calculated during a time interval defined by the user by invoking the **BER Average Period** command and selecting an interval from 1 to 60 seconds in length. To invoke the command, place the cursor at the bottom of the display shown in Figure 3-16 below and right click (resulting menu shown right).




Data Results

Avg Data

Bit Error Rate	0.00E+00
Error Count	0
Clock Count (Mbps)	0.010000

Current Data

Bit Error Rate	0.00E+00
Error Count	0
Clock Count (Mbps)	0.010000

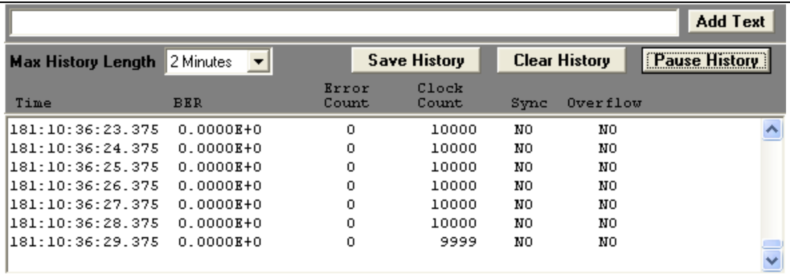
Sync Valid	🟨	NO
Clock Overflow	🟩	NO

Accumulated Errors	4494
Peak BER	1.92E-03

Figure 3-16 BER Data Results Display

3.2.1.5.16 History Display

At the bottom of the BERT configuration and status display is the BER history recording as shown in Figure 3-17 below. The history is listed chronologically and has a user defined length from 2 minutes to 24 hours. The history may be annotated by entering text in the text box and clicking the **Add Text** button. To save the history, click the **Save History** button and enter a file name and location in the resulting dialog box. At any time, the history may be suspended by clicking the **Pause History** button. To clear the history and begin again, click the **Clear History** button.



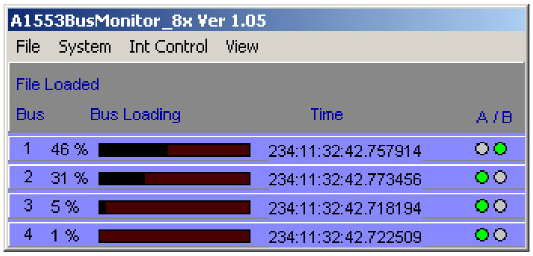
Max History Length: 2 Minutes

```
                                     Error      Clock
Time                BER              Count      Count      Sync   Overflow
181:10:36:23.375   0.0000E+0          0         10000      NO       NO
181:10:36:24.375   0.0000E+0          0         10000      NO       NO
181:10:36:25.375   0.0000E+0          0         10000      NO       NO
181:10:36:26.375   0.0000E+0          0         10000      NO       NO
181:10:36:27.375   0.0000E+0          0         10000      NO       NO
181:10:36:28.375   0.0000E+0          0         10000      NO       NO
181:10:36:29.375   0.0000E+0          0          9999      NO       NO
```

Figure 3-17 BER History Display

3.3 The MIL-STD-1553 Bus Monitor

The LDPS 1553 Bus Monitor allows the monitoring of up to eight (8) dual redundant MIL-STD-1553 busses, and also displays some statistics for the busses. The 1553 interface is through a DLL, which can be developed by the user to support any vendor brand of 1553 card.



File Loaded

Bus	Bus Loading	Time	A / B
1	46 %	234:11:32:42.757914	○●
2	31 %	234:11:32:42.773456	●○
3	5 %	234:11:32:42.718194	●○
4	1 %	234:11:32:42.722509	●○

The MIL-STD-1553 bus monitor setup GUI shown above right is a small window launched from the device manager on the Server. This GUI controls all MIL-STD-1553 cards[15] installed in the system. The 1553 bus monitor setup GUI has three major parts: The caption area, the menu area, and the display/status area.

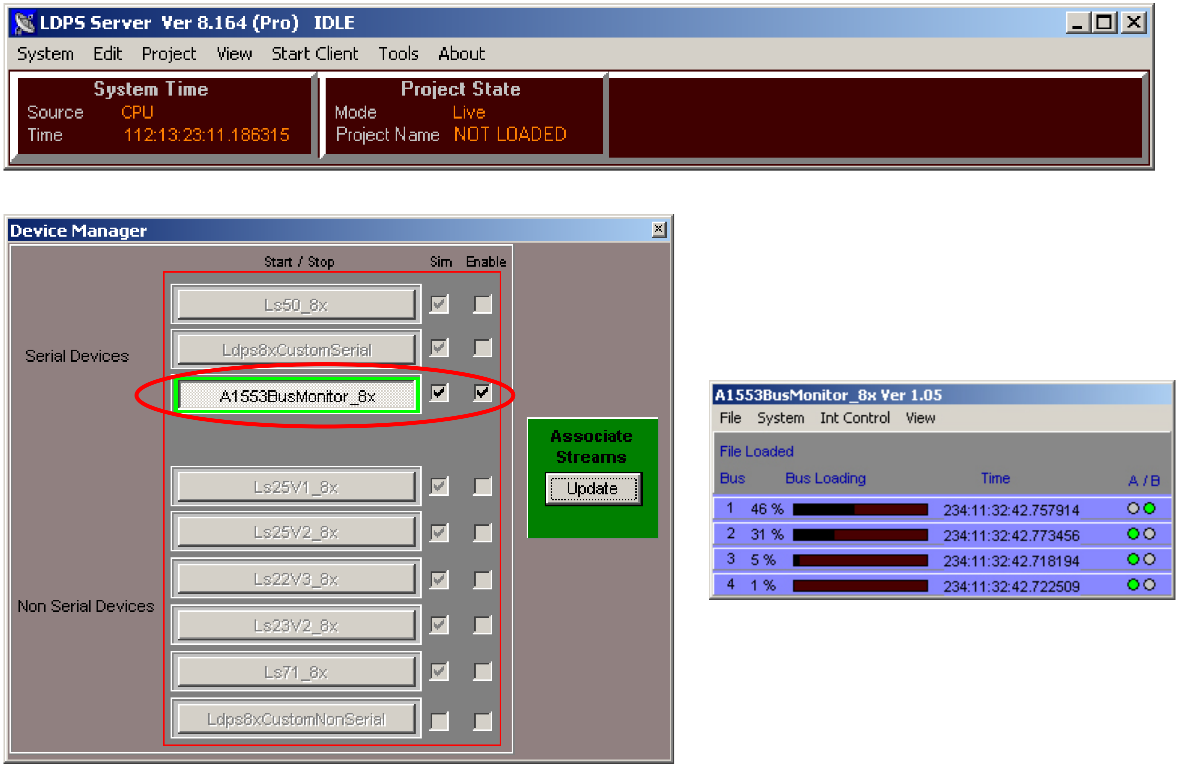
The caption area (top of the window) displays the name of the window. Note there is no normal window closing mechanism in the caption area (the X normally found in the upper right corner of windows). This is because the window must be active in order to gather data from the card, and to control it.

Below the caption area and setup menu is the status display for all MIL-STD-1553 cards installed in the system. For each 1553 bus, the status display shows the time, the bus loading percent numerical value, along with a horizontal bar graph of same. Also displayed are two LEDs indicating which bus (A or B) is currently active.

To initially configure the 1553 Bus Monitor, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select "Devices" and then "Manage" *(System→ Devices→ Manage)*
2. From the System Manager shown in Figure 3-18 below left, select the "Enable" check box next to the 1553 Bus Monitor button. The "A1553BusMonitor_8x" button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the "Sim" check box next to the "Enable" check box. Checking this box allows the LDPS application to operate when a 1553 board is not installed in the system.
3. From the System Manager, click the "A1553BusMonitor_8x " button. This will launch the "A1553BusMonitor_8x display shown in Figure 3-18 below right. Note that the red rectangle around the button has changed to green indicating that the application is now running.

[15] It should be noted that Lumistar does not currently manufacture any MIL-STD-1553 hardware. LDPS only supplies the GUI and programming hooks for the user to develop their own 1553 capabilities.



Source CPU
Time 112:13:23:11.186315

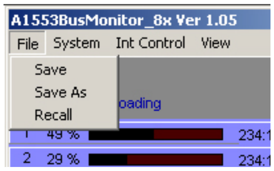
Mode Live
Project Name NOT LOADED

Bus	Bus Loading	Time	A / B
1	46 %	234:11:32:42.757914	○○
2	31 %	234:11:32:42.773456	○○
3	5 %	234:11:32:42.718194	○○
4	1 %	234:11:32:42.722509	○○

Figure 3-18 Server Application Windows (1553 Bus Monitor Enabled)

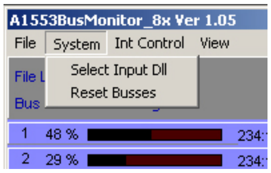
3.3.1 File Menu

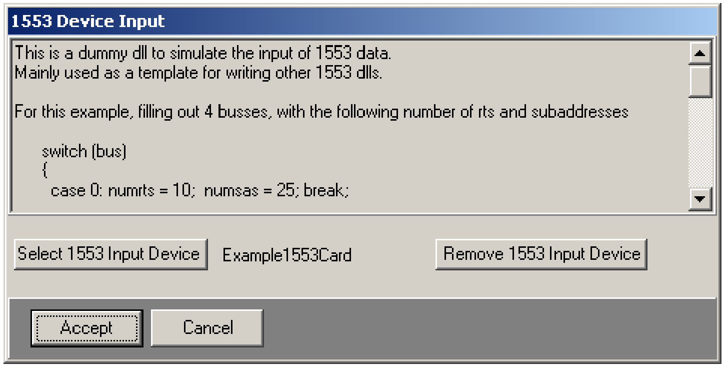
The File Menu allows the user to **Recall** previously defined 1553 hardware configurations in the system. The configuration(s) may be used as is, or the user may change and **Save** the configuration changes to the current file, or **Save As** to a new/different file. After a configuration file has been recalled and/or saved, the name of the file will appear in the title bar of the window. Note the file extension (*.1553) for the 1553 hardware configuration file.



3.3.2 System Menu

The System Menu allows the user to select the file name of the device control DLL by clicking the **Select Input DLL** command. This will invoke the 1553 device DLL input window shown in Figure 3-19 below. The DLL input window defaults to a sample 1553 DLL supplied with LDPS. To select an alternate DLL, click the “Remove 1553 Input Device” button and select the required DLL file name by clicking the “Select 1553 Input Device” button. Selecting the DLL file name will start the DLL running. To reset the device (s) and their 1553 busses to a know initial state, select the **Reset Busses** command. The DLL input window also displays a help file for the DLL. The full text of the help file for the sample 1553 DLL is shown below.





```
This is a dummy dll to simulate the input of 1553 data.
Mainly used as a template for writing other 1553 dlls.

For this example, filling out 4 busses, with the following number of rts and subaddresses

    switch (bus)
    {
     case 0: numrts = 10;  numsas = 25; break;
```

Example1553Card

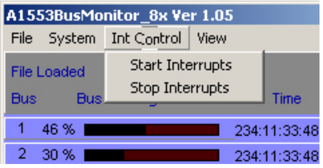
Figure 3-19 1553 Device DLL Input Window & Help File

Sample 1553 DLL help file contents:

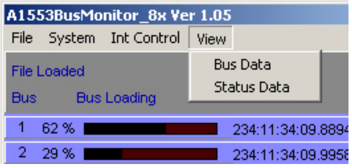
```
This is a dummy dll to simulate the input of 1553 data.
Mainly used as a template for writing other 1553 dlls.

For this example, filling out 4 busses, with the following number of RTs and subaddresses

      switch (bus)
      {
        case 0: numrts = 10;  numsas = 25; break;
        case 1: numrts = 10;  numsas = 15; break;
        case 2: numrts = 10;  numsas =  5; break;
        case 3: numrts = 5;  numsas = 2; break;
        default: numrts = 5;  numsas = 5; break;
      }

The t/r bit is true every other bus

Dataword Values are 1 thru 32

The status info throws errors randomly, as well as changing which bus it came from (A/B)
//----status info
  status->SequenceType = (a1553msgseqtype)(i % 3);

  i =i % 1000;
  status->BusA = (i >= 100);
  status->MsgError = (i == 20);
  status->FormatError = (i == 30);
  status->ResponseTimeOut = (i == 40);
  status->WordCountError = (i == 50);
  status->SyncError = (i == 60);
  status->InvalidWordError = (i == 70);
```

3.3.3 Int Control Menu

The interrupt control function allows the user to start or stop interrupt activity in unison for all MIL-STD-1553 cards installed in the system. The **Start Interrupts** command enables interrupts for all 1553 cards. When new data is ready from the 1553 card, they signal with an interrupt. The LDPS program picks up the data and stores/processes the data. This is also a convenient way to reload, or download, the setup data to the cards. When starting the interrupts in this manner, the cards are downloaded with their setup data first. The **Stop Interrupts** command stops interrupt activity on all 1553 cares in the system.



3.3.4 View Menu



The view menu allows the user to display data and status for up to eight (8) MIL-STD-1553 bus supported by the hardware in the system. The **Bus Data** command, invokes the 1553 Bus Monitor display shown in Figure 3-20 on page 124. The **Status Data** command invokes the 1553 Bus Monitor Status display shown in Figure 3-21 on page 125. This display shows the updating values for all 1553 parameters defined in the database.

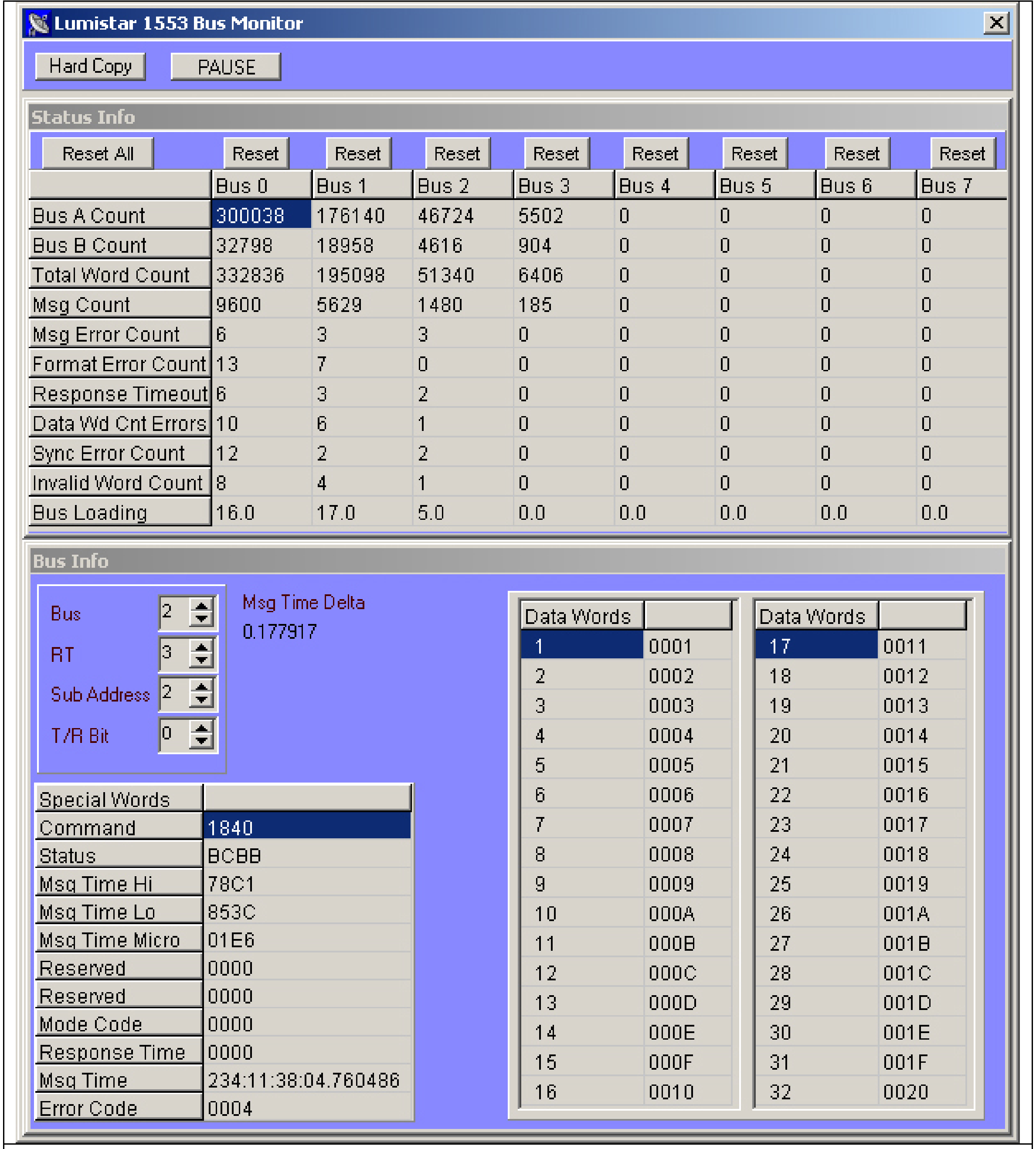
3.3.5 The 1553 Bus Monitor Display Window

The 1553 Bus Monitor display window is shown in Figure 3-20 on page 124. To invoke this display, select the View Bus Data command (View → *Bus Data*) from the 1553 bus monitor setup GUI. There are three main sections of the GUI. At the top left are two controls: Pause, and Hard Copy. Clicking the **Pause** button halts the updating (at a 10 Hertz rate) of numerical data on the display. Clicking the **Hard Copy** button takes a snapshot of the application window and saves the image (Windows BMP format) in the root directory where the bus monitor application is located.

The bus status information occupies the upper half of the display below the controls mentioned above. This area contains statistical counters for each bus, as follows:

- Bus A Count – Counts every word occurrence on the A bus.
- Bus B Count – Counts every word occurrence on the B bus.
- Total Word Count – Counts the total number of word occurrences on both the A and B buses.
- Msg Count – The number of times a new message sequence is stored.
- Msg Error Count – The number of times a message had an error.
- Format Error Count – The number of times a message had a format error.
- Response Timeout – The number of times there was a message response timeout.
- Data Word Error – The number of times a data word count error occurred.
- Sync Error – The number of times a sync error on the bus occurred.
- Invalid Word Count – The number of times an invalid word count on the bus occurred.
- Bus Loading – The percentage of the maximum theoretical usage of the bus. The percentage is based on the total word count, each word being 20 bits long with a rate of 1,024,000 bits per second at maximum percentage of use. The bus loading percentage is calculated at a 1 hertz rate. The calculation is thus:
    - totalbits = AryTotalWordCount[busnum] * 20.0;
    - maxbits = dtime * 1024000.0; //max theoretical bits per second
    - loading = totalbits/maxbits;

Individual statistics for each bus may independently be reset by clicking the **Reset** button above the respective column (0 through 7). Click the **Reset All** button to reset all statistics to zero.



	Bus 0	Bus 1	Bus 2	Bus 3	Bus 4	Bus 5	Bus 6	Bus 7
Bus A Count	300038	176140	46724	5502	0	0	0	0
Bus B Count	32798	18958	4616	904	0	0	0	0
Total Word Count	332836	195098	51340	6406	0	0	0	0
Msg Count	9600	5629	1480	185	0	0	0	0
Msg Error Count	6	3	3	0	0	0	0	0
Format Error Count	13	7	0	0	0	0	0	0
Response Timeout	6	3	2	0	0	0	0	0
Data Wd Cnt Errors	10	6	1	0	0	0	0	0
Sync Error Count	12	2	2	0	0	0	0	0
Invalid Word Count	8	4	1	0	0	0	0	0
Bus Loading	16.0	17.0	5.0	0.0	0.0	0.0	0.0	0.0

Bus 2
RT 3
Sub Address 2
T/R Bit 0

Msg Time Delta
0.177917

Special Words	
Command	1840
Status	BCBB
Msg Time Hi	78C1
Msg Time Lo	853C
Msg Time Micro	01E6
Reserved	0000
Reserved	0000
Mode Code	0000
Response Time	0000
Msg Time	234:11:38:04.760486
Error Code	0004

Data Words	
1	0001
2	0002
3	0003
4	0004
5	0005
6	0006
7	0007
8	0008
9	0009
10	000A
11	000B
12	000C
13	000D
14	000E
15	000F
16	0010

Data Words	
17	0011
18	0012
19	0013
20	0014
21	0015
22	0016
23	0017
24	0018
25	0019
26	001A
27	001B
28	001C
29	001D
30	001E
31	001F
32	0020

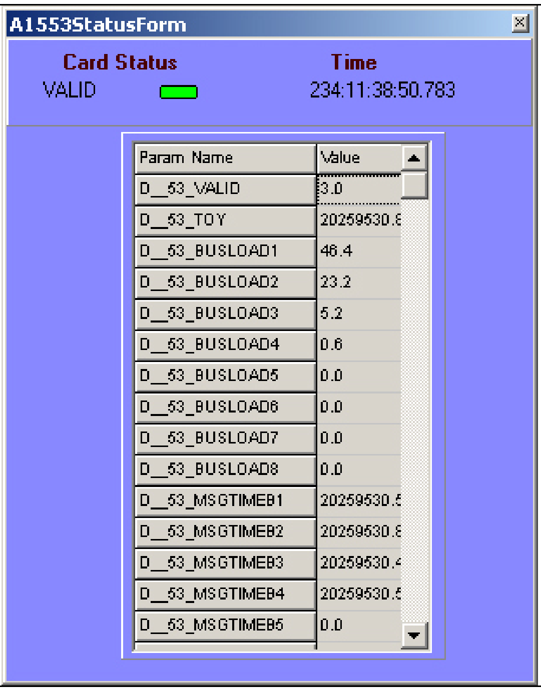
Figure 3-20 MIL-STD-1553 Bus Monitor Window

The bus information section occupies the lower portion the 1553 bus monitor window shown in the figure above. The bus info area displays the data values for the selected bus, sub address, R/T number and T/R bit position. There are four main areas of the bus info section that include:

- Word Selection – It is impossible to display all 600,000+ words on a single display. Using four click wheels to select the desired Bus, R/T number, Sub address number, and Transmit bit position, the user may select the area of interest in the stream. The other sections will display data specific to these selections.

- Message Time Delta – To the right of the word selector is displayed the time difference between the 1553 cards' IRIG time and the time the last message was updated. The time delta can indicate if the data is stale.
- Special Words – There are forty-one (41) words in each array of 1553 data. The array consisting of ([bus][T/R bit][subaddress][RT num][word]) with lengths of [8][2][32][32][41]. Of those forty-one words, thirty-two (32) are data words. The other words are defined as special words as follows:
  1) Command Word
  2) Status Word
  3) Message Time Hi
  4) Message Time Lo
  5) Message Time Micro
  6) Reserved 1
  7) Reserved 2
  8) Mode Code
  9) Response Time

  These values are displayed in hex. For convenience, the three message time words (3,4, & 5) are put together to display the time of day format.
- Data Words – There are thirty-two (32) data words in the 1553 array selected. The values are displayed in hex.



Card Status
VALID 🟩

Time
234:11:38:50.783

Param Name	Value
D__53_VALID	3.0
D__53_TOY	20259530.8
D__53_BUSLOAD1	46.4
D__53_BUSLOAD2	23.2
D__53_BUSLOAD3	5.2
D__53_BUSLOAD4	0.6
D__53_BUSLOAD5	0.0
D__53_BUSLOAD6	0.0
D__53_BUSLOAD7	0.0
D__53_BUSLOAD8	0.0
D__53_MSGTIMEB1	20259530.5
D__53_MSGTIMEB2	20259530.8
D__53_MSGTIMEB3	20259530.4
D__53_MSGTIMEB4	20259530.5
D__53_MSGTIMEB5	0.0

Figure 3-21 MIL-STD-1553 Bus Monitor Status Window

3.3.6 Some Notes About the 1553 Bus Display Window

The serial data display window shown in Figure 2-2 on page 46, was intended more for a PCM frame format rather than a 1553 structure. Thus the 1553 bus monitor seen in Figure 3-20 on page 124 is a more appropriate tool for viewing 1553 data. In LDPS, MIL-STD-1553 data is input from the interface card as a major PCM frame containing 128 minor frames, with each minor frame of data containing a 1553 message structure of 47 words. The meanings of the minor frame words are as follows:

- Message timestamp MSW – Most significant two bytes of a double precision number for the timestamp.
- Message timestamp NMSW – Next most significant two bytes of a double precision number for the timestamp.
- Message timestamp NLSW – Next to least significant two bytes of a double precision number for the timestamp.
- Message timestamp LSW – Least significant two bytes of a double precision number for the timestamp.
- Bus ID/Response Time 10th Microsecond – The lower byte is the bus ID, the upper byte is the response time in tenths of microseconds.
- Message Type – 0 = BC-to-RT, 1 = RT-to-BC, 2 = RT-to-RT.
- Bus A/Msg Error – The lower byte indicates if the message is from Bus A. The upper byte indicates if there is a message error.
- Format Error/Response Time Out – The lower byte indicates if there is a format error. The upper byte indicates if there is a response timeout.
- Word Count Error/Sync Error – The lower byte indicates if there is a word count error. The upper byte indicates if there is a sync error.
- Invalid Word/Spare – The lower byte indicates if there is an invalid word. The upper byte is a spare byte.
- Command Word 1 – Only one command word is used except for RT-to-RT type messages.
- Command Word 2 – Only valid for RT-to-RT type messages.
- Data Word 1 through Data Word 32 – The data words are located in word numbers 13 through 45. If the message only uses “N” data words, then the other data word values will be zero.
- Status Word 1 – Only one status word is used except for RT-to-RT type messages.
- Status Word 2 – Only valid for RT-to-RT type messages.

Below is shown the actual data structure used in the code.

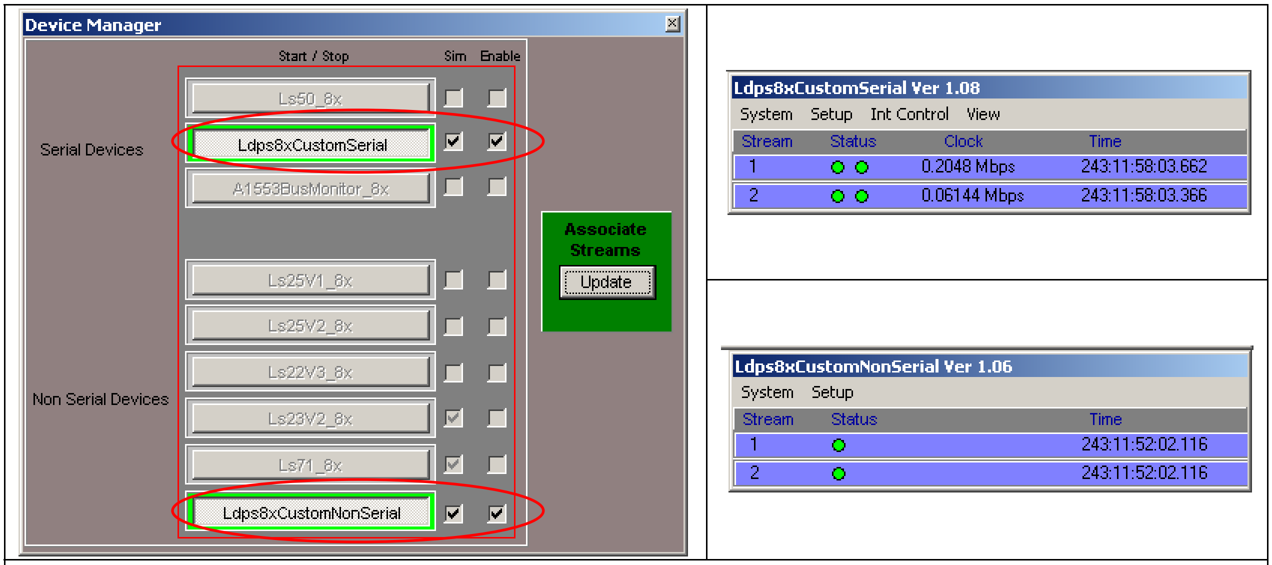
```
typedef struct //12 bytes
{
  a1553msgseqtype SequenceType; //type of sequence
  bool BusA;  //if the msg seq is from bus a (true) or bus b
  bool MsgError; //a 1553 message error was encountered
  bool FormatError; //indicates a 1553 frame error was encountered
  bool ResponseTimeOut; //a 1553 response timed out
  bool WordCountError; //a 1553 word count error occured
  bool SyncError; //a 1553 sync type error occured
  bool InvalidWordError; //a 1553 invalid word occured
  bool Spare; //so can have an even number of bytes
}a1553msgseqstatusstruct;

//this is the root for decoding and archiving and playback
typedef struct  //94 bytes
{
  double SeqTimeStamp; //time stamp for the sequence
  unsigned char BusId; //busid (0..7) this belongs to
  unsigned char ResponseTime10thMicro; //response time in tenths of microseconds
  a1553msgseqstatusstruct SequenceStatus; //status for the sequence
  unsigned short CommandWordAry[2];  //command words, (rt to rt has 2 command words)
  unsigned short DataWordAry[32];    //max of 32 data words
  unsigned short StatusWordAry[2];   //max of 2 status words (rt to rt has 2 status
words)
}a1553msgseqstruct;
```

3.4 Custom Serial & Non-Serial Devices

The LDPS Custom Serial and Non-Serial device support allows the monitoring of multiple serial and non-serial data streams, and also displays some status for the streams. The serial and non-serial interfaces are supported through a DLL, which can be developed by the user to support any third party serial or non-serial hardware device.

To initially configure the custom serial or non-serial hardware devices installed in the system, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select "Devices" and then "Manage" *(System→ Devices→ Manage)*
2. From the System Manager shown in Figure 3-22 below left, select the "Enable" check box next to either of the Ldps8xCustomSerial or Ldps8xCustomNonSerial buttons. The enabled button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the "Sim" check box next to the "Enable" check box. Checking this box allows the LDPS application to operate when a serial or non-serial board(s) are not installed in the system.
3. From the System Manager, click either of the "Ldps8xCustomSerial," or "Ldps8xCustomNonSerial" buttons. This will launch the "Ldps8xCustomSerial," or "Ldps8xCustomNonSerial" displays shown in Figure 3-22 below right. Note that the red rectangle around the button has changed to green indicating that the application is now running.



Device Manager

	Start / Stop	Sim	Enable
Serial Devices	Ls50_8x	□	□
	Ldps8xCustomSerial	☑	☑
	A1553BusMonitor_8x	□	□
Non Serial Devices	Ls25V1_8x	□	□
	Ls25V2_8x	□	□
	Ls22V3_8x	□	□
	Ls23V2_8x	☑	□
	Ls71_8x	☑	□
	Ldps8xCustomNonSerial	☑	☑

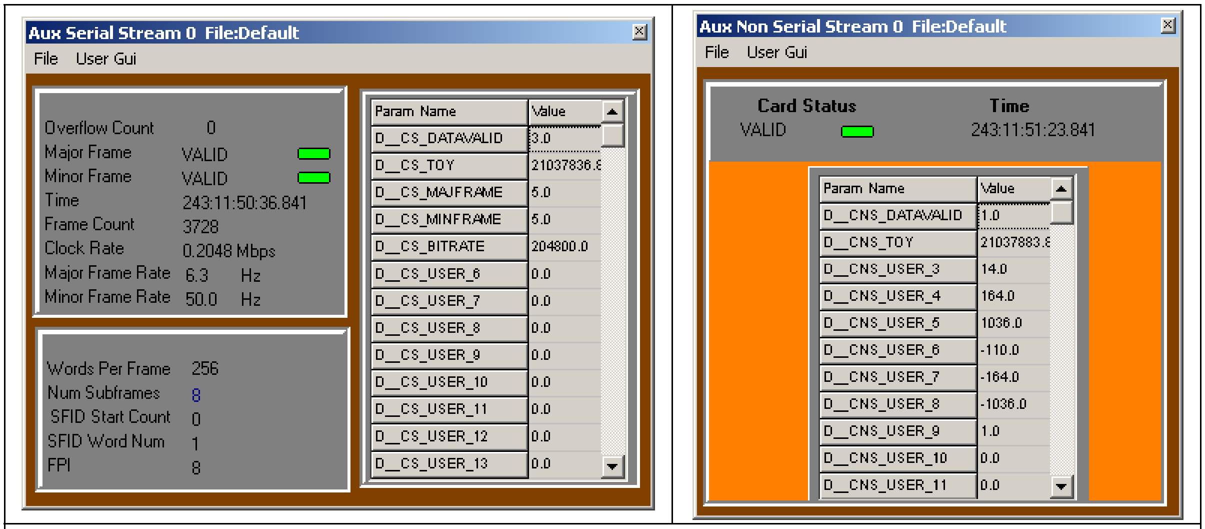
Ldps8xCustomSerial Ver 1.08

Stream	Status	Clock	Time
1	● ●	0.2048 Mbps	243:11:58:03.662
2	● ●	0.06144 Mbps	243:11:58:03.366

Ldps8xCustomNonSerial Ver 1.06

Stream	Status	Time
1	●	243:11:52:02.116
2	●	243:11:52:02.116

Figure 3-22 Invoking Custom Serial & Non-Serial Devices


Aux Serial Stream 0 File:Default
File
User Gui
Aux Non Serial Stream 0 File:Default
File
User Gui

Overflow Count	0	
Major Frame	VALID	🟩
Minor Frame	VALID	🟩
Time	243:11:50:36.841	
Frame Count	3728	
Clock Rate	0.2048 Mbps	
Major Frame Rate	6.3	Hz
Minor Frame Rate	50.0	Hz

Words Per Frame	256
Num Subframes	8
SFID Start Count	0
SFID Word Num	1
FPI	8

Param Name	Value
D__CS_DATAVALID	3.0
D__CS_TOY	21037836.8
D__CS_MAJFRAME	5.0
D__CS_MINFRAME	5.0
D__CS_BITRATE	204800.0
D__CS_USER_6	0.0
D__CS_USER_7	0.0
D__CS_USER_8	0.0
D__CS_USER_9	0.0
D__CS_USER_10	0.0
D__CS_USER_11	0.0
D__CS_USER_12	0.0
D__CS_USER_13	0.0

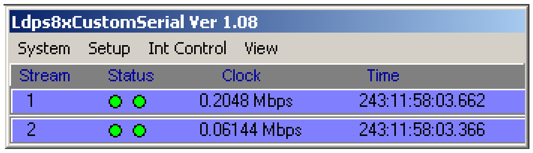
Card Status		Time
VALID	🟩	243:11:51:23.841

Param Name	Value
D__CNS_DATAVALID	1.0
D__CNS_TOY	21037883.8
D__CNS_USER_3	14.0
D__CNS_USER_4	164.0
D__CNS_USER_5	1036.0
D__CNS_USER_6	-110.0
D__CNS_USER_7	-164.0
D__CNS_USER_8	-1036.0
D__CNS_USER_9	1.0
D__CNS_USER_10	0.0
D__CNS_USER_11	0.0

Figure 3-23 Serial & Non-Serial Device Setup Windows

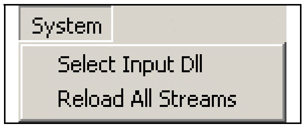
3.4.1 Custom Serial Devices

The GUI for the custom serial hardware device application is shown below. This window supports setup and configuration of all custom serial hardware devices installed in the system. The GUI has four menus including System, Setup, Int Control, and View. Hardware status information is displayed below the command menus and includes system time, data rate, and hardware status for each stream supported by the device.

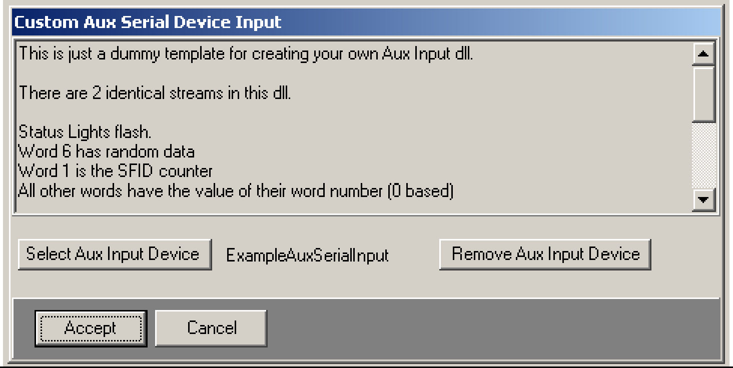


Stream	Status	Clock	Time
1	○ ○	0.2048 Mbps	243:11:58:03.662
2	○ ○	0.06144 Mbps	243:11:58:03.366

3.4.1.1 System Menu



The System Menu allows the user to select the file name of the device control DLL by clicking the **Select Input DLL** command. This will invoke the Custom Serial device DLL input window shown in Figure 3-24 below. The DLL input window defaults to a sample custom serial DLL supplied with LDPS. To select an alternate DLL, click the **Remove Aux Input Device** button and select the required DLL file name by clicking the **Select Aux Input Device** button. Selecting the DLL file name will start the DLL running. To reset the device (s) and their serial streams to a known initial state, select the **Reload All Streams** command. The DLL input window also displays a help file for the DLL. The full text of the help file for the sample Aux Serial Input DLL is shown below.



Custom Aux Serial Device Input

This is just a dummy template for creating your own Aux Input dll.

There are 2 identical streams in this dll.

Status Lights flash.
Word 6 has random data
Word 1 is the SFID counter
All other words have the value of their word number (0 based)

Figure 3-24 Custom Serial Device DLL Input Window & Help File

Sample Custom Serial DLL help file contents:

```
This is just a dummy template for creating your own Aux Input dll.

There are 2 identical streams in this dll.

Status Lights flash.
Word 6 has random data
Word 1 is the SFID counter
All other words have the value of their word number (0 based)

Words Per Frame = 256
Num Subframes = 8
SfId Word Number = 0  //0 based
SfId Start Count = 1
Frames Per Interrupt = 8

Stream 1 Minor Frame Rate = 50 hz
Stream 2 Minor Frame Rate = 15 hz
```

3.4.1.2 Setup Menu

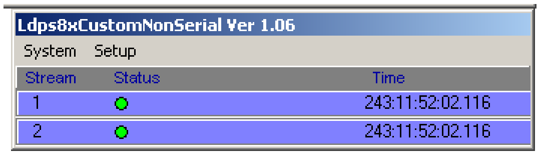
The Setup Menu allows the user to setup and configure each stream supported by the device control DLL by clicking the **Stream 1** (or Stream 2) command. This will invoke the Aux Serial Stream window shown left in Figure 3-23 on page 129. Shown in the window is device status, setup configuration, and a scrolling list of database parameters associated with the stream. The Aux Serial Stream window supplies a GUI hook for launching a custom device hardware configuration window that is developed by the user. Clicking the **User GUI** command, will launch the users' hardware setup GUI.

3.4.2 Int Control Menu

The interrupt control function allows the user to start or stop interrupt activity in unison for all custom serial cards installed in the system. The **Start Interrupts** command enables interrupts for all custom serial cards. When new data is ready from the custom serial card, they signal with an interrupt. The LDPS program picks up the data and stores/processes the data. This is also a convenient way to reload, or download, the setup data to the cards. When starting the interrupts in this manner, the cards are downloaded with their setup data first. The **Stop Interrupts** command stops interrupt activity on all custom serial cares in the system.

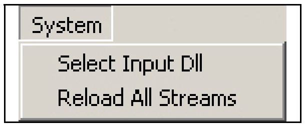
3.4.3 Custom Non-Serial Devices

The GUI for the custom non-serial hardware device application is shown below. This window supports setup and configuration of all custom non-serial hardware devices installed in the system. The GUI has two menus including System and Setup. Hardware status information is displayed below the command menus and includes system time, and hardware status for each stream supported by the device.

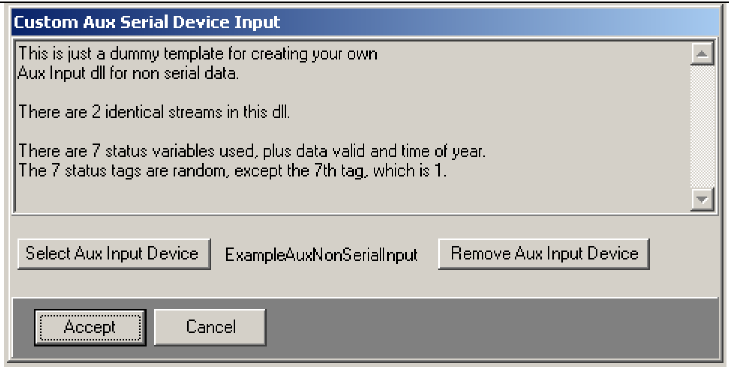


Stream	Status	Time
1	●	243:11:52:02.116
2	●	243:11:52:02.116

3.4.3.1 System Menu



The System Menu allows the user to select the file name of the non-serial device control DLL by clicking the **Select Input Dll** command. This will invoke the Custom Non-Serial device DLL input window shown in Figure 3-25 below. The DLL input window defaults to a sample custom non-serial DLL supplied with LDPS. To select an alternate DLL, click the **Remove Aux Input Device** button and select the required DLL file name by clicking the **Select Aux Input Device** button. Selecting the DLL file name will start the DLL running. To reset the device (s) and their non-serial streams to a known initial state, select the **Reload All Streams** command. The DLL input window also displays a help file for the DLL. The full text of the help file for the sample Aux Non-Serial Input DLL is shown below.



This is just a dummy template for creating your own
Aux Input dll for non serial data.

There are 2 identical streams in this dll.

There are 7 status variables used, plus data valid and time of year.
The 7 status tags are random, except the 7th tag, which is 1.

Figure 3-25 Custom Non-Serial Device DLL Input Window & Help File

Sample Custom Non-Serial DLL help file contents:

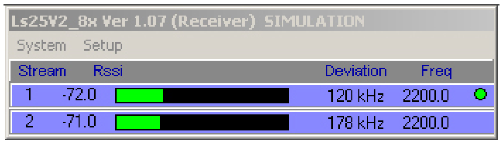
```
This is just a dummy template for creating your own
Aux Input dll for non serial data.

There are 2 identical streams in this dll.

There are 7 status variables used, plus data valid and time of year.
```

```
The 7 status tags are random, except the 7th tag, which is 1.
```

3.4.3.2 Setup Menu

The setup menu allows the user to setup and configure each stream supported by the device control DLL by clicking the **Stream 1** (or Stream 2) command. This will invoke the Aux Non-Serial Stream window shown right in Figure 3-23 on page 129. Shown in the window is a scrolling list of database parameters associated with the stream. The Aux Non-Serial Stream window supplies a GUI hook for launching a custom device hardware configuration window that is developed by the user. Clicking the **User GUI** command will launch the users' hardware setup GUI.

3.5 LS-25 Telemetry Receiver Setup GUI



Stream	Rssi	Deviation	Freq
1	-72.0	120 kHz	2200.0
2	-71.0	178 kHz	2200.0

The telemetry receiver setup GUI shown right is a small window launched from the main application window on the Server. The setup display GUI shown in Figure 3-26 is launched by the user by selecting *Setup → Stream 1* on the main application window, or when a project is loaded. This GUI controls all LS-25 receiver cards installed in the system. The receiver setup GUI has three major parts: The caption area, the menu area, and the display/status area.

The caption area (top of the window) displays the name of the window, and if a simulation is running, the word "SIMULATION" will be flashing. Note there is no normal window closing mechanism in the caption area (the X normally found in the upper right corner of windows). This is because the window must be active in order to gather data from the multifunction card, and to control it.

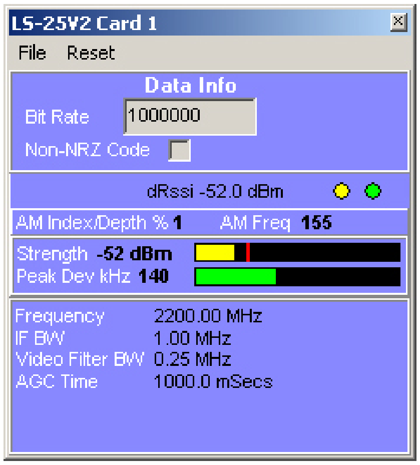
Below the caption area and setup menu is the status display for all LS-25 cards installed in the system. For each stream, the status display shows the Received Signal Strength (RSSI) numerical value along with a horizontal bar graph of the same. Also displayed is the peak FM deviation and receiver frequency.

3.5.1 System Menu

The system menu has a single command: Flash Board ID Leds. To identify a specific LS-25V2 card(s) in a system with multiple cards, the user may invoke the **Flash Board ID Leds** command, and select the desired stream number from the list of available streams. Remember that each LS-25V2 card in the system has a unique and specific stream number assigned to it.

3.5.2 Setup Menu

The setup menu allows the user to configure each stream in the system and the LS-25V2 hardware associated with each stream. Selecting a specific stream from the list of possible streams will invoke the configuration setup screen shown right.

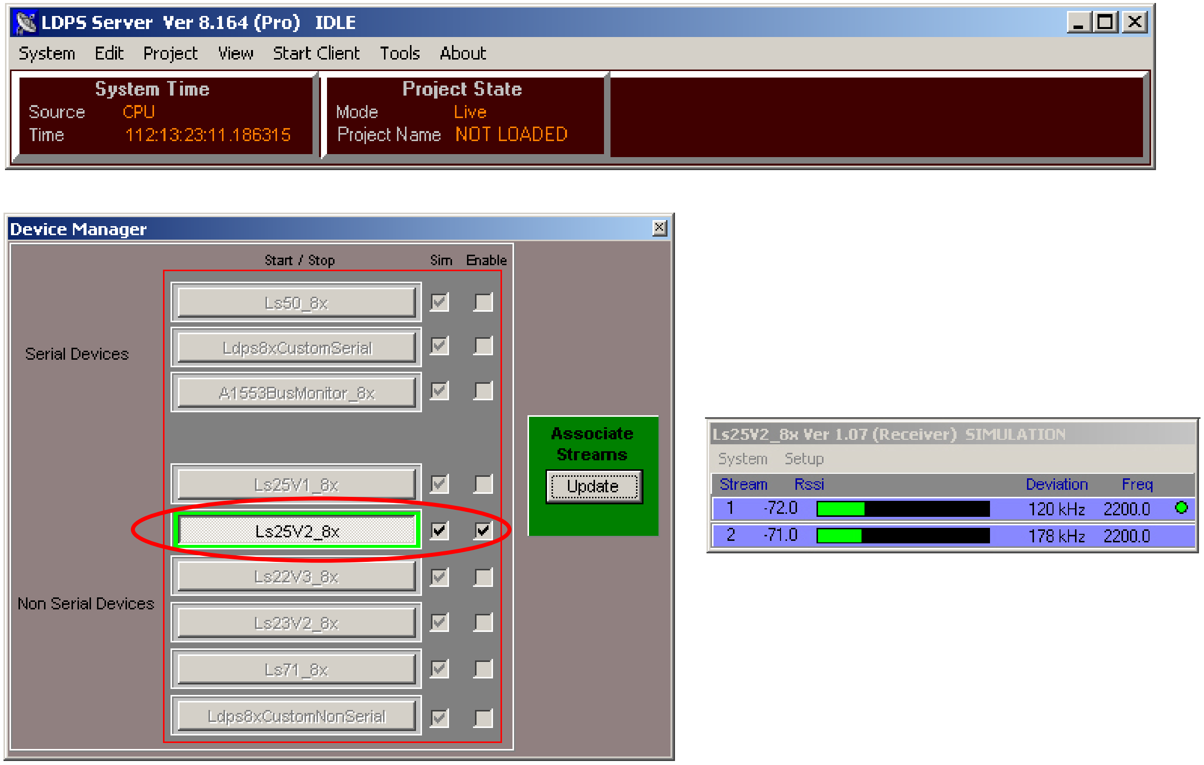


Bit Rate 1000000
Non-NRZ Code □
dRssi -52.0 dBm
AM Index/Depth % 1 AM Freq 155
Strength -52 dBm
Peak Dev kHz 140
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Figure 3-26 Setup Display for the LS-25-D/LS-25

To initially configure the LS-25-D/LS-25, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select "Devices" and then "Manage" *(System→ Devices→ Manage)*
2. From the System Manager shown in Figure 3-27 below left, select the "Enable" check box next to the LS25V2 button. The "Ls25V2_8x" button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the "Sim" check box next to the "Enable" check box. Checking this box allows the LDPS application to operate when a LS-25 board is not installed in the system.
3. From the System Manager, click the "Ls25V2_8x" button. This will launch the "Ls25V2_8x (Receiver)" display shown in Figure 3-27 below right. Note that the red rectangle around the button has changed to green indicating that the application is now running.
4. To setup and configure the LS-25-D and LS-25 cards, follow the procedures outlined in paragraphs 3.5.3.



System Time
Source CPU
Time 112:13:23:11.186315

Project State
Mode Live
Project Name NOT LOADED

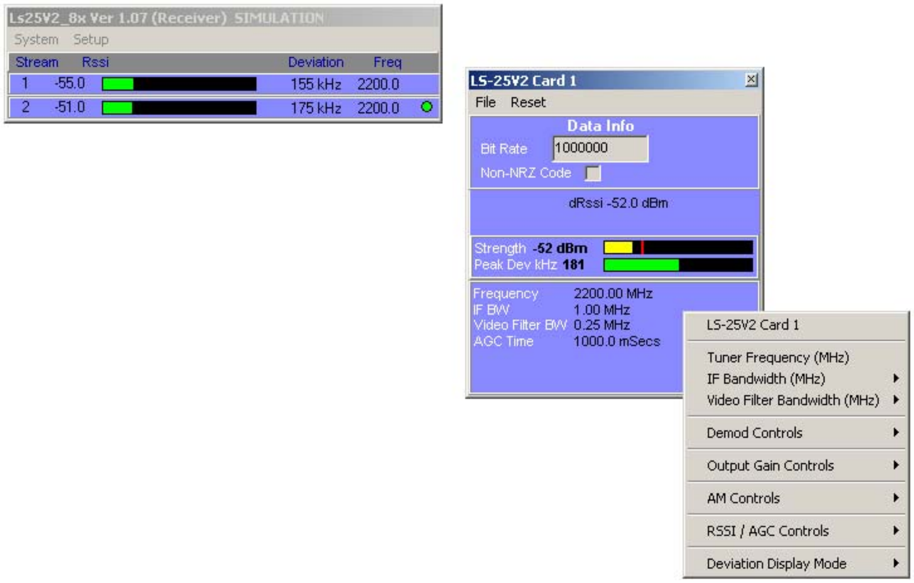
	Start / Stop	Sim	Enable
Serial Devices	Ls50_8x	☑	□
	Ldps8xCustomSerial	☑	□
	A1553BusMonitor_8x	☑	□
Non Serial Devices	Ls25V1_8x	☑	□
	Ls25V2_8x	☑	☑
	Ls22V3_8x	☑	□
	Ls23V2_8x	☑	□
	Ls71_8x	☑	□
	Ldps8xCustomNonSerial	☑	□

Stream	Rssi	Deviation	Freq
1	-72.0	120 kHz	2200.0
2	-71.0	178 kHz	2200.0

Figure 3-27 Server Application Windows (LS-25 Enabled)

3.5.3 Configuring The LS-25-D/LS-25 Hardware

From the “Ls25V2_8x (Receiver)” display shown below upper left in Figure 3-28, click “Setup” and then “Stream 1” *(Setup →Stream 1).*



Stream	Rssi	Deviation	Freq
1	-55.0	155 kHz	2200.0
2	-51.0	175 kHz	2200.0

Bit Rate 1000000
Non-NRZ Code ☐
dRssi -52.0 dBm
Strength -52 dBm
Peak Dev kHz 181
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Figure 3-28 Configuration Menus/Controls for the LS-25-D/LS-25

The “LS-25V2 Card 1” display shown center in Figure 3-28 above is divided into several regions. The number of, and function of each region is dependent on the various modes the cards can be configured for.

To invoke the controls for the display, simply place the mouse curser in the center region and right click. The resulting menus are shown in Figure 3-29 right are discussed in detail in the following paragraphs.

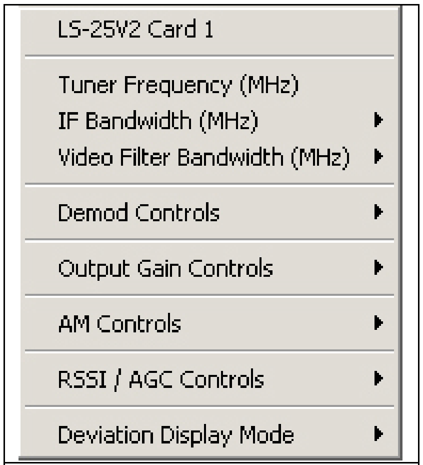


Figure 3-29 The LS-25-D/LS-25 Configuration Menu

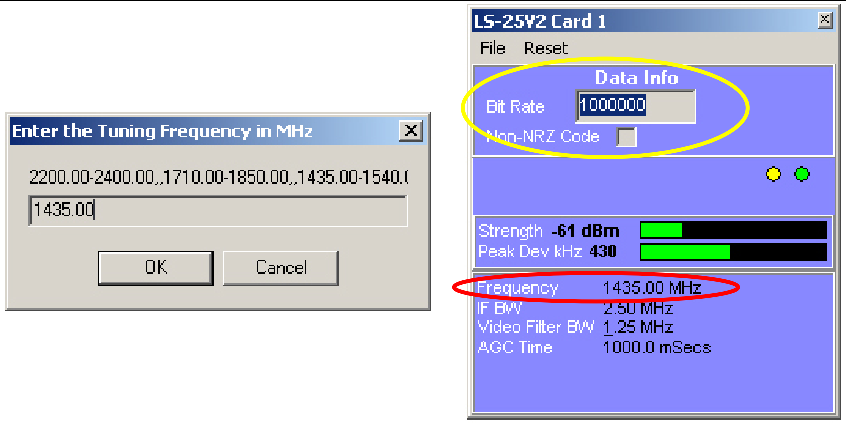
3.5.3.1 Tuner Frequency

The Version 2 Lumistar LS-25-D Multi-band RF Down-converter and the LS-25 Multi-band FM Receiver concurrently support AM & FM demodulation in three the following five possible frequency bands. *Note that up to three (3) of these bands may be specified at the time of order.*

2185.5 - 2485.5 MHz (NATO E-Band)
2200.5 - 2399.5 MHz (S-Band)
1710.5 - 1849.5 MHz (Upper L-Band)
1435.5 - 1539.5 MHz (Lower L-Band)
215.5 - 319.5 MHz (P-Band)

To select a receive frequency, click on the *"Tuner Frequency (MHz)"* menu item. Enter the frequency in the pop-up dialog box shown in Figure 3-30 below left and click OK. The updated frequency will be displayed as shown in Figure 3-30 below right (red oval). Note that the supported tuner resolution is 50 KHz, and that entered values will be rounded off the nearest 50 KHz value in the display. Frequency values outside of the ranges specified above will result in an error message with no change in frequency (the default frequency is 2200.00 MHz).

Also note that the user may enter the data stream bit rate in bits/second and indicate if the stream encoding is non-NRZ (bi-phase, miller, etc.) as shown in the yellow oval in the figure below.



2200.00-2400.00,,1710.00-1850.00,,1435.00-1540.(

1435.00

Bit Rate 1000000

Non-NRZ Code ☐

Strength -61 dBm

Peak Dev kHz 430

Frequency 1435.00 MHz

IF BW 2.50 MHz

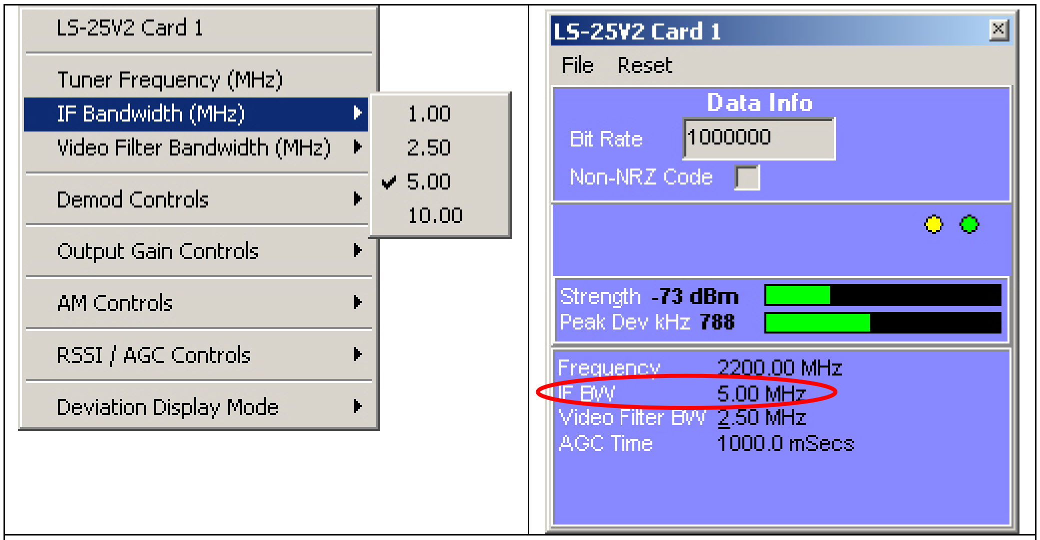
Video Filter BW 1.25 MHz

AGC Time 1000.0 mSecs

Figure 3-30 Tuning Frequency Display

3.5.3.2 IF Bandwidth

The Version 2 Lumistar LS-25-D Multi-band RF Down-converter and the LS-25 Multi-band FM Receiver are factory configured to support up to twelve (12) separate 2nd IF bandwidths. Standard bandwidths include: 500kHz, 1MHz, 1.5MHz, 2.5MHz, 3.5MHz, 4MHz, 6MHz, 8MHz, 10MHz, 12MHz, 16MHz, and 20MHz. At the factory the selected 2nd IF bandwidth values are programmed into a configuration PROM and are used by the LDPS application to populate frequency values in the pop-up list box shown in Figure 3-31 below left. In the example shown, the unit is factory configured with 2nd IF bandwidths of 1, 2.5, 5, and 10 MHz. The actual 2nd IF bandwidths the individual user will see when setting the IF bandwidth are likely to be different than those show here. Once selected, the 2nd IF bandwidth will be displayed as shown in the figure below right (red oval).

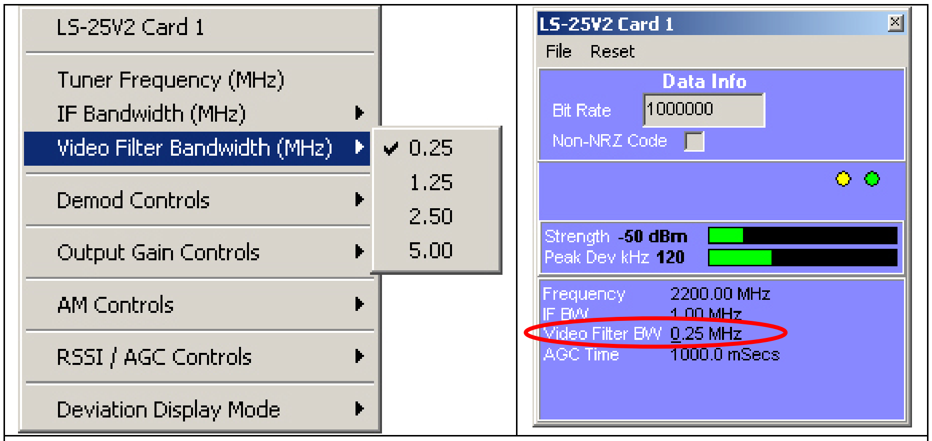


Bit Rate 1000000
Non-NRZ Code □
Strength -73 dBm
Peak Dev kHz 788
Frequency 2200.00 MHz
IF BW 5.00 MHz
Video Filter BW 2.50 MHz
AGC Time 1000.0 mSecs

Figure 3-31 2nd IF Bandwidth (MHz)

3.5.3.3 Video Filter Bandwidth

The Version 2 Lumistar LS-25-D Multi-band RF Down-converter and the LS-25 Multi-band FM Receiver are factory configured to support up to twelve (12) separate Video Filter bandwidths. Standard bandwidths include: 250kHz, 500kHz, 750kHz, 1.25MHz, 1.75MHz, 2MHz, 3MHz, 4MHz, 5MHz, 6MHz, 8MHz, and 10MHz. At the factory the selected Video Filter bandwidth values are programmed into a configuration PROM and are used by the LDPS application to populate frequency values in the pop-up list box shown in Figure 3-32 below left. In the example shown, the unit is factory configured with Video Filter bandwidths of 0.25, 1.25, 2.50, and 5 MHz. The actual Video Filter bandwidths the individual user will see when setting the Video Filter bandwidth are likely to be different than those show here. Once selected, the Video Filter bandwidth will be displayed as shown in the figure below right (red oval).



Bit Rate 1000000
Non-NRZ Code □
Strength -50 dBm
Peak Dev kHz 120
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Figure 3-32 Video Filter Bandwidth (MHz)

3.5.3.4 Demodulation Control

The LS-25V2 Demodulation Control has three sub-modes: “External Demod Source”, “AC Couple TLM2”, and “Data Polarity Inverted’ as shown in Figure 3-33 below.

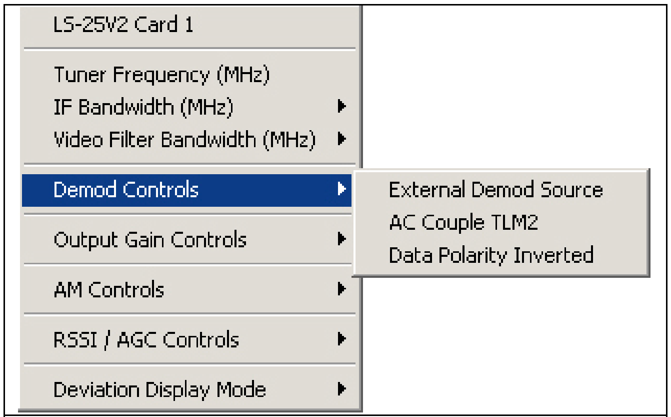
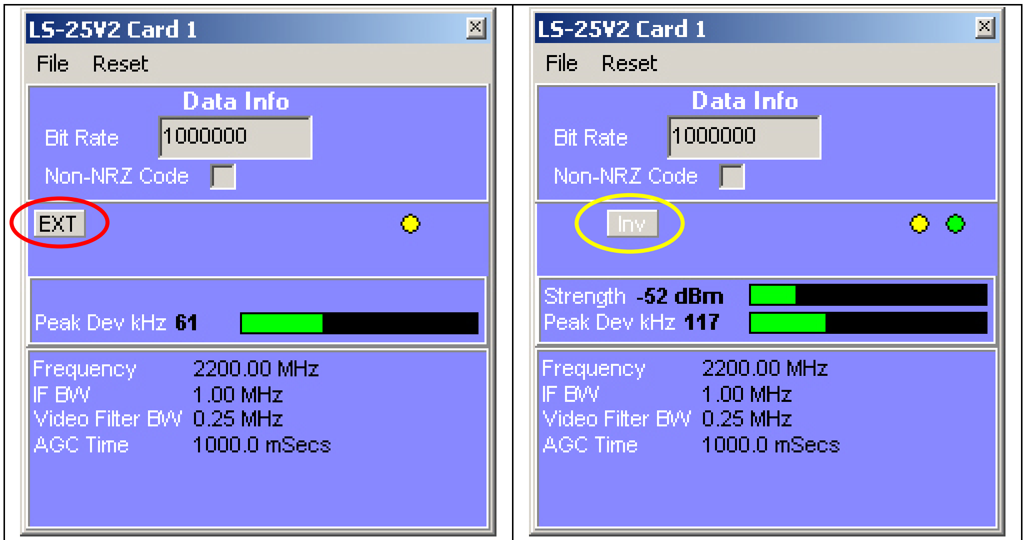


Figure 3-33 Demodulation Controls

The “External Demod Source” sub-mode enables the LS-25 FM Discriminator and Video Filter to be driven by an external input source. The selection of this sub-mode is indicated in the display as shown in Figure 3-34 below left (red oval).



Bit Rate 1000000
Non-NRZ Code □
Peak Dev kHz 61
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

“External Demod Source”

Bit Rate 1000000
Non-NRZ Code □
Strength -52 dBm
Peak Dev kHz 117
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

“Data Polarity Inverted”

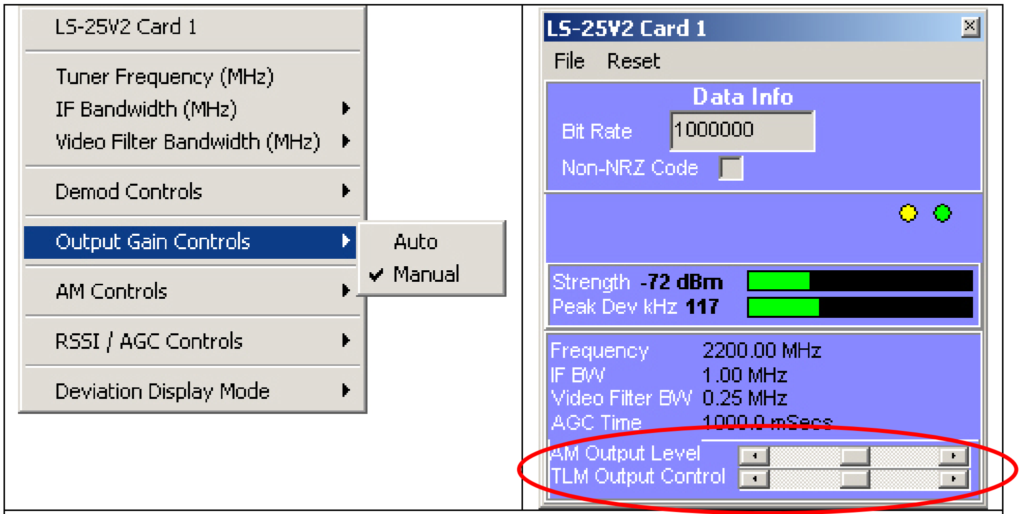
Figure 3-34 Demodulation Controls & Resulting Displays

The “AC Couple TLM2” sub-mode allows the user to select the AC coupling configuration for the TLM2 output. If this sub-mode is not selected (no check mark in the menu), then the coupling mode for the TLM2 output will default to DC coupling.

The “Data Polarity Inverted” sub-mode allows the user to invert the logic polarity of the PCM data output. The selection of this sub-mode is indicated in the display as shown in Figure 3-34 above right (yellow oval).

3.5.3.5 Output Gain Controls

The LS-25V2 Output Gain Control has two sub-modes: “Auto”, and “Manual” as shown in Figure 3-35 below left. When the “Manual” sub-mode is selected, two additional slider controls will appear on the display as shown below right (red oval). The “AM Output Level” slider control allows the user to manually alter the output voltage level of the AM output on the LS-25V2. The “TLM Output Level” slider control allows the user to manually alter the output voltage level of the TLM1 and TLM2 output signals produced by the LS-25V2. In either case, no additional feedback is provided in the display as the sliders are adjusted. The actual voltage levels of the AM and TLM signals will need to be measured via some form of external instrumentation (volt meter, oscilloscope, etc) as each slider is adjusted.

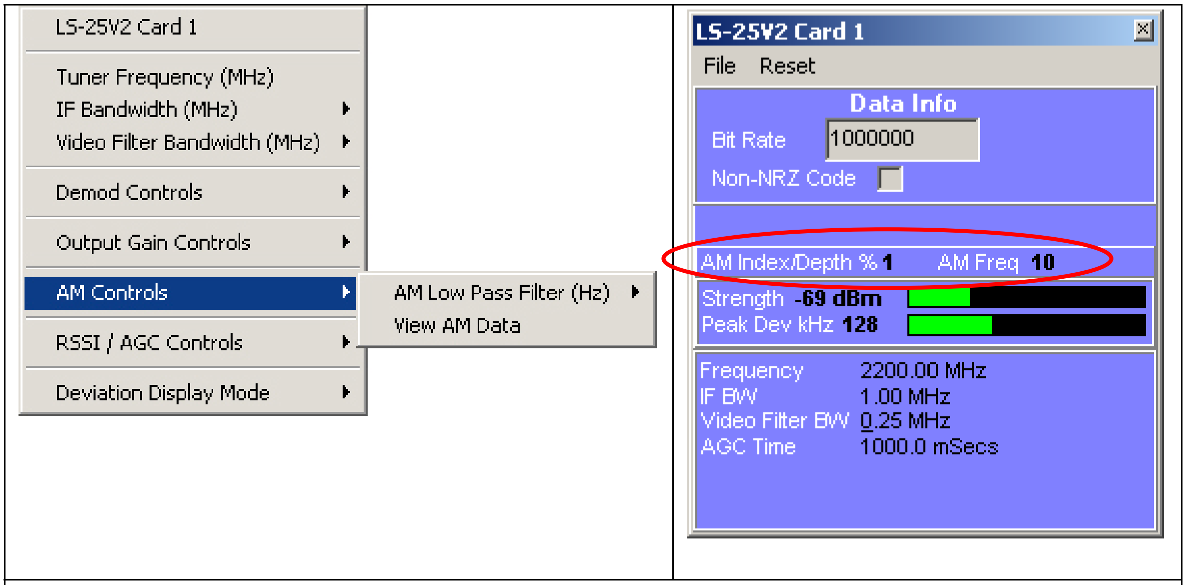


Bit Rate 1000000
Strength -72 dBm
Peak Dev kHz 117
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Figure 3-35 Output Gain Controls & Resulting Display

3.5.3.6 AM Controls

The LS-25V2 AM Control has two sub-modes: “AM Low Pass Filter (Hz)”, and “View AM Data” as shown in Figure 3-36 below left. When the “View AM Data” sub-mode is selected, two additional data displays will appear on the display as shown below right (red oval). The “AM Index Depth%” is the amplitude modulation index detected in the post-processing of the AM demodulation. The “AM Freq” is the instantaneous frequency value of the AM demodulated signal (Hz).



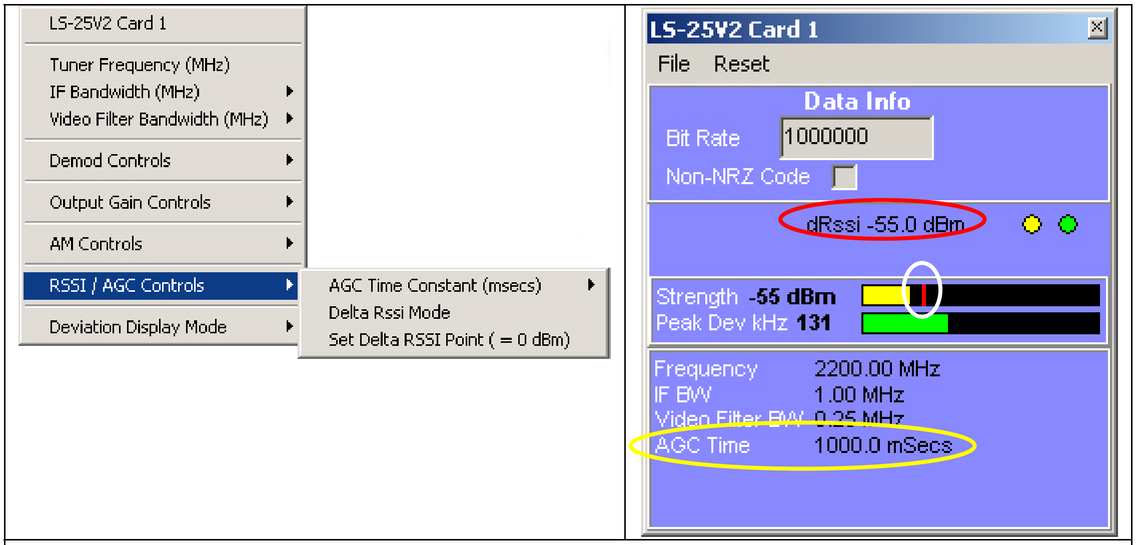
Bit Rate 1000000
Non-NRZ Code □
AM Index/Depth % **1** AM Freq **10**
Strength **-69 dBm**
Peak Dev kHz **128**
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Figure 3-36 AM Controls

The "AM Low Pass Filter (Hz)" sub-mode allows the user to select one of four low-pass filters on the AM output including; 50, 500, 5000, or 50,000 Hz. Note, the 50 KHz selection is the same as low-pass filter bypass.

3.5.3.7 RSSI/AGC Controls

The LS-25V2 RSSI/AGC Control has three sub-modes: "AGC Time Constant", "Delta RSSI Mode", and "Set Delta RSSI Point" as shown in Figure 3-37 below left. The "AGC Time Constant" sub-mode allows the user to select one of four possible AGC time constants including; 33, 100, 330, and 1000 ms. The selected time constant is displayed as shown in Figure 3-37 below right (yellow oval).



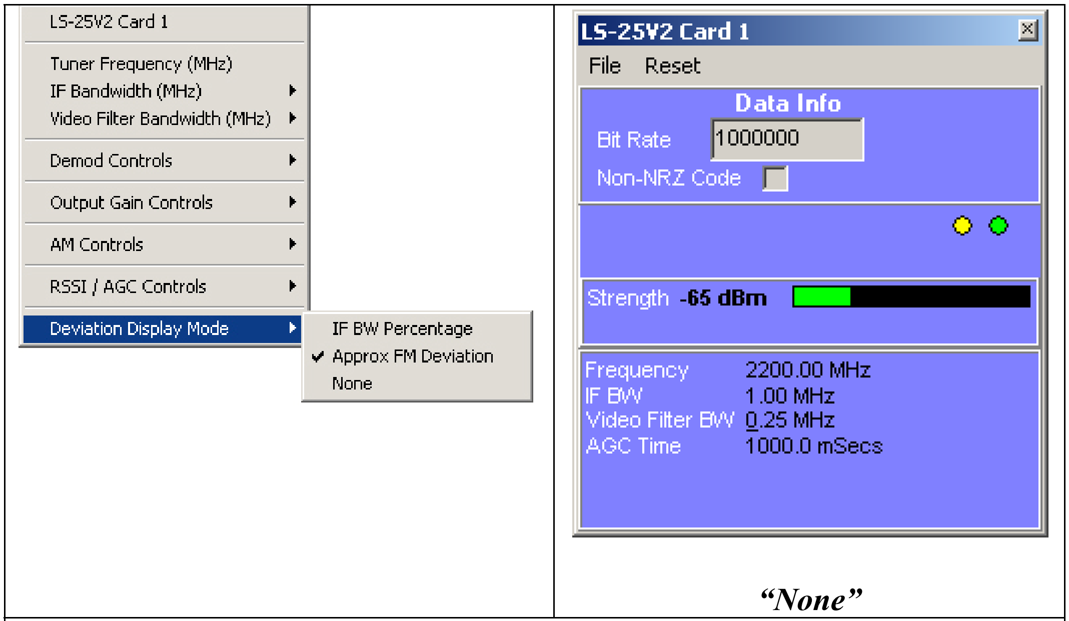
Bit Rate 1000000
Non-NRZ Code □
dRssi -55.0 dBm
Strength -55 dBm
Peak Dev kHz 131
Frequency 2200.00 MHz
IF BW 1.00 MHz
Video Filter BW 0.25 MHz
AGC Time 1000.0 mSecs

Figure 3-37 RSSI/AGC Controls & Resulting Display

The "Delta RSSI Mode", and "Set Delta RSSI Point" sub-modes are used in concert with each other. The "Set Delta RSSI Point" sub-mode initiates the acquisition of the instantaneous RF input power (dB) or signal strength level at the input to connector J2. This *snapshot* captures and establishes an absolute input power reference level that is subsequently compared continuously with the instantaneous RF input power level. The difference, or delta between the reference and instantaneous levels is displayed in two ways as shown in Figure 3-37 above right. The numerical value of the delta is shown as indicated by the red oval. The reference level is represented by a vertical red line shown in the white oval portion of the signal strength bar graph.

3.5.3.8 Deviation Display Modes

The LS-25V2 Deviation Display Mode Control has three sub-modes: "IF BW Percentage", "Approx FM Deviation", and "None" as shown in Figure 3-38 below left. The "None" mode display is shown in Figure 3-38 below right.



Bit Rate 1000000
Non-NRZ Code □

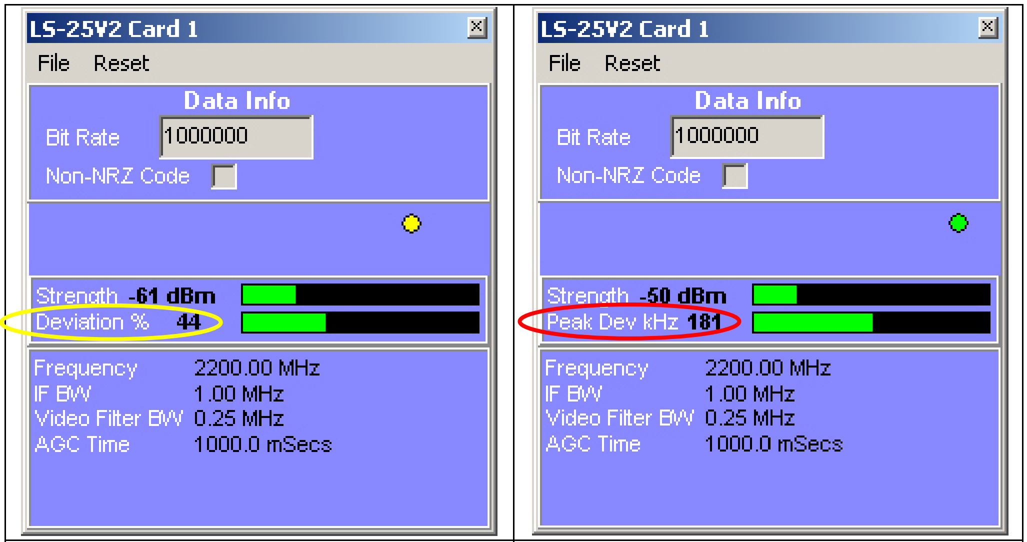
Strength -65 dBm

Frequency	2200.00 MHz
IF BW	1.00 MHz
Video Filter BW	0.25 MHz
AGC Time	1000.0 mSecs

"None"

Figure 3-38 Deviation Display Modes

The "Approx FM Deviation" sub-mode is shown in Figure 3-39 below right and indicates the peak FM deviation of the signal (red oval). The "IF BW Percentage" sub-mode is shown in Figure 3-39 below left and indicates the deviation percentage of the signal (yellow oval).



Bit Rate 1000000
Non-NRZ Code □

Strength -61 dBm
Deviation % 44

Frequency	2200.00 MHz
IF BW	1.00 MHz
Video Filter BW	0.25 MHz
AGC Time	1000.0 mSecs

Bit Rate 1000000
Non-NRZ Code □

Strength -50 dBm
Peak Dev kHz 181

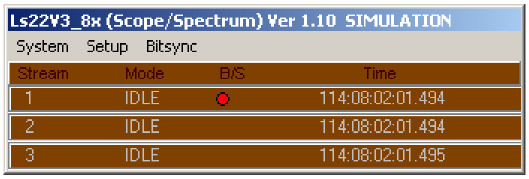
Frequency	2200.00 MHz
IF BW	1.00 MHz
Video Filter BW	0.25 MHz
AGC Time	1000.0 mSecs

"Deviation Percentage"	***"Peak Deviation"***

Figure 3-39 Additional Deviation Displays

3.6 LS-22 Spectral Display

The Scope/Spectrum Display setup GUI shown right is a small window launched from the main application window on the Server. It is launched by the user by selecting *Setup → Card Input 1* on the main application window, or when a project is loaded. This GUI controls all LS-22 Scope/Spectrum Display cards installed in the system. The Scope/Spectrum Display setup GUI has three major parts: The caption area, the menu area, and the display/status area.



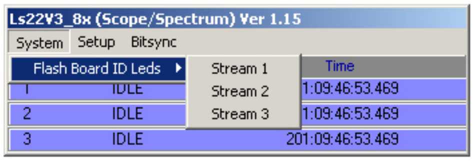
Stream	Mode	B/S	Time
1	IDLE	●	114:08:02:01.494
2	IDLE		114:08:02:01.494
3	IDLE		114:08:02:01.495

The caption area (top of the window) displays the name of the window, and if a simulation is running, the word "SIMULATION" will be flashing. Note there is no normal window closing mechanism in the caption area (the X normally found in the upper right corner of windows). This is because the window must be active in order to gather data from the multifunction card, and to control it.

Below the caption area and setup menu is the status display for all LS-22 cards installed in the system. For each stream, the status display shows the system time, Bit Sync lock status, and display mode (spectral/time domain).

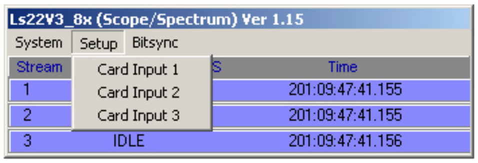
3.6.1 System

The system menu has a single command: Flash Board ID Leds. To identify a specific LS-22 card(s) in a system with multiple cards, the user may invoke the **Flash Board ID Leds** command, and select the desired stream number from the list of available streams. Remember that each LS-22 card in the system has a unique and specific stream number assigned to it.



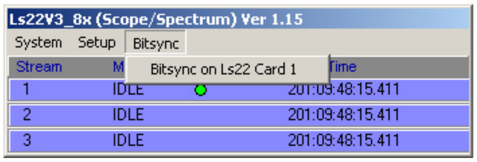
3.6.2 Setup

The setup menu allows the user to configure each of the three inputs on each LS-22 card(s) installed in the system. Selecting a specific card input from the list of possible card inputs will invoke the configuration setup screen shown in Figure 3-40 on page 144. The example shown right corresponds to a system with <u>one</u> LS-22 card installed. This single card will have inputs 1,2, and 3 as shown. For a system with <u>two</u> LS-22 cards, the first card will have inputs 1,2, and 3, and the second card will have inputs 4,5, and 6. The third card in a system with three LS-22 cards will have inputs 7,8, and 9, and so on.

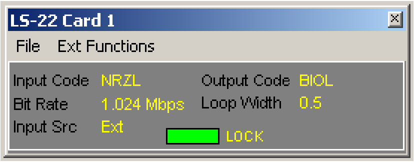


3.6.3 Bitsync

Optionally, the Lumistar **LS-40-DB10** or **LS-40-DB20** Bit Sync Daughterboard may be installed on the LS-22-SE Spectral & Oscilloscope Display Card. By invoking the **Bitsync on LS22 Card N** (where "N" is the card number) command the user may produce the status and configuration setup display shown below right.



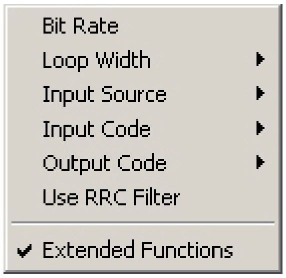
Stream	M	Time
1	IDLE	201:09:48:15.411
2	IDLE	201:09:48:15.411
3	IDLE	201:09:48:15.411

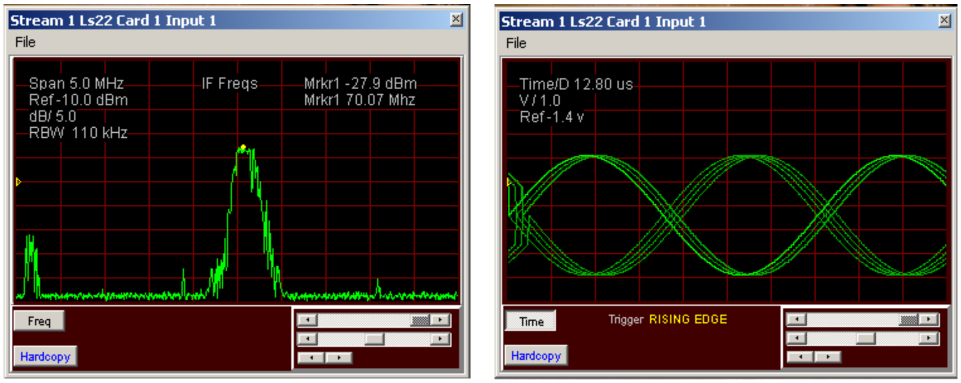


Input Code NRZL
Bit Rate 1.024 Mbps
Input Src Ext
Output Code BIOL
Loop Width 0.5
LOCK

The LS-22 Card N status and configuration setup display has a **File** menu, and an **External Functions** command. The external functions command invokes a status and configuration display identical to the one shown in the upper left of Figure 3-12 on page 108. To invoke the bit sync setup commands shown below right, place the cursor in the lower portion of the status and configuration setup display and right click. Each of the bit sync setup commands are described in more detail in the paragraphs listed below left.

Command	Paragraph
Bit Rate	3.2.1.3.1 on page 106
Loop Width	3.2.1.3.4on page 106
Input Source	3.2.1.3.2 on page 106
Input Code	3.2.1.3.3on page 106
Output Code	3.2.1.3.6 on page 107
Use RRC Filter	3.2.1.3.5 on page 107



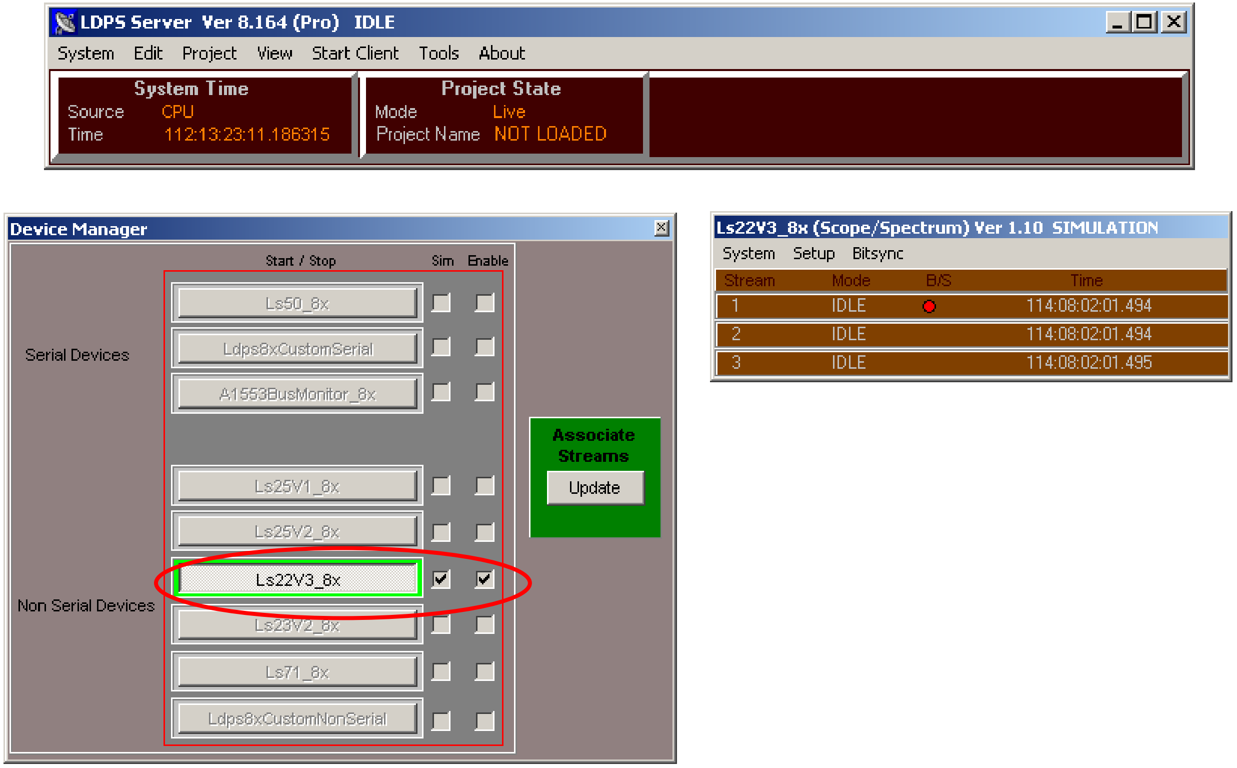


Spectral Display *Time Domain Display (eye pattern)*

Figure 3-40 Displays for the LS-22-SE

To initially configure the LS-22-SE, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select “Devices” and then “Manage” *(System→ Devices→ Manage)*
2. From the System Manager shown below left, select the “Enable” check box. The “Ls22Vx_8x” button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the “Sim” check box next to the “Enable” check box. Checking this box allows the LDPS application to operate when a LS-22-SE board is not installed in the system.
3. From the System Manager, click the “Ls22Vx_8x” button. This will launch the “Ls22Vx_8x (Scope/Spectrum)” display shown below right.
4. For the spectrum and time-domain displays, follow the procedures outlined in paragraphs 3.6.4 and 3.6.5.



LDPS Server Ver 8.164 (Pro) IDLE

System Time	
Source	CPU
Time	112:13:23:11.186315

Project State	
Mode	Live
Project Name	NOT LOADED

Device Manager

	Start / Stop	Sim	Enable
Serial Devices	Ls50_8x	□	□
	Ldps8xCustomSerial	□	□
	A1553BusMonitor_8x	□	□
Non Serial Devices	Ls25V1_8x	□	□
	Ls25V2_8x	□	□
	Ls22V3_8x	☑	☑
	Ls23V2_8x	□	□
	Ls71_8x	□	□
	Ldps8xCustomNonSerial	□	□

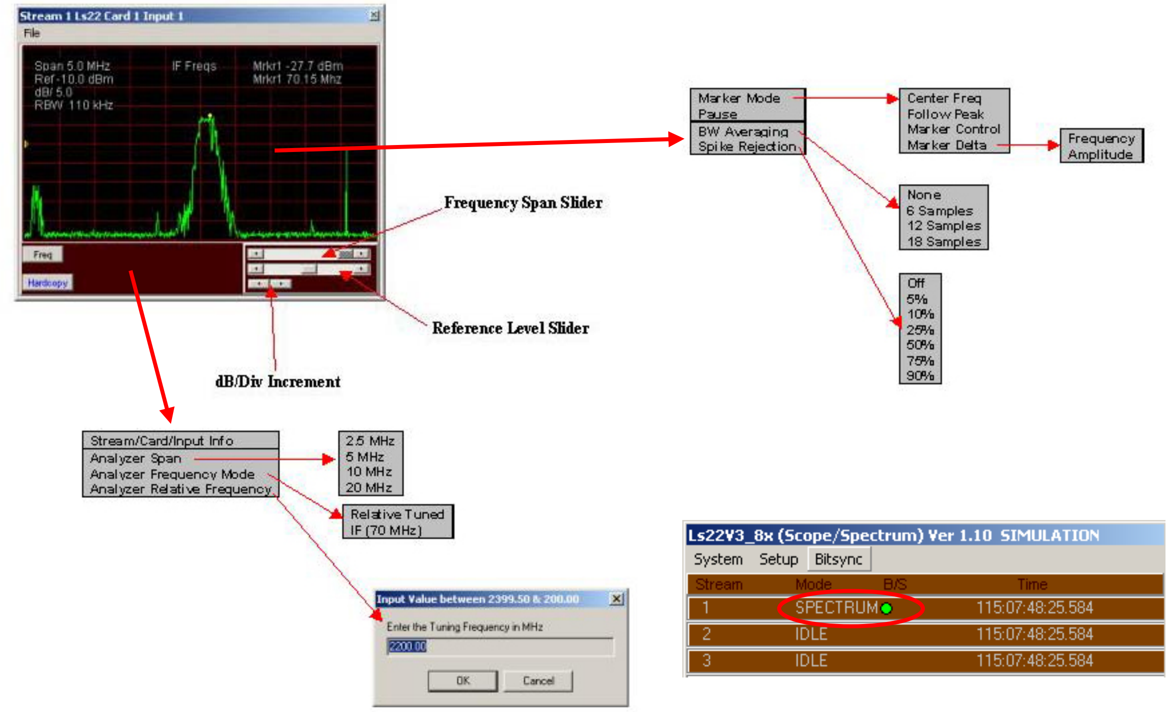
Ls22V3_8x (Scope/Spectrum) Ver 1.10 SIMULATION

Stream	Mode	B/S	Time
1	IDLE	●	114:08:02:01.494
2	IDLE		114:08:02:01.494
3	IDLE		114:08:02:01.495

Figure 3-41 Server Application Windows (LS-22 Enabled)

3.6.4 The Spectral Display

From the “Ls22Vx_8x (Scope/Spectrum)” display shown right in Figure 3-41 on page 145, click “Setup” and then “Card Input 1” *(Setup →Card Input 1)*. The LS-22-SE Time Domain window shown in the upper left of Figure 3-49 on page 154 will be launched. Click on the “Time” button of the Time Domain display to toggle into the Spectral Display mode. The resulting window is shown in Figure 3-42 below upper left. Also notice that the “Mode” of Card Input 1 has changed from “IDLE” to “SPECTRUM” as shown in Figure 3-42 below lower right. The other inputs of the LS-22-SE card are configured in a similar manner *(Setup →Card Input 2, 3, etc).*


Stream 1 Ls22 Card 1 Input 1

Span 5.0 MHz
Ref -10.0 dBm
dB/ 5.0
RBW 110 kHz

IF Freqs

Mrkr1 -27.7 dBm
Mrkr1 70.15 Mhz

Input Value between 2399.50 & 200.00

Enter the Tuning Frequency in MHz

2200.00

Ls22V3_8x (Scope/Spectrum) Ver 1.10 SIMULATION

System Setup Bitsync

Stream	Mode	B/S	Time
1	SPECTRUM ●		115:07:48:25.584
2	IDLE		115:07:48:25.584
3	IDLE		115:07:48:25.584

Figure 3-42 Spectral Display Menus/Controls

The “Ls22Vx_8x (Scope/Spectrum)” display shown in Figure 3-43 below is divided into several regions. In the upper portion of the window is the graphical data display area where the spectrum graphs are generated. Also in the upper left portion of the spectrum display as shown in the red rectangle are the frequency span, the reference level (dB), dB/division, and the resolution bandwidth. These parameters are controlled in the lower right region below the graphical display has slider or knob controls as shown in the figure. Addition information associated with frequency and amplitude markers are shown in the upper right portion of the spectrum display.

To invoke the controls for either the upper or lower portions of the display, simply place the mouse curser in the region and right click. The resulting menus are shown in Figure 3-42 above and are discussed in detail in the following paragraphs.

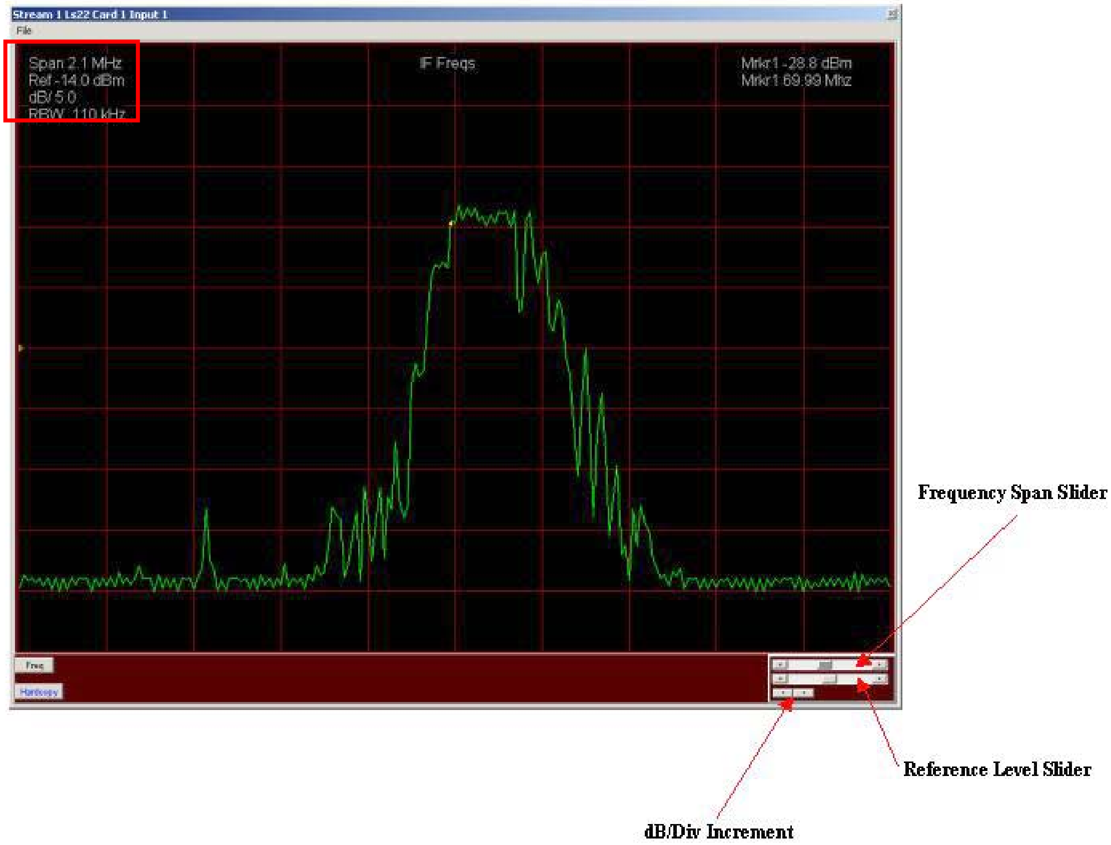
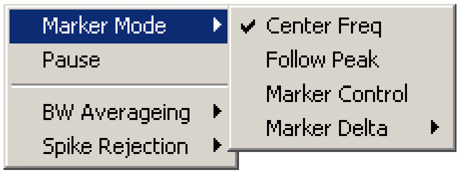


Figure 3-43 The Spectrum Display

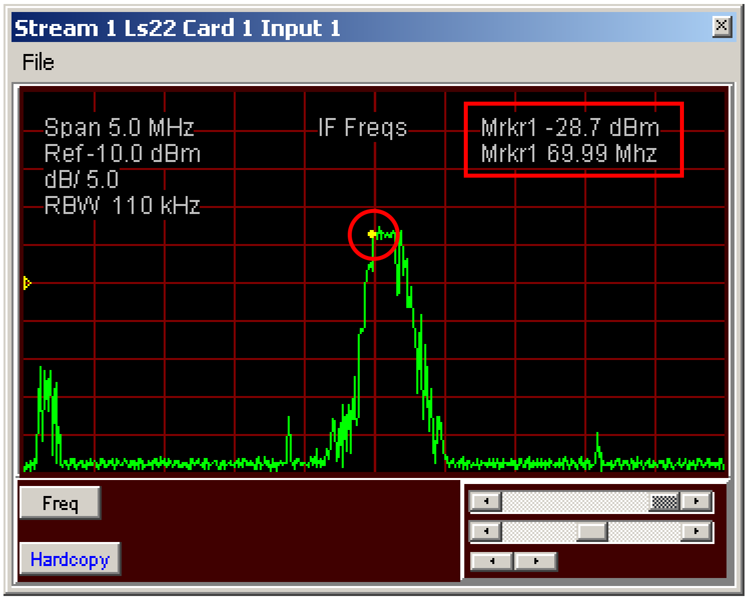
3.6.4.1 Marker Mode

Select the Marker Mode by right clicking in the upper portion of the spectrum display. The Marker Mode has four sub-modes: Center Frequency, Follow Peak, Marker Control, and Marker Delta. Each in turn is discussed in the following paragraphs.



3.6.4.1.1 Center Frequency

The Center Frequency marker mode places the marker (shown in the red circle in the figure below) at the center of the span of frequencies shown in the display. Note also that the markers amplitude and frequency are shown on the display as indicated in the red square in the figure below.


Stream 1 Ls22 Card 1 Input 1

File

Span 5.0 MHz
Ref -10.0 dBm
dB/ 5.0
RBW 110 kHz

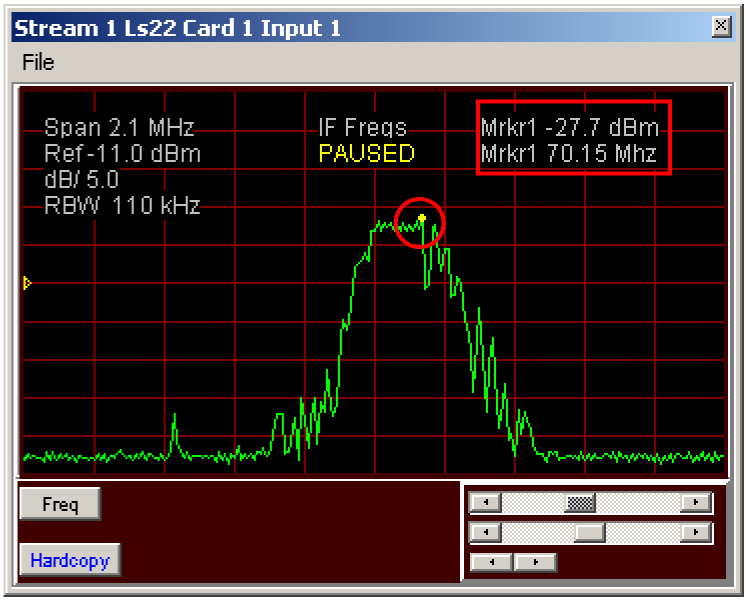
IF Freqs

Mrkr1 -28.7 dBm
Mrkr1 69.99 Mhz

Figure 3-44 Marker Mode → Center Frequency

3.6.4.1.2 Follow Peak

The Follow Peak marker mode places the marker (shown in the red circle in the figure below) at the peak amplitude value of the trace shown in the display. As the peak value changes in both amplitude and frequency, the marker will move up and down, left and right to follow. Also note as before that the markers amplitude and frequency are shown on the display as indicated in the red square in the figure below.


Stream 1 Ls22 Card 1 Input 1

File

Span 2.1 MHz
Ref -11.0 dBm
dB/ 5.0
RBW 110 kHz

IF Freqs
PAUSED

Mrkr1 -27.7 dBm
Mrkr1 70.15 Mhz

Figure 3-45 Marker Mode → Follow Peak

3.6.4.1.3 Marker Control

The Marker Control mode places the marker (shown in the red circle in the figure below) at a specific frequency point value on the trace shown in the display. As the amplitude value at that frequency changes, the marker will move up and down. Also note the location of the frequency control shown on the display as indicated in the red square in the figure below. Clicking on the right pointing arrow advances the frequency of the marker. Clicking on the left pointing arrow decrease the frequency of the marker.

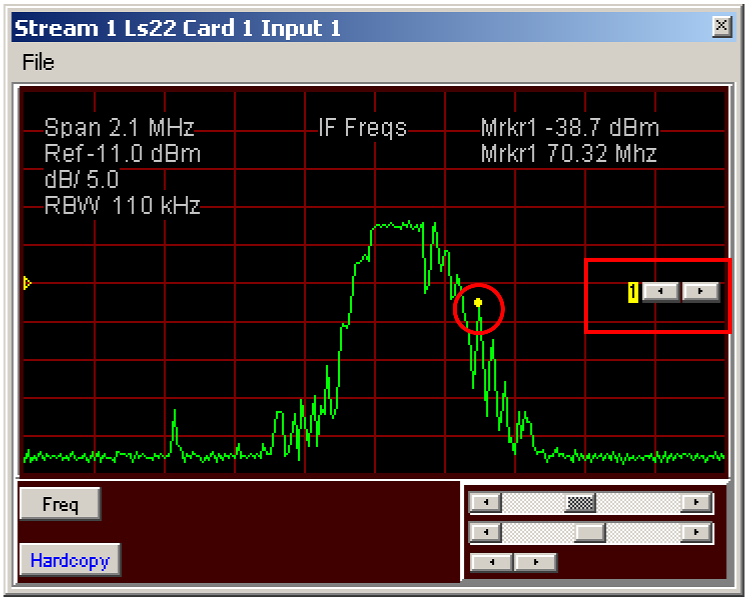
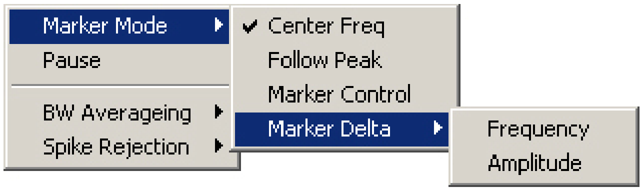
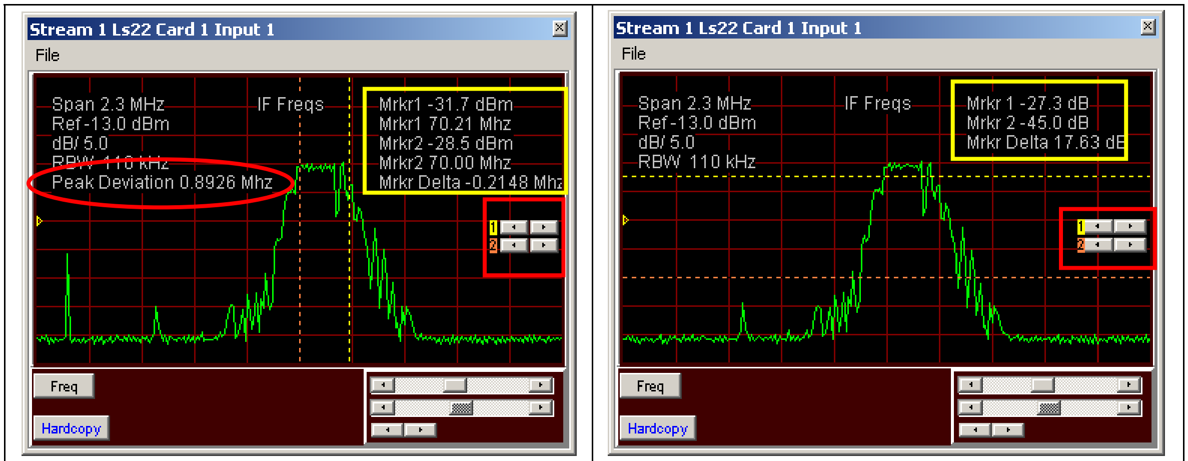


Figure 3-46 Marker Mode → Marker Control

3.6.4.1.4 Marker Delta

The Marker Delta mode has two sub-modes: Frequency and Amplitude. The Frequency Marker Delta mode is shown left in Figure 3-47 below and is represented by two vertical dotted lines. Each marker line advances or recedes in frequency via the control shown on the display as indicated in the red square in the figure below. Note that the markers amplitude, frequency, and frequency delta are shown on the display as indicated in the yellow square in the figure below. Also note the "Peak Deviation" parameter shown in the red oval on the left side of the spectral display. This parameter only has meaning when the spectrum of a NRZ/FM signal is being displayed, and is meaningless otherwise.





Span 2.3 MHz
Ref -13.0 dBm
dB/ 5.0
RBW 110 kHz
Peak Deviation 0.8926 Mhz
Mrkr1 -31.7 dBm
Mrkr1 70.21 Mhz
Mrkr2 -28.5 dBm
Mrkr2 70.00 Mhz
Mrkr Delta -0.2148 Mhz

Span 2.3 MHz
Ref -13.0 dBm
dB/ 5.0
RBW 110 kHz
Mrkr 1 -27.3 dB
Mrkr 2 -45.0 dB
Mrkr Delta 17.63 dB

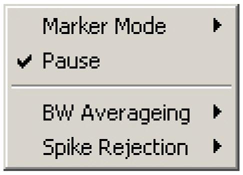
Delta Frequency Markers ***Delta Amplitude Markers***

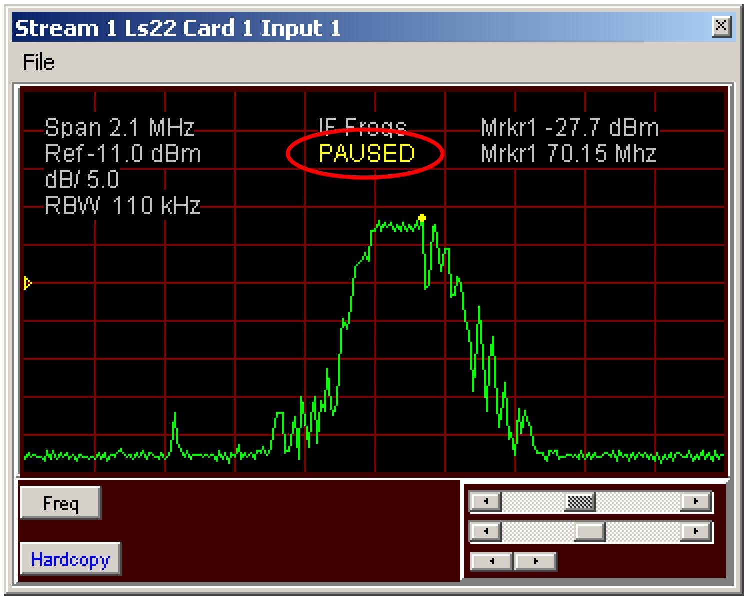
Figure 3-47 Marker Mode → Marker Delta

The Amplitude Marker Delta mode is shown right in Figure 3-47 above and is represented by two horizontal dotted lines. Each marker line increases or decreases in amplitude via the control shown on the display as indicated in the red square in the figure above. Also note that the markers amplitude, and amplitude delta are shown on the display as indicated in the yellow square in the figure above.

3.6.4.2 Pause Mode

Select the Pause Mode by right clicking in the upper portion of the spectrum display. This will freeze the updating of the spectrum display and is indicated as shown in the red oval in Figure 3-36 below. To resume the dynamic updating of the spectrum display, again select the Pause Mode by right clicking in the upper portion of the spectrum display. The activation of this mode is indicated by a check mark symbol (√) next to the “PAUSE” item in the menu.

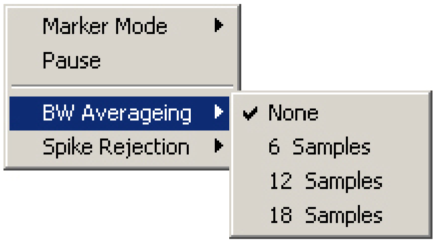




Span 2.1 MHz
Ref -11.0 dBm
dB/ 5.0
RBW 110 kHz
IF Freqs
PAUSED
Mkr1 -27.7 dBm
Mkr1 70.15 Mhz

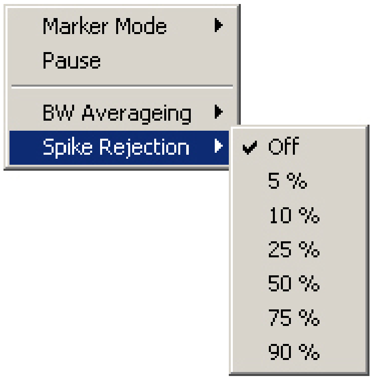
Figure 3-48 Spectrum Display in Pause Mode

3.6.4.3 Bandwidth (BW) Averaging



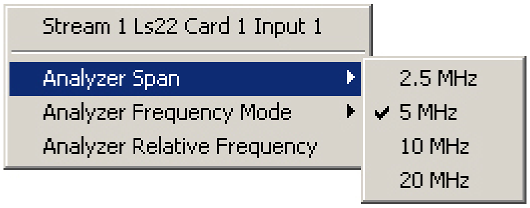
Select the Bandwidth Averaging Mode by right clicking in the upper portion of the spectrum display. The default mode is set to None, but may be set to 6, 12, and 18 samples. When the Bandwidth Averaging mode is selected, the specified number of samples are accumulated and averaged for each point on the display. Thus, each point on the display is updated every 6, 12, or 18 samples, and the value for each point is the average over the 6, 12, or 18 samples. Use the BW Averaging mode to eliminate any random “spikes” that may be seen in the display.

3.6.4.4 Spike Rejection

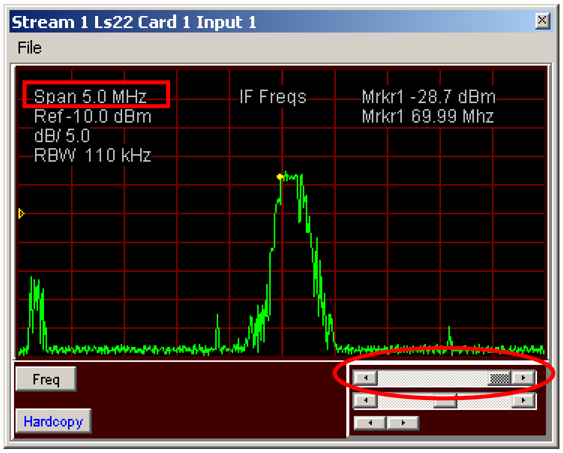


Select the Spike Rejection Mode by right clicking in the upper portion of the spectrum display. The default mode is set to None, but may be set to 5, 10, 25, 50, 75, or 90 percent. The Spike Rejection mode runs each point in the display through a filter that continuously calculates a running standard deviation for each point. For each new point, if the value for the point is outside one standard deviation by the selected value (5, 10, 25, 50, 75, or 90 percent), then that point is thrown away and not displayed. The Spike Rejection mode is more sophisticated than BW averaging, in that it gives the user more fidelity in selecting the elimination of any random “spikes” that may be seen in the display.

3.6.4.5 Analyzer Span



Select the Analyzer Span by right clicking in the lower portion of the spectrum display. The user may select four different frequency display spans that include: 2.5 MHz, 5 MHz, 10 MHz, and 20 MHz. The frequency span value selected is displayed in the upper left portion of the spectrum display as indicated by the red rectangle in the figure below. Frequency spans other than the standard values mentioned in this paragraph may be selected by the user via the frequency span slider shown in the red oval in the figure below.



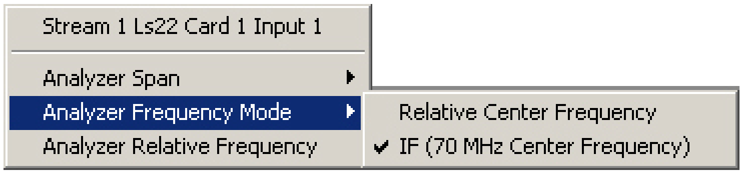
Span 5.0 MHz
Ref -10.0 dBm
dB/ 5.0
RBW 110 kHz

IF Freqs

Mrkr1 -28.7 dBm
Mrkr1 69.99 Mhz

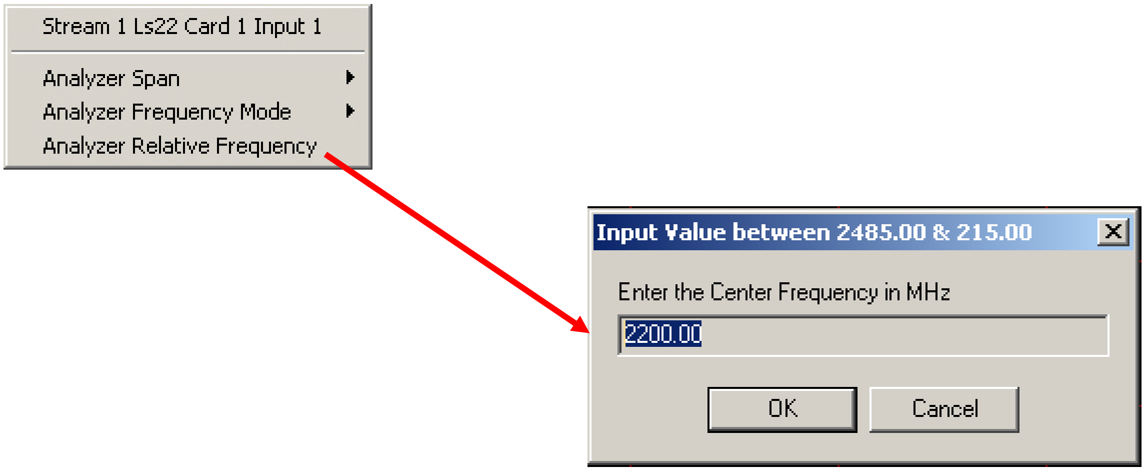
3.6.4.6 Analyzer Frequency Mode

Select the Analyzer Frequency Mode by right clicking in the lower portion of the spectrum display. The user may select Relative Center Frequency, or IF (70 MHz Center Frequency). The default mode is IF (70 MHz Center Frequency). In the IF mode, the center frequency of the display is set to 70 MHz and all measurements are made relative to that. In the Relative Center Frequency mode, the center frequency of the display is set to a value established by the Analyzer Relative Frequency mode.


Stream 1 Ls22 Card 1 Input 1
Analyzer Span
Analyzer Frequency Mode
Analyzer Relative Frequency
Relative Center Frequency
✔ IF (70 MHz Center Frequency)

3.6.4.7 Analyzer Relative Frequency

Select the Analyzer Relative Frequency by right clicking in the lower portion of the spectrum display. The value entered in the dialog box may be any frequency between 200 and 2399.5 MHz. In the Relative Frequency mode, the center frequency of the display is set to the entered value and all measurements are made relative to that.



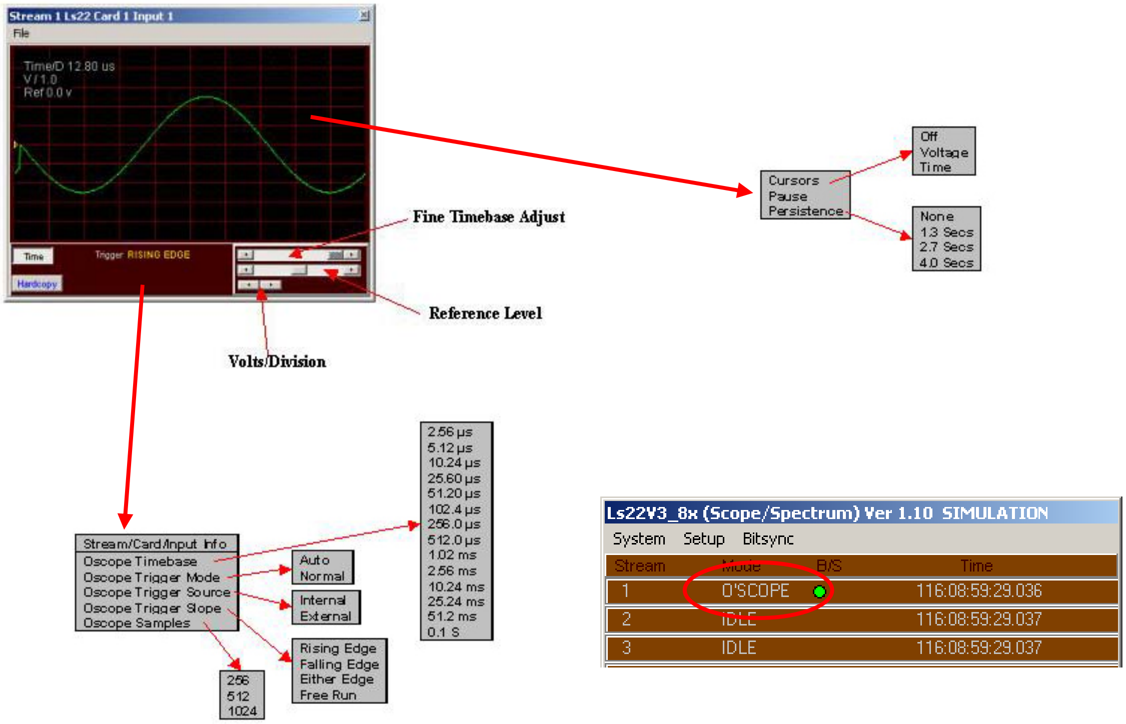
Input Value between 2485.00 & 215.00

Enter the Center Frequency in MHz

2200.00

3.6.5 The Time-Domain Display

From the “Ls22Vx_8x (Scope/Spectrum)” display shown right in Figure 3-1 on page 77, click “Setup” and then “Card Input 1” *(Setup →Card Input 1).* The LS-22-SE Time Domain window shown in the upper left of Figure 3-49 below will be launched. Also notice that the “Mode” of Card Input 1 has changed from “IDLE” to “O’SCOPE” as shown in Figure 3-49 below lower right. The other inputs of the LS-22-SE card are configured in a similar manner *(Setup → Card Input 2, 3, etc).*

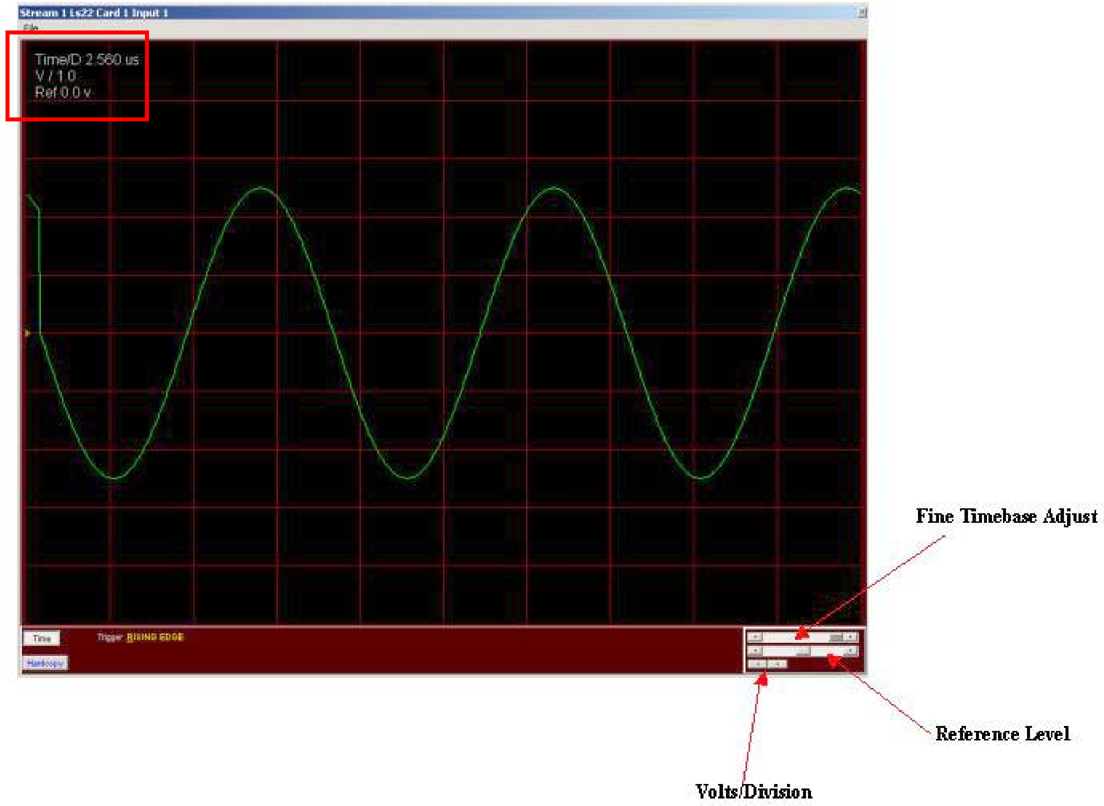


Stream	Mode	B/S	Time
1	O'SCOPE	●	116:08:59:29.036
2	IDLE		116:08:59:29.037
3	IDLE		116:08:59:29.037

Figure 3-49 Time Domain Display Menus/Controls

The “Ls22Vx_8x (Scope/Spectrum)” display shown in Figure 3-50 below is divided into several regions. In the upper portion of the window is the graphical data display area where the time domain graphs are generated. Also in the upper left portion of the time-domain display as shown in the red rectangle are the time base setting, the volts/division, and the reference voltage level. These parameters are controlled in the lower right region below the graphical display has slider or knob controls as shown in the figure.

To invoke the controls for either the upper or lower portions of the display, simply place the mouse curser in the region and right click. The resulting menus are shown in Figure 3-49 above and are discussed in detail in the following paragraphs.



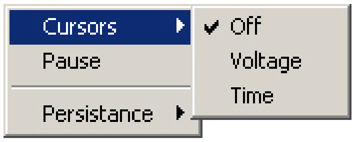
Time/D 2.560 us
V / 1.0
Ref 0.0 v

Trigger: RISING EDGE

Figure 3-50 The Time-Domain Display (Oscilloscope)

3.6.5.1 Cursor Controls

Select the Cursor Control Mode by right clicking in the upper portion of the time domain display. The Cursor Control Mode has three sub-modes: Off, Voltage, and Time. Each in turn is discussed in the following paragraphs.



3.6.5.1.1 Voltage Cursors

The Voltage Cursor mode is shown in Figure 3-51 below and is represented by two horizontal dotted lines. Each marker line increases or decreases in voltage via the control shown on the display as indicated in the red square in the figure below. Also note that the markers voltage, and voltage delta are shown on the display as indicated in the yellow square in the figure below.

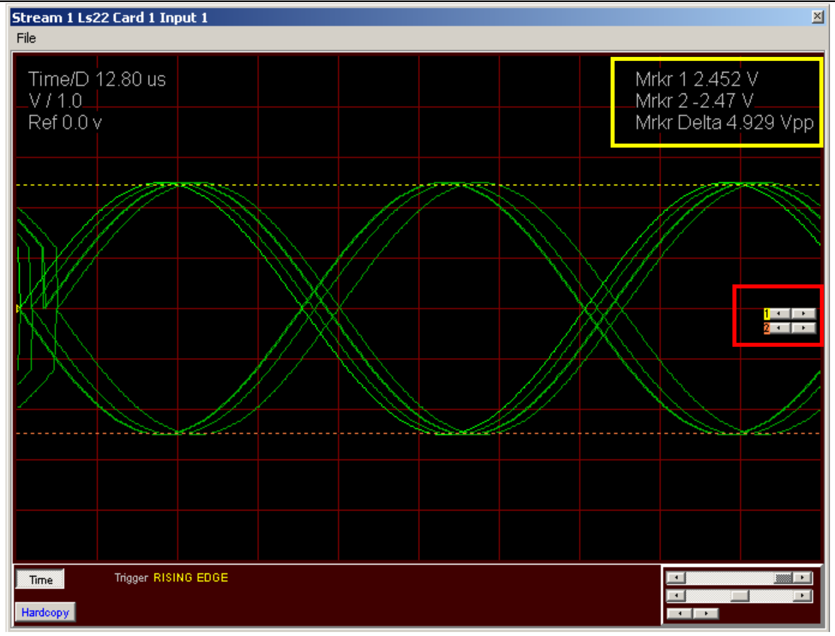
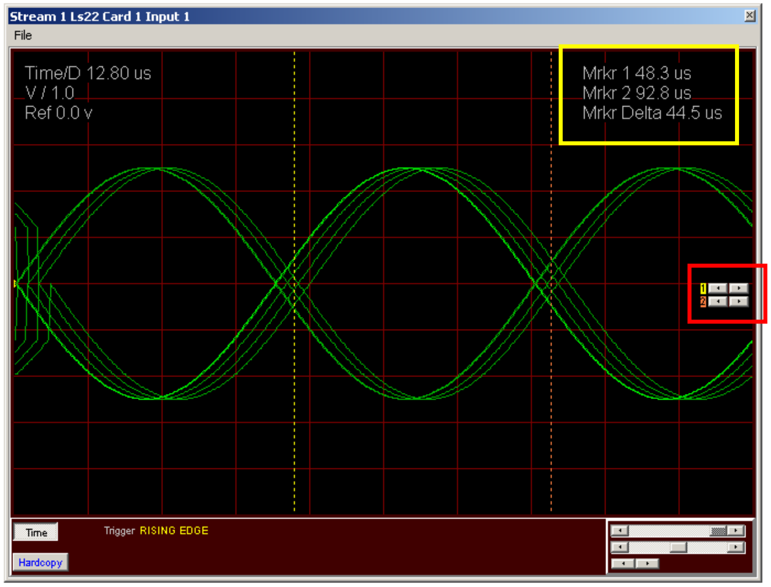


Figure 3-51 Voltage Cursor Mode

3.6.5.1.2 Time Cursors

The Time Cursor mode is shown in Figure 3-52 below and is represented by two vertical dotted lines. Each marker line advances or recedes in time via the control shown on the display as indicated in the red square in the figure below. Also note that the markers time, and time delta are shown on the display as indicated in the yellow square in the figure below.



Time/D 12.80 us
V / 1.0
Ref 0.0 v

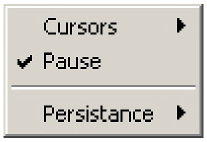
Mrkr 1 48.3 us
Mrkr 2 92.8 us
Mrkr Delta 44.5 us

Trigger RISING EDGE

Figure 3-52 Time Cursor Mode

3.6.5.2 Pause Mode

Select the Pause Mode by right clicking in the upper portion of the time domain display. This will freeze the updating of the time domain display and is indicated as shown in the red oval in Figure 3-53 below. To resume the dynamic updating of the time domain display, again select the Pause Mode by right clicking in the upper portion of the time domain display. The activation of this mode is indicated by a check mark symbol (√) next to the “PAUSE” item in the menu.



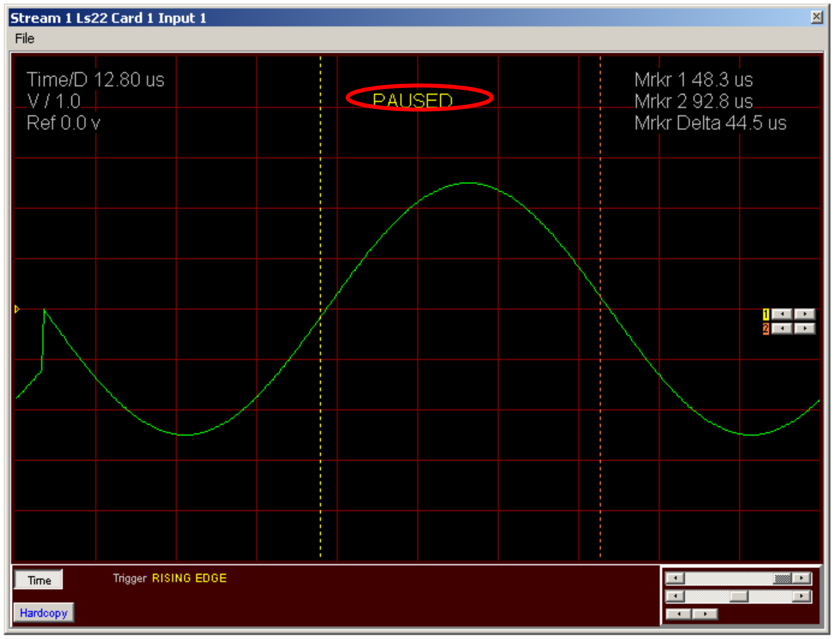
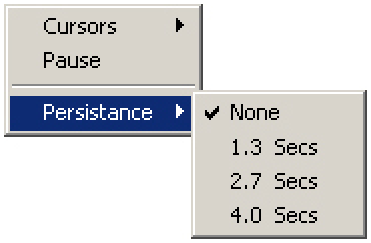
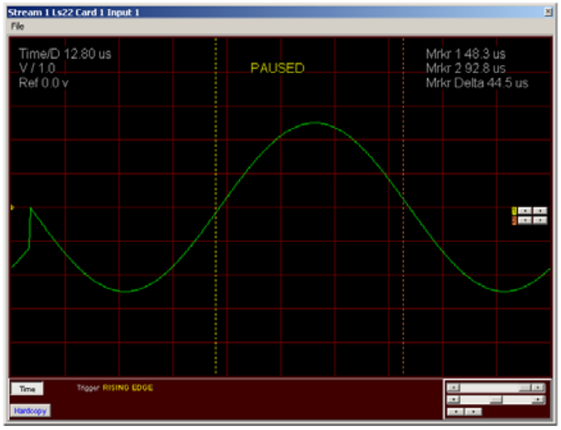


Figure 3-53 Time Domain Display in Pause Mode

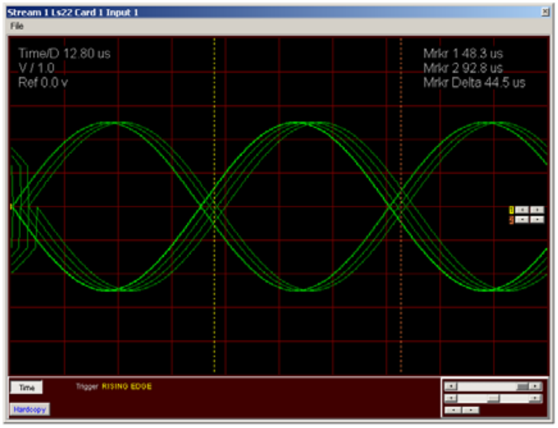
3.6.5.3 Persistence Control



Select the Persistence Control Mode by right clicking in the upper portion of the time domain display. The Persistence Control has four sub-modes: None, 1.3, 2.7, and 4.0 Seconds. Use the Persistence Control to examine for example the "eye-pattern" of a demodulated datastream. With persistence of the display, successive cycles of a time-varying signal are in essence, displayed one on top of the other, thus creating a temporal layering of successive and often transitory events. An example of this effect is shown right in Figure 3-54 below.



Persistence → None

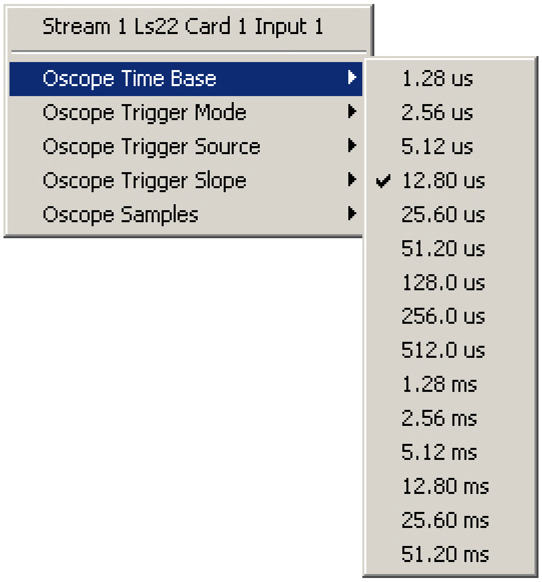


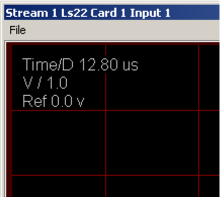
Persistence → 4.0 Seconds

Figure 3-54 Example of Display Persistence

3.6.5.4 Oscilloscope Timebase Control

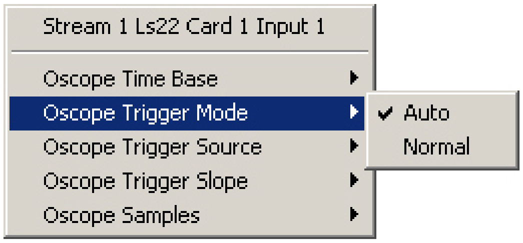
Select the Oscilloscope Timebase Control Mode by right clicking in the lower portion of the time domain display. With the time base control, the user may select fourteen (14) different display time base settings. The range of time base settings depends on the sample size selected from the Oscope Samples Control Mode (see paragraph 3.6.5.8 for more details), and range from 640 ns to 0.10 s. The time base setting (time/division) is display as shown right in the upper left corner of the display area. The range of time base settings shown below corresponds to a sample size of 512.



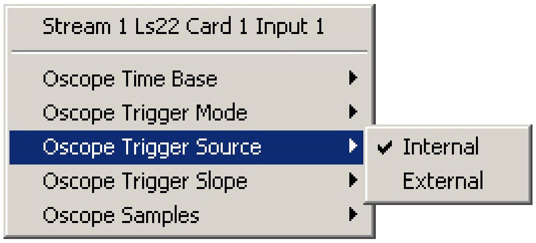


3.6.5.5 Oscilloscope Trigger Mode

Select the Oscilloscope Trigger Mode by right clicking in the lower portion of the time domain display. The Trigger Mode has two sub-modes: Auto and Normal. In normal mode the oscilloscope only sweeps if the input signal reaches the set trigger condition. Auto mode causes the oscilloscope to sweep, even without a trigger.

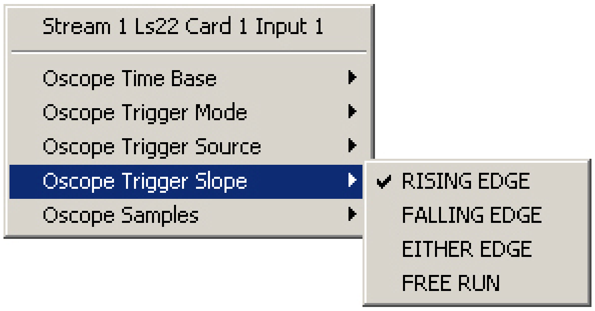


3.6.5.6 Oscilloscope Trigger Source



Select the Oscilloscope Trigger Source by right clicking in the lower portion of the time domain display. The Trigger Source Mode has two sub-modes: Internal and external. In the Internal mode, the display will be triggered from the signal being displayed. In the External mode, the external trigger signals for each of the three possible channels of the LS-22-SE will be used to trigger the display.

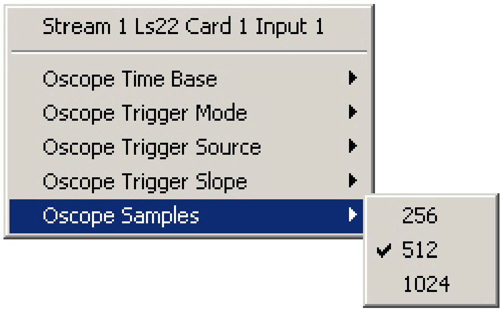
3.6.5.7 Oscilloscope Trigger Slope



Select the Oscilloscope Trigger Slope by right clicking in the lower portion of the time domain display. The Trigger Slope Mode has four sub-modes: Rising Edge, Falling Edge, Either Edge, or Free Run. In the Rising Edge mode, a positive transition (rising edge) of either the Internal or External trigger will trigger the display. In the Falling Edge mode, a negative transition (falling edge) of either the Internal or External trigger will trigger the display. For the Either Edge mode, both a rising or falling edge of either the Internal or External trigger will trigger the display. In the Fee Run mode, the trigger conditions (rising/falling edges) are ignored and the display continuously updates.

3.6.5.8 Oscilloscope Samples

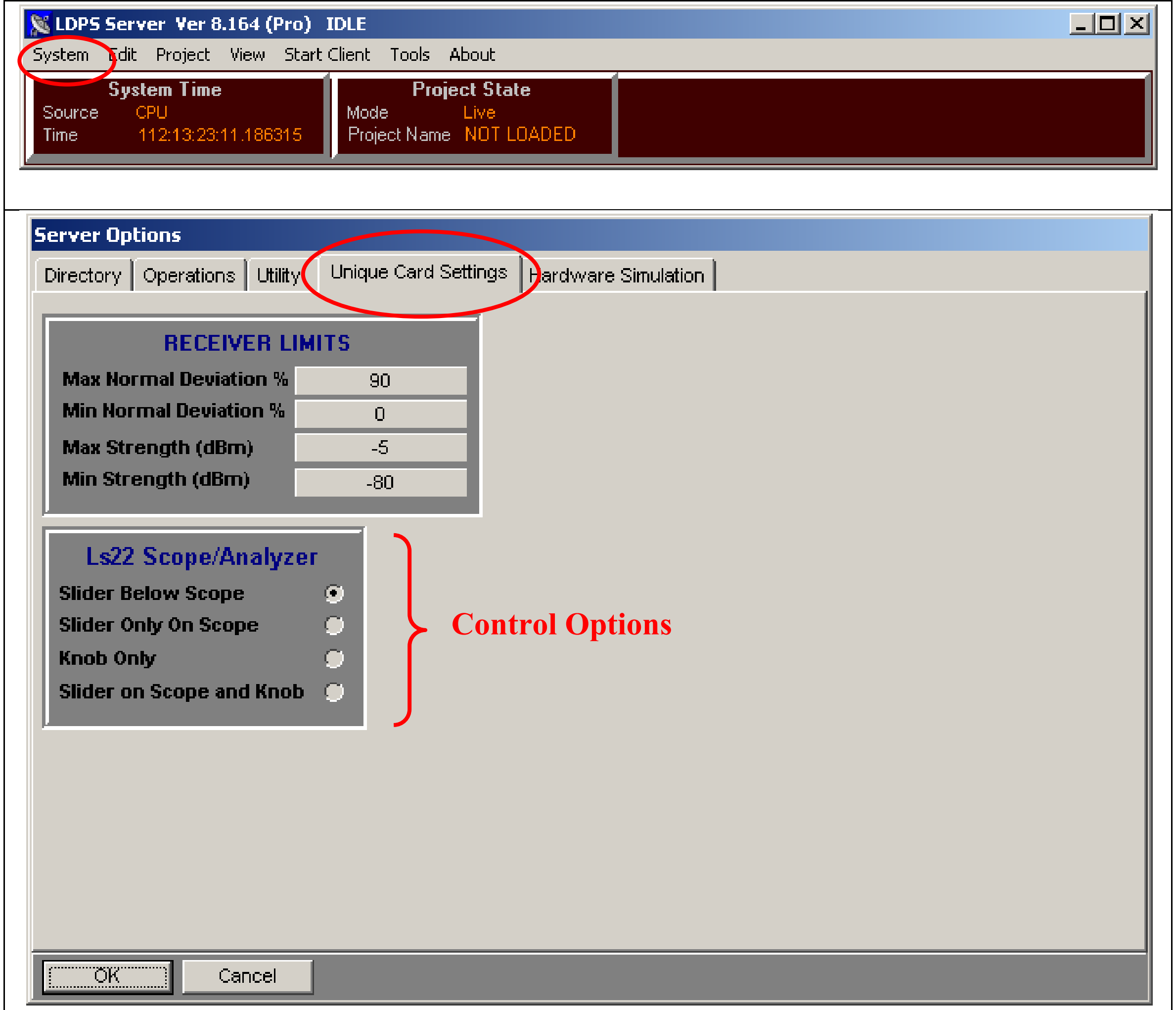
Select the Oscilloscope Sample size by right clicking in the lower portion of the time domain display. The Oscilloscope Sample Mode has three sub-modes: 256, 512, and 1024 samples.



3.6.6 Modifying the Spectral and Time-Domain Controls

Both the Spectral and Time-Domain displays offer several different types of graphical controls. In addition to the standard “slider” controls shown in Figure 3-4 on page 81 and Figure 3-50 on page 155, several additional control configurations are available. To change the type of controls found on the displays, select the server options from the

LDPS Server window (System → Options) and click on the "Unique Card Settings" tab. The four different types of controls for the LS-22-SE are shown in Figure 3-55 below. After selecting a particular controls configuration, click the OK button and then from the LDPS Server window restart all devices (System → Devices → Restart). Note: the change in control configuration will only take effect after a reset.



System Time
Source CPU
Time 112:13:23:11.186315

Project State
Mode Live
Project Name NOT LOADED

RECEIVER LIMITS

Max Normal Deviation %	90
Min Normal Deviation %	0
Max Strength (dBm)	-5
Min Strength (dBm)	-80

Ls22 Scope/Analyzer

- Slider Below Scope ●
- Slider Only On Scope ○
- Knob Only ○
- Slider on Scope and Knob ○

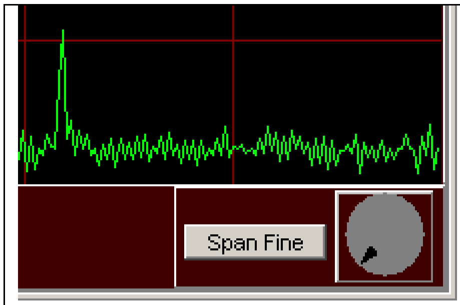
Control Options

Figure 3-55 Server Options - Unique Card Settings

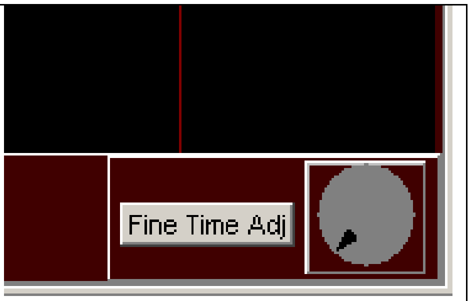
The default control configuration, *"Slider Below Scope"* are shown in the various displays seen in the document thus far (Figure 3-4 and Figure 3-50 for example).

In the *"Slider Only On Scope"* configuration, the three sliders at the lower right of the display are moved to locations in the upper portion of the display. For example, the frequency span slider in the spectral display or the fine time base adjustment slider in the time-domain display appear as a large horizontal scroll bar extending across the entire upper edge of the display. The reference level slider in both the spectral and time-domain displays appear as a large vertical scroll bar extending across the entire left edge of the display. The dB/division for the spectral display and Volts/division for the time-domain sliders appear as single up/down arrows in the upper left portion of the display. To invoke any of these slider controls for this configuration, place the mouse cursor in the upper portion of the display and right click and select the parameter to adjust. Once selected, the slider control will persist until a different slider is selected.

In the *"Knob Only"* configuration, the three horizontal sliders at the lower right of the display are replaced by a single multi-function virtual knob and a parameter selection button. Clicking on the button will cycle through the available adjustment parameters. To make an adjustment, select a parameter and place the mouse cursor on the knob and click and hold the mouse button while rotating the knob with the mouse. For the spectral display, the available adjustment parameters include: Fine Span, Course Span, dB/Division, and Reference Level. For the time-domain display, the available adjustment parameters include: Fine Timebase Adjust, Volts/Division, Vertical Position, and Course Timebase Adjust.



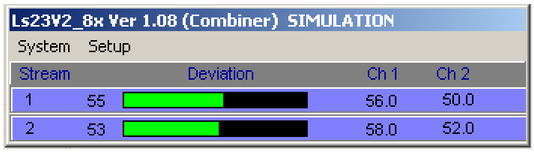
Spectral Display



Time-Domain Display

The *"Slider On Scope and Knob"* configuration is a simple union of the "Slider Only On Scope" and "Knob Only" configurations described previously.

3.7 LS-23 Pre-Detect and Post-Detect Combiner



Stream	Deviation	Ch 1	Ch 2
1	55	56.0	50.0
2	53	58.0	52.0

The Combiner setup GUI shown right is a small window launched from the main application window on the Server. It is launched by the user by selecting *Setup → Card Input 1* on the main application window, or when a project is loaded. This GUI controls all LS-23 Pre-Detect & Post-Detect Combiner cards installed in the system. The Pre-Detect & Post-Detect Combiner setup GUI has three major parts: The caption area, the menu area, and the display/status area.

The caption area (top of the window) displays the name of the window, and if a simulation is running, the word “SIMULATION” will be flashing. Note there is no normal window closing mechanism in the caption area (the X normally found in the upper right corner of windows). This is because the window must be active in order to gather data from the multifunction card, and to control it.

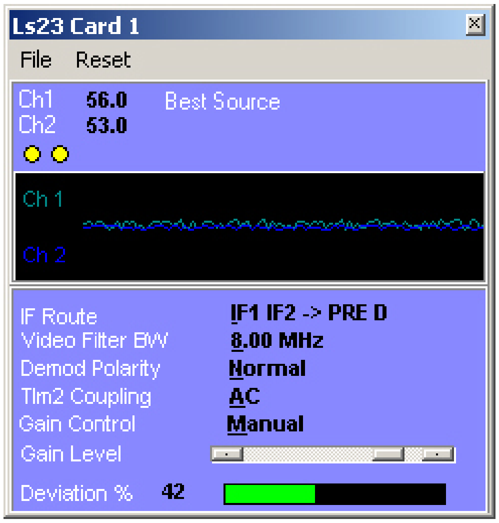
Below the caption area and setup menu is the status display for all LS-23V2 cards installed in the system. For each stream, the status display shows the optimal signal deviation numerical value along with a horizontal bar graph of same. Also displayed are the individual numerical values for channel 1 and channel 2 signal deviations.

3.7.1 System

The system menu has a single command: Flash Board ID Leds. To identify a specific LS-23 card(s) in a system with multiple cards, the user may invoke the **Flash Board ID Leds** command, and select the desired stream number from the list of available streams. Remember that each LS-23 card in the system has a unique and specific stream number assigned to it.

3.7.2 Setup

The setup menu allows the user to configure each stream in the system and the LS-23 hardware associated with each stream. Selecting a specific stream from the list of possible streams will invoke the configuration setup screen shown right. The setup and configuration of the LS-23 is described in more detail in paragraphs 3.7.3 on page 165.



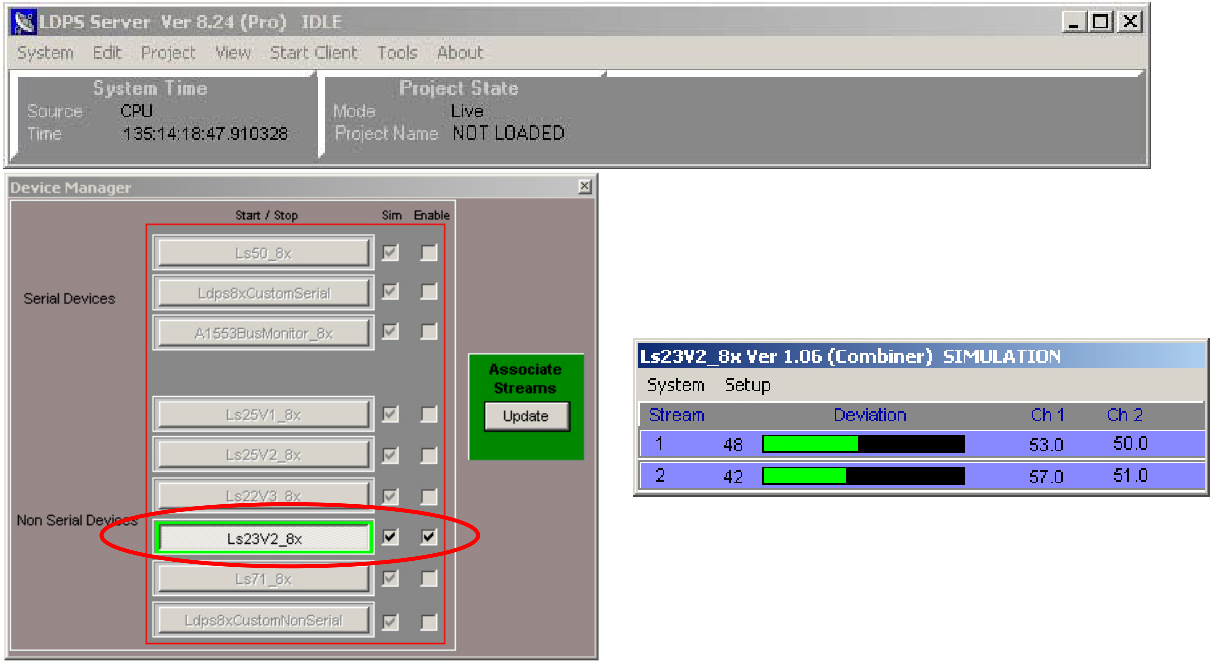
Ch1 56.0 Best Source
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Normal
Tlm2 Coupling AC
Gain Control Manual
Gain Level
Deviation % 42

Figure 3-56 Setup Display for the LS-23

To initially configure the LS-23, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select "Devices" and then "Manage" *(System→ Devices→ Manage)*
2. From the System Manager shown below left, select the "Enable" check box next to the LS23V2 button. The "Ls23V2_8x" button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the "Sim" check box next to the "Enable" check box. Checking this box allows the LDPS application to operate when a LS-23 board is not installed in the system.
3. From the System Manager, click the "Ls23V2_8x" button. This will launch the "Ls23V2_8x (Combiner)" display shown below right. Note that the red rectangle around the button has changed to green indicating that the application is now running.
4. To setup and configure the LS-23 card(s), follow the procedures outlined in paragraphs 3.7.3.



System Time
Source CPU
Time 135:14:18:47.910328

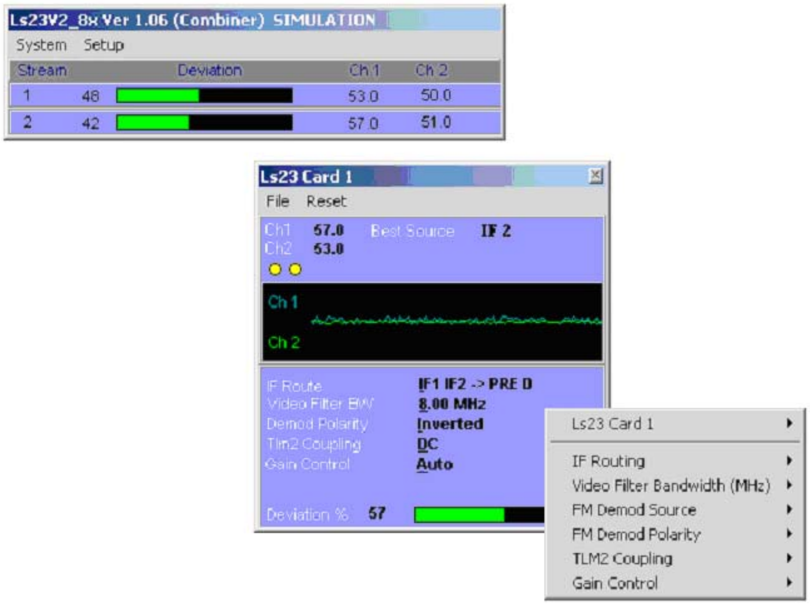
Project State
Mode Live
Project Name NOT LOADED

Stream	Deviation	Ch 1	Ch 2
1	48	53.0	50.0
2	42	57.0	51.0

Figure 3-57 Server Application Windows (LS-23 V2 Enabled)

3.7.3 Configuring The LS-23 Hardware

From the "Ls23V2_8x (Combiner)" display shown below left in Figure 3-58, click "Setup" and then "Stream 1" *(Setup →Stream 1).*



Stream		Deviation	Ch 1	Ch 2
1	48		53.0	50.0
2	42		57.0	51.0

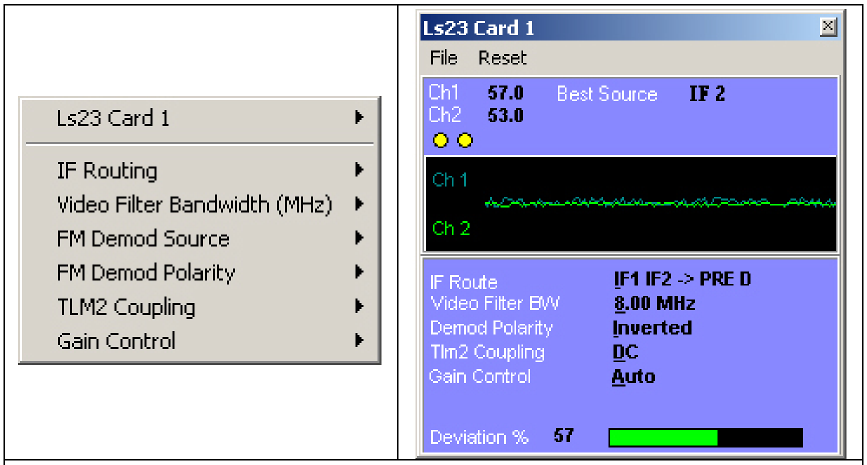
Ch1 57.0 Best Source IF 2
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto

Deviation % 57

Figure 3-58 Configuration Menus/Controls for the LS-23

The "Ls23 Card 1" display shown in Figure 3-59 below right is divided into several regions. The number of, and function of each region is dependent on the various modes the card can be configured for. To invoke the controls for the display, simply place the mouse curser in the region and right click. The resulting menus shown in Figure 3-59 below left are discussed in detail in the following paragraphs.



Ch1 57.0 Best Source IF 2
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto

Deviation % 57

Figure 3-59 The LS-23 Configuration Menu

Once the LS-23 has been completely configured, save the configuration state by invoking the “File→SaveAs” command. Note the “*.LS23V2” file extension for the configuration file and the “Dbase” directory where it is stored. Previously defined LS-23 configurations may be recalled and applied by invoking the “File→Recall” command. Changes made to an existing LS-23 configuration can be applied to the hardware by invoking the “Reset” command.

3.7.3.1 Status and Configuration Elements of the LS-23 Display

Depending on the IF Routing mode, the upper portion of the LS-23 configuration display shown above in Figure 3-59 will present the instantaneous numerical values of the relative signal strength percentage for Ch1, Ch2, or both. It will also display the “Best Source” currently being used by the LS-23 to combine the two inputs when in the “IF1&IF2 → PRED” IF routing mode. The best sources can be “IF1,” “IF2,” or “OPTIMAL.” (OPTIMAL indicates that the IF 1 and IF 2 inputs are both equal in magnitude and are in phase).

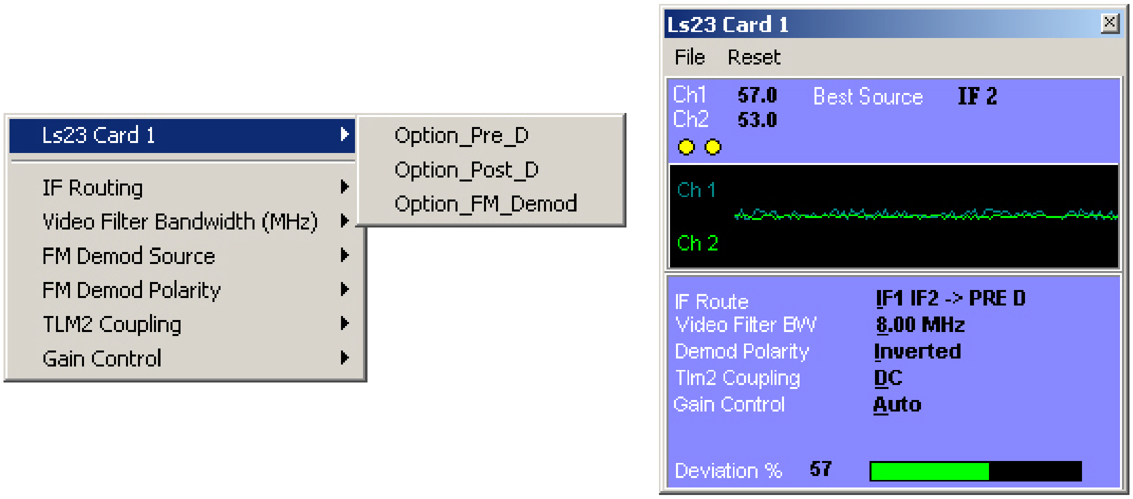
Below the CH1/CH2 numerical displays for relative signal strength are two status “LEDs.” The LEDs indicate yellow, green or gray. Green indicates an in-sync condition, yellow, out-of-sync. Gray indicates that the option is not installed. The left-hand LED represents the lock status of the phase-lock-loops in the Pre-Detection combiner section of the LS-23. The right-hand LED represents the lock status of the Post-Detection combiner section of the LS-23.

The rolling strip chart in the center of the configuration display is used to graphically represent the time-varying waveforms of the received signal strengths for CH1, CH2, or both, depending on the IF Routing mode.

The numerical displays of the lower portion of the LS-23 configuration display show the configuration status of the combiner and include such parameters as: IF Routing mode, Post Detection Video Filter bandwidth, Demodulator Polarity, TLM2 Output Coupling Mode, Gain Control Mode, and FM Deviation Percentage.

3.7.3.2 LS-23 Configuration Options

To determine the configuration options installed on the selected LS-23 card, right click in the “Ls23 Card 1” display shown below right. The LS-23 can have up to three factory-installed configurations that include: Pre-Detection, Post-Detection, and FM Demodulation. In the example shown in the figure below left, all three options are installed. During the initialization of the LS-23, the LDPS application reads the configuration PROM and determines what configurations options have been installed on the card. The selected LS-23 card can have all, or any combination of the three configuration options.



Ch1 57.0 Best Source IF 2
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto

Deviation % 57

Figure 3-60 LS-23 Mode Controls

Each of the operating configurations is discussed in the following paragraphs.

3.7.3.2.1 Pre-Detection Option

The LS-23 design contains a 70MHz IF Pre-Detection combiner option designed to blend two 70MHz sources in an optimally weighted fashion and provide the best possible IF output to the demodulation stages that follow.

3.7.3.2.2 Post-Detection Option

The LS-23 design contains a baseband Post-Detect combiner option designed to blend two baseband sources in an optimally weighted fashion that provides the best possible baseband output.

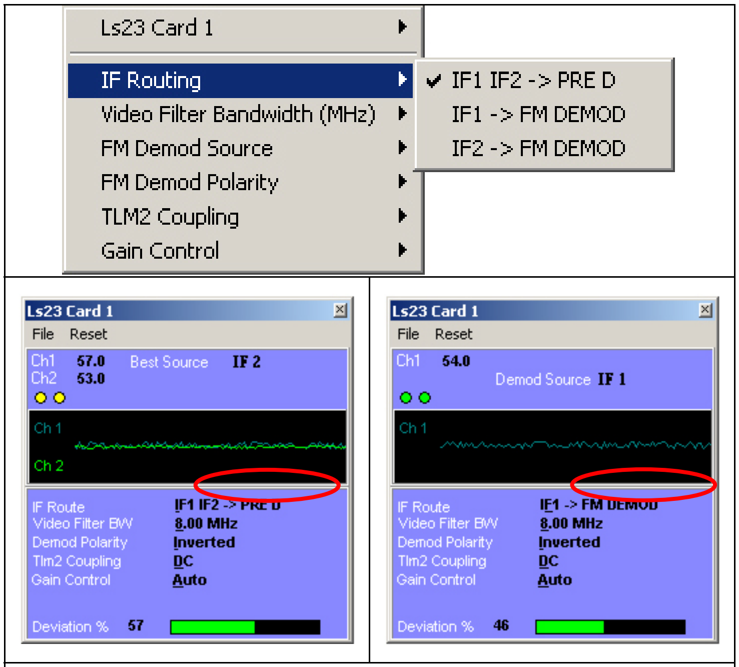
3.7.3.2.3 FM Demodulator Option

The LS-23 design contains a dual FM discriminator option that may be applied to the output of the Pre-Detect Combining stage or to one of two external inputs.

3.7.3.3 IF Routing

The LS-23 provides the user with the ability to control the IF Input channels. The IF input channels may be routed to the Pre-Detection Combiner Stage, or they may be routed, one at a time directly to the FM demodulation stage. Once selected, the IF Routing mode is indicated in Figure 3-61 below as shown in the red ovals. Select the LS-23 IF Routing mode by right clicking in the “Ls23 Card 1” display shown below. The LS-23 has three IF Routing modes:

IF1&IF2 → PRED, IF1 → FM DEMOD, and IF2 → FM DEMOD.



Ch1 57.0 Best Source IF 2
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto
Deviation % 57

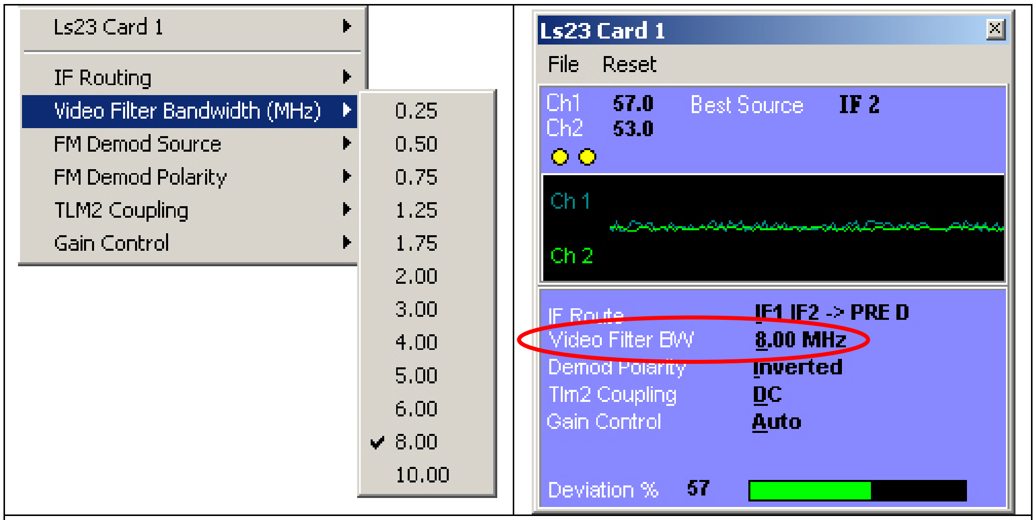
Ch1 54.0 Demod Source IF 1

IF Route IF1 -> FM DEMOD
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto
Deviation % 46

Figure 3-61 IF Routing Controls & Resulting Displays

3.7.3.4 Video Filter Bandwidth

To further reduce the noise bandwidth of the FM demodulated baseband output signal, the LS-23 combiner contains twelve video filters. The user can select one of the twelve filters (0.25 to 10.0 MHz) from the menu shown below left. Once selected, the Video Filter Bandwidth is indicated in the display as shown below right in the red oval. Select the LS-23 Video Filter Bandwidth by right clicking in the "Ls23 Card 1" display shown below right.



Ch1 57.0 Best Source IF 2
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto
Deviation % 57

Figure 3-62 Video Filter Bandwidth (MHz)

3.7.3.5 FM Demodulation Source

The LS-23 combiner design allows the user to select the input source of the dual FM discriminator stage. Either the internal 70MHz source may be used as the discriminator input, or an external input via the J5 or J6 connector may be selected. Select the LS-23 FM Demodulation source by right clicking in the "Ls23 Card 1" display shown in Figure 3-62 above.

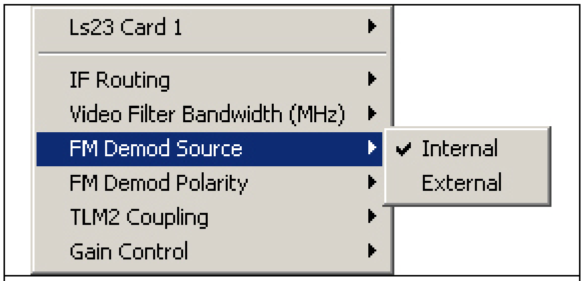
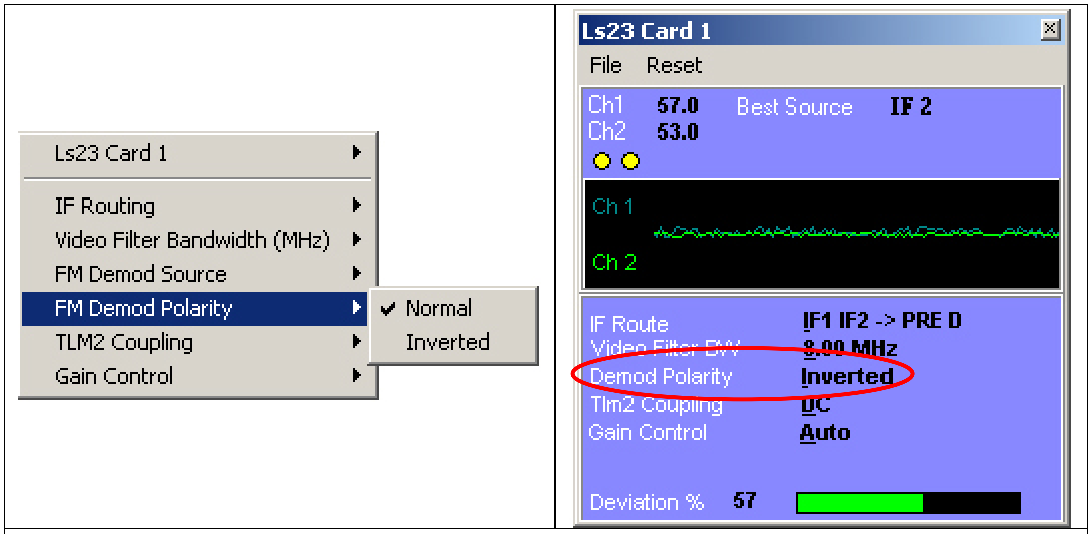


Figure 3-63 FM Demodulation Source

3.7.3.6 FM Demodulation Polarity

The baseband processing section of the LS-23 combiner allows the user to select the polarity of the PCM output data. This function may be necessary for spectrally inverted RF input sources. Two polarities are available: Normal, and Inverted. Select the LS-23 FM Demodulation Polarity by right clicking in the "Ls23 Card 1" display shown below right.



Ch1 57.0 Best Source IF 2
Ch2 53.0

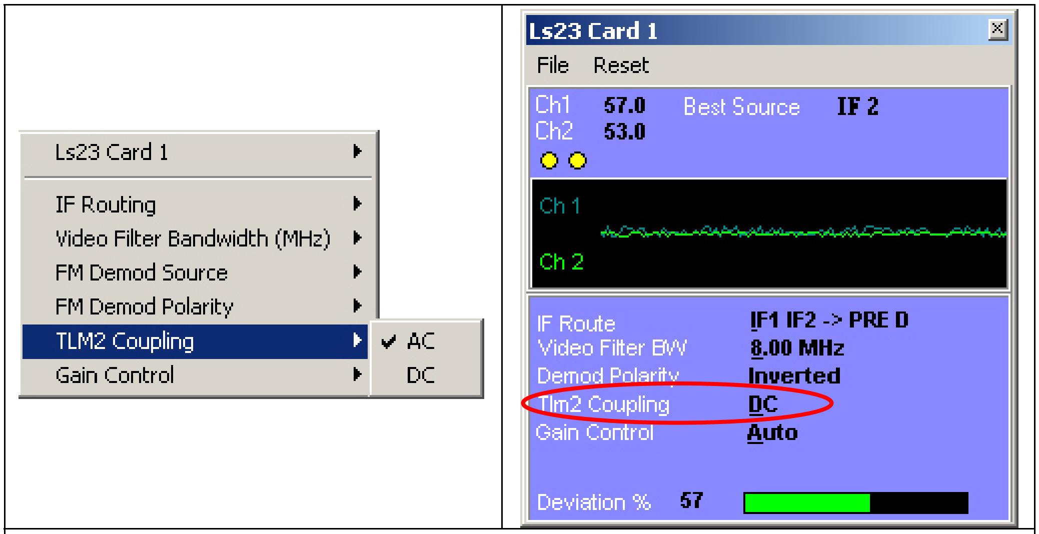
IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto

Deviation % 57

Figure 3-64 FM Demodulation Polarity

3.7.3.7 TLM2 Coupling

The LS-23 combiner allows the user to select the coupling mode for the TLM2 PCM output port. Two coupling modes are available: AC coupled, and DC coupled. The TLM2 output coupling command only affects the TLM2 output. The TLM1 output-coupling mode (AC coupled) is not affected. Select the LS-23 TLM2 output-coupling mode by right clicking in the "Ls23 Card 1" display shown below right.



Ch1 57.0 Best Source IF 2
Ch2 53.0

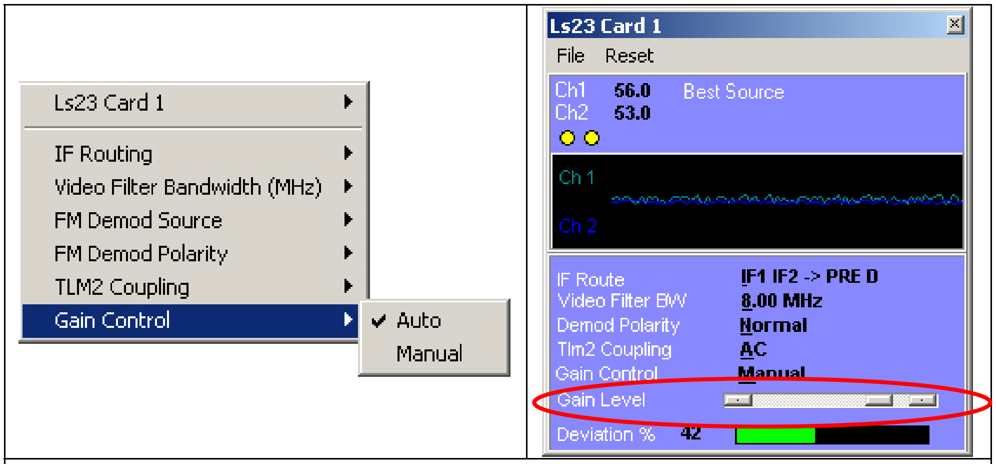
IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Inverted
Tlm2 Coupling DC
Gain Control Auto

Deviation % 57

Figure 3-65 TLM2 Coupling Controls

3.7.3.8 Gain Controls

The LS-23 allows the user to automatically or manually control the output voltage of the telemetry (TLM1 and TLM2) signals produced in the FM Demodulation stage. Two gain control modes are available: Auto, and Manual. In Auto mode, the LS-23's TLM outputs are factory calibrated to produce a 4Vp-p video filter output based on 100% FM deviation. In manual mode, the user is provided a control to adjust the TLM output level. When this control is set to it's minimum, the output level is set to approximately 500mVp-p. When set to maximum, the output level is set to the maximum amount for the given deviation level. Select the LS-23 Gain Control mode by right clicking in the "Ls23 Card 1" display shown below right. Note that when in Manual mode, an additional gain control slider control will appear on the display as shown below right in the red oval.



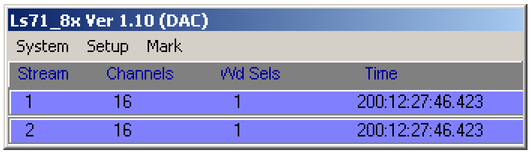
Ch1 56.0 Best Source
Ch2 53.0

IF Route IF1 IF2 -> PRE D
Video Filter BW 8.00 MHz
Demod Polarity Normal
Tlm2 Coupling AC
Gain Control Manual

Deviation % 42

Figure 3-66 Gain Controls & Resulting Display

3.8 LS-71 Multi-channel Analog Output Card



Stream	Channels	Wd Sels	Time
1	16	1	200:12:27:46.423
2	16	1	200:12:27:46.423

The multi-channel analog output card setup GUI shown right is a small window launched from the main application window on the Server. The GUI is launched by the user by selecting *Setup → Stream 1* on the main application window, or when a project is loaded. This GUI controls all LS-71 multi-channel analog output cards installed in the system. The multi-channel analog output card setup GUI has three major parts: The caption area, the menu area, and the display/status area.

The caption area (top of the window) displays the name of the window, and if a simulation is running, the word "SIMULATION" will be flashing. Note there is no normal window closing mechanism in the caption area (the X normally found in the upper right corner of windows). This is because the window must be active in order to gather data from the multifunction card, and to control it.

Below the caption area and setup menu is the status display for all LS-71 cards installed in the system. For each stream, the status display shows the number of channels (16 or 32), the system time, and the number of word selectors configured for each card.

3.8.1 System Menu

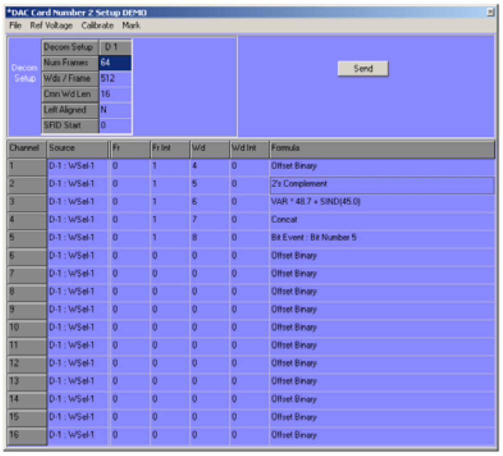
The system menu has a single command: Flash Board ID Leds. To identify a specific LS-71 card(s) in a system with multiple cards, the user may invoke the **Flash Board ID Leds** command, and select the desired stream number from the list of available streams. Remember that each LS-71 card in the system has a unique and specific stream number assigned to it.

3.8.2 Setup Menu

The setup menu allows the user to configure each stream in the system and the LS-71 hardware associated with each stream. Selecting a specific stream from the list of possible streams will invoke the configuration setup screen shown right. The setup and configuration of the LS-71 is described in more detail in paragraphs 3.8.4 on page 173.

3.8.3 Mark Menu

Selecting the "Mark" item will deflect the output to 0 scale, then ¾ scale, then 0 scale for each channel on all LS-71 cards installed.



Decom Setup	D 1
Num Frames	64
Wds / Frame	512
Cmn Wd Len	16
Left Aligned	N
SFID Start	0

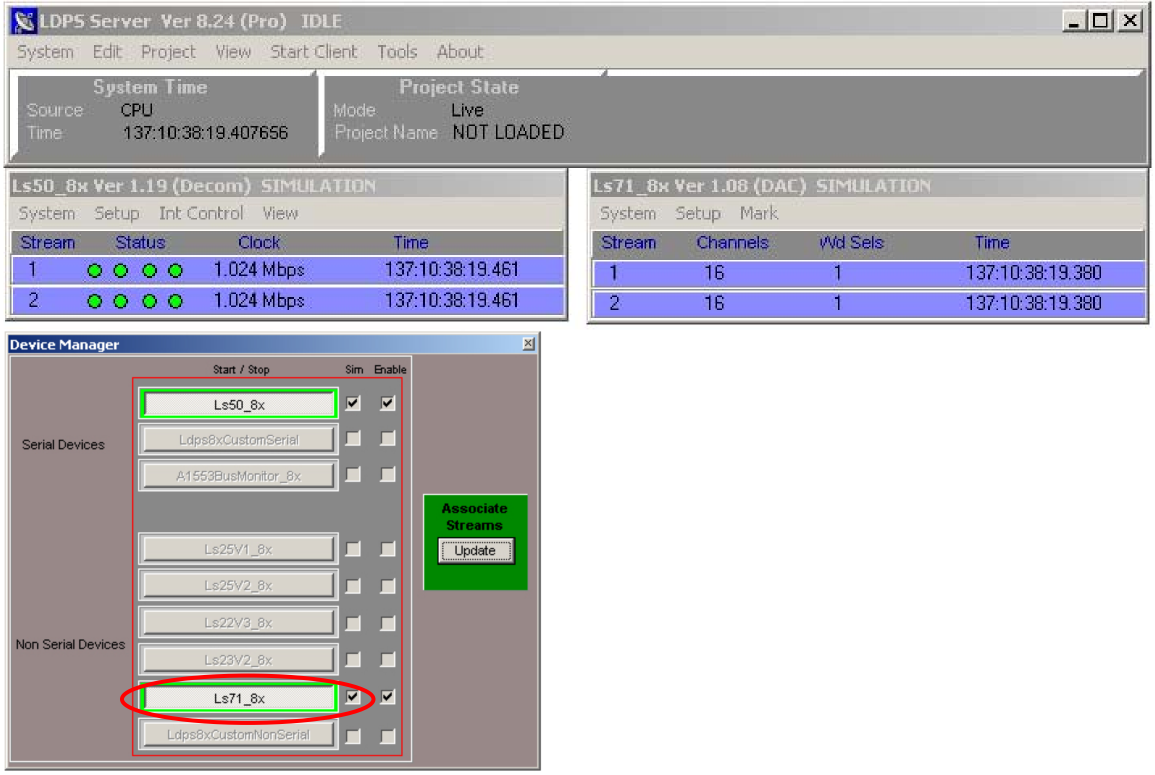
Channel	Source	Fr	Fr Int	Wd	Wd Int	Formula
1	D-1 : WSel-1	0	1	4	0	Offset Binary
2	D-1 : WSel-1	0	1	5	0	2's Complement
3	D-1 : WSel-1	0	1	6	0	VAR * 48.7 + SIND(45.0)
4	D-1 : WSel-1	0	1	7	0	Concat
5	D-1 : WSel-1	0	1	8	0	Bit Event : Bit Number 5
6	D-1 : WSel-1	0	0	0	0	Offset Binary
7	D-1 : WSel-1	0	0	0	0	Offset Binary
8	D-1 : WSel-1	0	0	0	0	Offset Binary
9	D-1 : WSel-1	0	0	0	0	Offset Binary
10	D-1 : WSel-1	0	0	0	0	Offset Binary
11	D-1 : WSel-1	0	0	0	0	Offset Binary
12	D-1 : WSel-1	0	0	0	0	Offset Binary
13	D-1 : WSel-1	0	0	0	0	Offset Binary
14	D-1 : WSel-1	0	0	0	0	Offset Binary
15	D-1 : WSel-1	0	0	0	0	Offset Binary
16	D-1 : WSel-1	0	0	0	0	Offset Binary

Figure 3-67 Setup Display for the LS-71

The LS-71 is designed for use in conjunction with the Lumistar LS-50-P Multifunction PCM Decommutator board and uses an over-the-top communication bus directly from the decommutator to the LS-71 to avoid any glitches in the analog outputs due to PCI bus activity. The LS-71 design allows two LS-50-P Multifunction boards to be used with a single LS-71 board allowing tagged data from either data stream to be directed to any LS-71 channel.

To initially configure the LS-71, perform the following steps:

1. Run the LDPS server program and from the System menu shown below, select "Devices" and then "Manage" *(System→ Devices→ Manage)*
2. From the System Manager shown below left, select the "Enable" check box next to the LS71 button. The "Ls71_8x" button will then become active (not grayed out). Note the red rectangle around the button - this indicates that the application has not yet started. Note also the "Sim" check box next to the "Enable" check box. Checking this box allows the LDPS application to operate when a LS-71 board is not installed in the system.
3. From the System Manager, click the "Ls71_8x" button. This will launch the "Ls71_8x (DAC)" display shown below right. Note that the red rectangle around the button has changed to green indicating that the application is now running.
4. To setup and configure the LS-71 card(s), follow the procedures outlined in paragraphs 3.8.4.



Source CPU
Time 137:10:38:19.407656

Mode Live
Project Name NOT LOADED

Stream	Status	Clock	Time
1	○ ○ ○ ○	1.024 Mbps	137:10:38:19.461
2	○ ○ ○ ○	1.024 Mbps	137:10:38:19.461

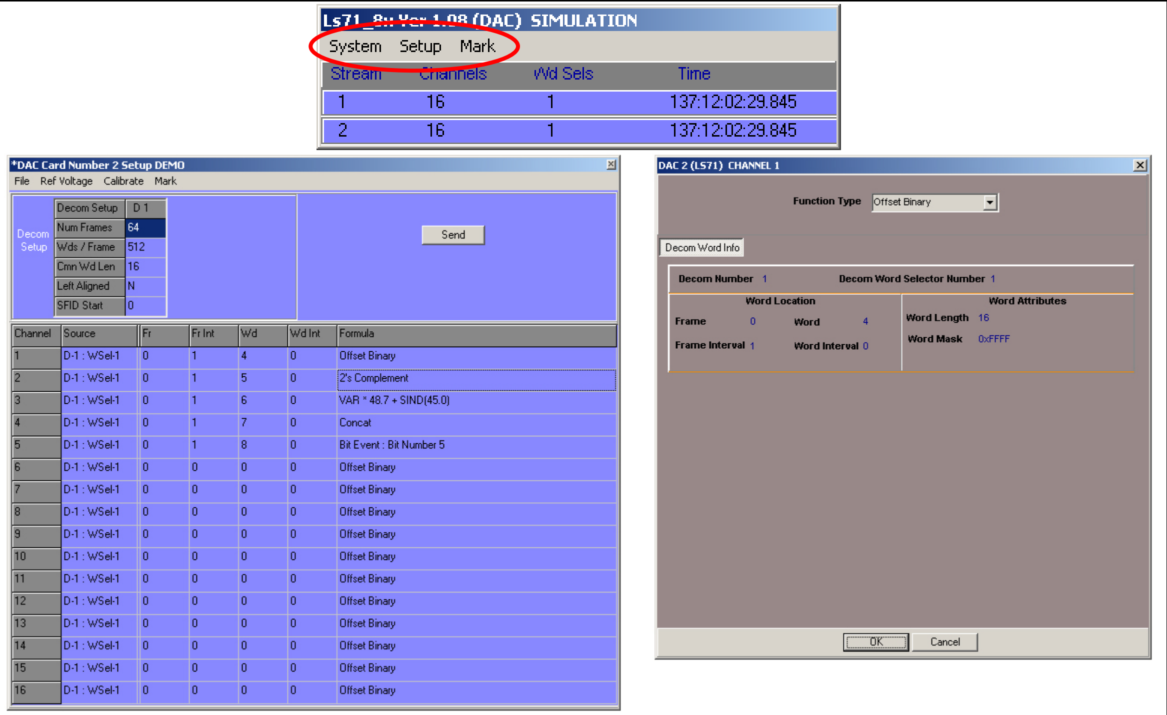
Stream	Channels	Wd Sels	Time
1	16	1	137:10:38:19.380
2	16	1	137:10:38:19.380

Start / Stop	Sim	Enable
Ls50_8x	☑	☑
Ldps8xCustomSerial	□	□
A1553BusMonitor_8x	□	□
Ls25V1_8x	□	□
Ls25V2_8x	□	□
Ls22V3_8x	□	□
Ls23V2_8x	□	□
Ls71_8x	☑	☑
Ldps8xCustomNonSerial	□	□

Figure 3-68 Server Application Windows (LS-71 Enabled)

3.8.4 Configuring The LS-71 Hardware

From the “Ls71_8x (DAC)” display shown below center in Figure 3-69, click “Setup” and then “Stream 1” *(Setup →Stream 1)*. The resulting “DAC Card Number X Setup” display shown below left in Figure 3-69 is divided into several regions. To invoke the controls for this display, simply select a particular menu item, or place the mouse curser in the region and right click. The resulting menus are shown in Figure 3-70 on page 174 are discussed in detail in the following paragraphs.



Stream	Channels	Wd Sels	Time
1	16	1	137:12:02:29.845
2	16	1	137:12:02:29.845

Decom Setup

Decom Setup	D 1
Num Frames	64
Wds / Frame	512
Cmn Wd Len	16
Left Aligned	N
SFID Start	0

Channel	Source	Fr	Fr Int	Wd	Wd Int	Formula
1	D-1 : WSel-1	0	1	4	0	Offset Binary
2	D-1 : WSel-1	0	1	5	0	2's Complement
3	D-1 : WSel-1	0	1	6	0	VAR * 48.7 + SIND(45.0)
4	D-1 : WSel-1	0	1	7	0	Concat
5	D-1 : WSel-1	0	1	8	0	Bit Event : Bit Number 5
6	D-1 : WSel-1	0	0	0	0	Offset Binary
7	D-1 : WSel-1	0	0	0	0	Offset Binary
8	D-1 : WSel-1	0	0	0	0	Offset Binary
9	D-1 : WSel-1	0	0	0	0	Offset Binary
10	D-1 : WSel-1	0	0	0	0	Offset Binary
11	D-1 : WSel-1	0	0	0	0	Offset Binary
12	D-1 : WSel-1	0	0	0	0	Offset Binary
13	D-1 : WSel-1	0	0	0	0	Offset Binary
14	D-1 : WSel-1	0	0	0	0	Offset Binary
15	D-1 : WSel-1	0	0	0	0	Offset Binary
16	D-1 : WSel-1	0	0	0	0	Offset Binary

Function Type: Offset Binary

Decom Number 1   Decom Word Selector Number 1

Word Location
Frame 0   Word 4
Frame Interval 1   Word Interval 0

Word Attributes
Word Length 16
Word Mask 0xFFFF

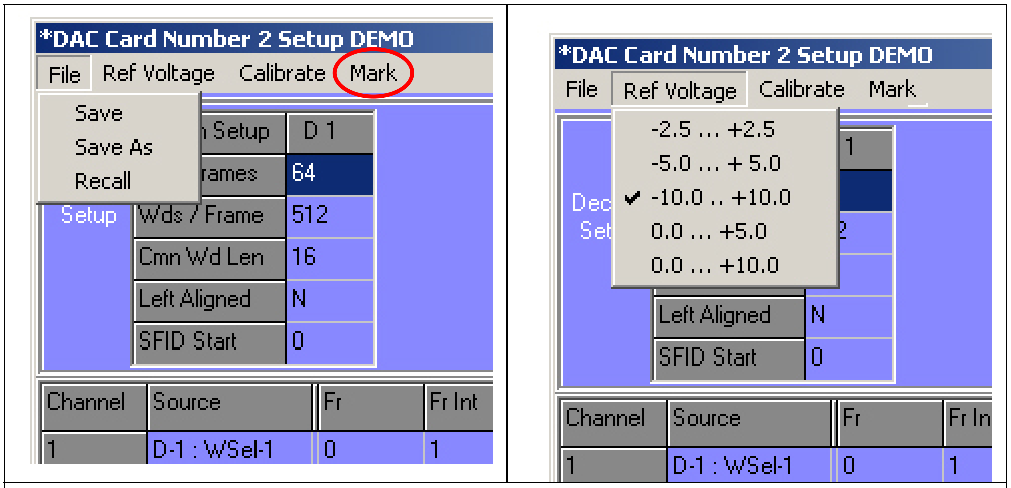
Figure 3-69 Configuration Menus/Controls for the LS-71

The “Ls71_8x (DAC)” display shown above center in Figure 3-69 has three menu items: “System”, “Setup”, and “Mark” (as indicated by red oval). For each LS-71 card installed, the number of channels per/card has will also be displayed. In the example discussed here, two 16-channel LS-71 cards are installed.

Selecting the “Mark” item will deflect the output to 0 scale, then ¾ scale, then 0 scale for each channel on all LS-71 cards installed.

To associate each stream in the system with the corresponding hardware, click “System” and then “Flash Board ID Leds” *(System → Flash Board ID Leds)*. Select the stream number from the drop-down list and the board ID LEDs of the corresponding LS-71 card will begin to flash several times.

To setup the LS-71 hardware for each stream, click “Setup” and then “Stream 1” *(Setup →Stream 1)*. The resulting “DAC Card Number X Setup” display is shown above left in Figure 3-69.



n Setup	D 1
rames	64
Wds / Frame	512
Cmn Wd Len	16
Left Aligned	N
SFID Start	0

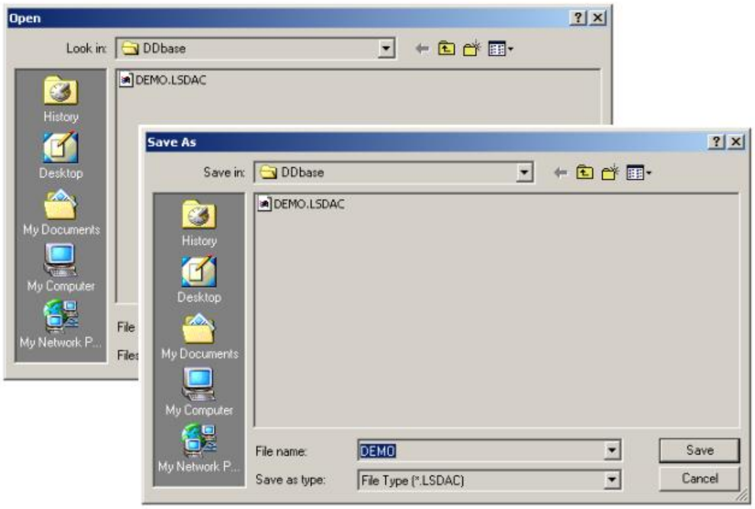
Channel	Source	Fr	Fr Int
1	D-1 : WSel-1	0	1

Figure 3-70 The LS-71 Configuration Menus

The LS-71 hardware configuration display shown above in Figure 3-70 has four menu items: “File,” “Ref Voltage,” “Calibrate” and “Mark.” Both “File” and “Ref Voltage” have sub-menus, the others don’t. All of the LS-71 hardware configuration functions are described in more detail in the following paragraphs.

3.8.4.1 File Menu

The File Menu options are shown in Figure 3-71 on page 175. The File Menu allows the user to “Recall” previously defined LS-71 hardware configurations for streams in the system. The configuration(s) may be used as is, or the user may change and “Save” the configuration changes to the current file, or “Save As” to a new/different file. After a configuration file has been recalled and/or saved, the name of the file will appear in the title bar of the window as shown in Figure 3-70 above left (red oval). Note the file extension (.LSDAC) for the LS-71 hardware configuration file.


Open

Look in: DDbase

DEMO.LSDAC

Save As

Save in: DDbase

DEMO.LSDAC

File name: DEMO

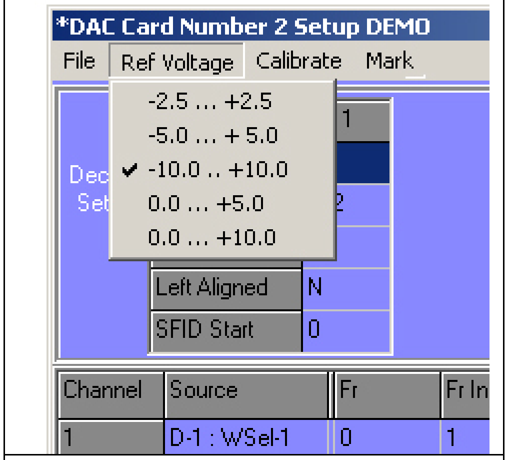
Save as type: File Type (*.LSDAC)

Figure 3-71 File Menu Options

3.8.4.2 Reference Voltage Menu

The Reference Voltage Manu shown right allows the user to select the output voltage range for all channels of a specific LS-71 card. Both unipolar and bipolar output voltage ranges are available. The available ranges listed below include:

1. –2.5 Vdc to + 2.5 Vdc
2. 5.0 Vdc to + 5.0 Vdc
3. –10.0 Vdc to + 10.0 Vdc
4. 0 Vdc to + 5.0 Vdc
5. 0 Vdc to + 10.0 Vdc


*DAC Card Number 2 Setup DEMO

Left Aligned: N

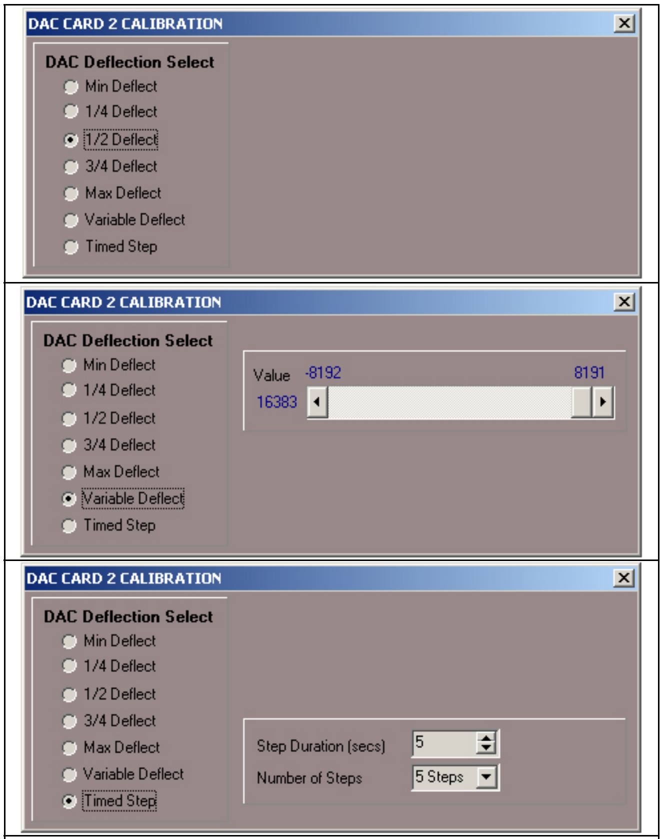
SFID Start: 0

Channel	Source	Fr	Fr In
1	D-1 : WSel-1	0	1

Figure 3-72 Reference Voltage Menu

3.8.4.3 Calibrate Function

The Calibrate function is used to drive all the channels on a particular LS-71 card to certain positions. This is done to align and/or adjust the behavior of strip chart recorders or other such devices used to monitor the analog signals produced by the LS-71. Several permutations of the "DAC CARD X CALIBRATION" window are shown below in Figure 3-73. The particular window displayed will be a function of the type of Deflection mode selected.



DAC CARD 2 CALIBRATION

DAC Deflection Select
○ Min Deflect
○ 1/4 Deflect
● 1/2 Deflect
○ 3/4 Deflect
○ Max Deflect
○ Variable Deflect
○ Timed Step

DAC CARD 2 CALIBRATION

DAC Deflection Select
○ Min Deflect
○ 1/4 Deflect
○ 1/2 Deflect
○ 3/4 Deflect
○ Max Deflect
● Variable Deflect
○ Timed Step

Value -8192 8191
16383

DAC CARD 2 CALIBRATION

DAC Deflection Select
○ Min Deflect
○ 1/4 Deflect
○ 1/2 Deflect
○ 3/4 Deflect
○ Max Deflect
○ Variable Deflect
● Timed Step

Step Duration (secs) 5
Number of Steps 5 Steps

Figure 3-73 Calibrate Menus

As shown above, the user may select one of the following deflection modes:

1. Min Deflect – Drives all outputs to 0, -10, or –5 volts, the minimum output of the LS-71 as specified by the Ref Voltage (see paragraph 3.8.4.2 on page 175).
2. ¼ Deflect – Drives all outputs to one fourth of the maximum output of the LS-71.
3. ½ Deflect - Drives all outputs to one half of the maximum output of the LS-71as specified by the Ref Voltage.
4. ¾ Deflect - Drives all outputs to three fourths of the maximum output of the LS-71as specified by the Ref Voltage.
5. Max Deflect - Drives all outputs to the maximum output of the LS-71as specified by the Ref Voltage.

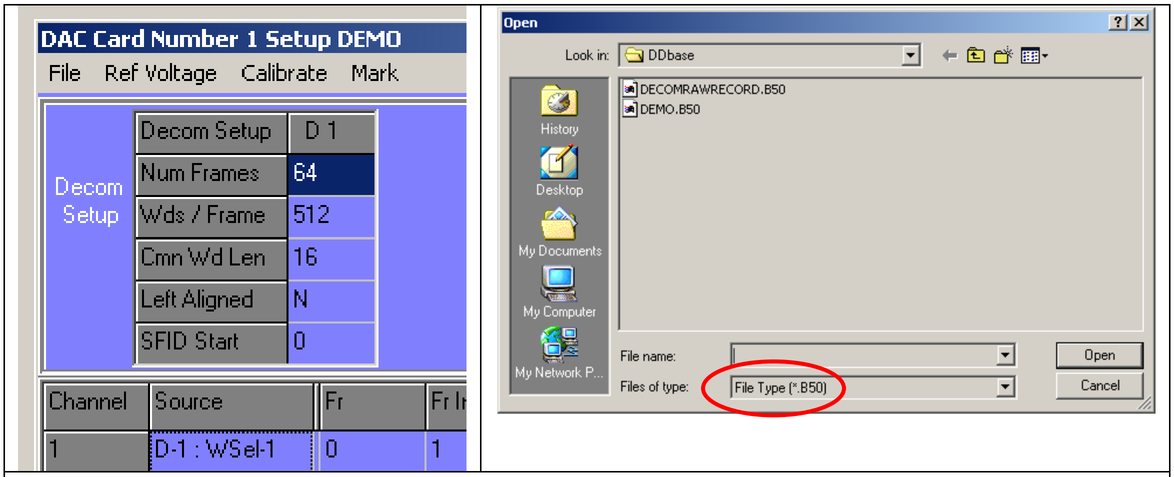
6. Variable Deflect – Displays a slider bar that allows the user to control the output to any range between minimum deflection and maximum deflection (as specified by the Ref Voltage).
7. Timed Step – Allows the user to set the duration in seconds and the number of steps to deflect the channels from minimum deflection to maximum deflection.

3.8.4.4 Mark Control

Selecting the “Mark” item will deflect the output to 0 scale, then ¾ scale, then 0 scale for each channel on all LS-71 cards installed.

3.8.4.5 Decom Setup

Below the menus in the LS-71 hardware configuration display shown below left in Figure 3-74 is the decommutator information and setup area. This area displays the decom setup for a particular stream number. The user may individually enter/modify each of the parameters shown in the display and save them permanently using the File command (see paragraph 3.8.4.1 on page 174). The user may also recall a previously defined LS-50 decom setup file by double clicking in decom setup area or right clicking and selecting the “Set To Decom Setup” button. The resulting file dialog box is shown below right in Figure 3-74 (red oval). Note the file extension (.B50) for the LS-50 decommutator.



Decom Setup	D 1
Num Frames	64
Wds / Frame	512
Cmn Wd Len	16
Left Aligned	N
SFID Start	0

Channel	Source	Fr	Fr In
1	D-1 : WSel-1	0	1

Look in: DDbase

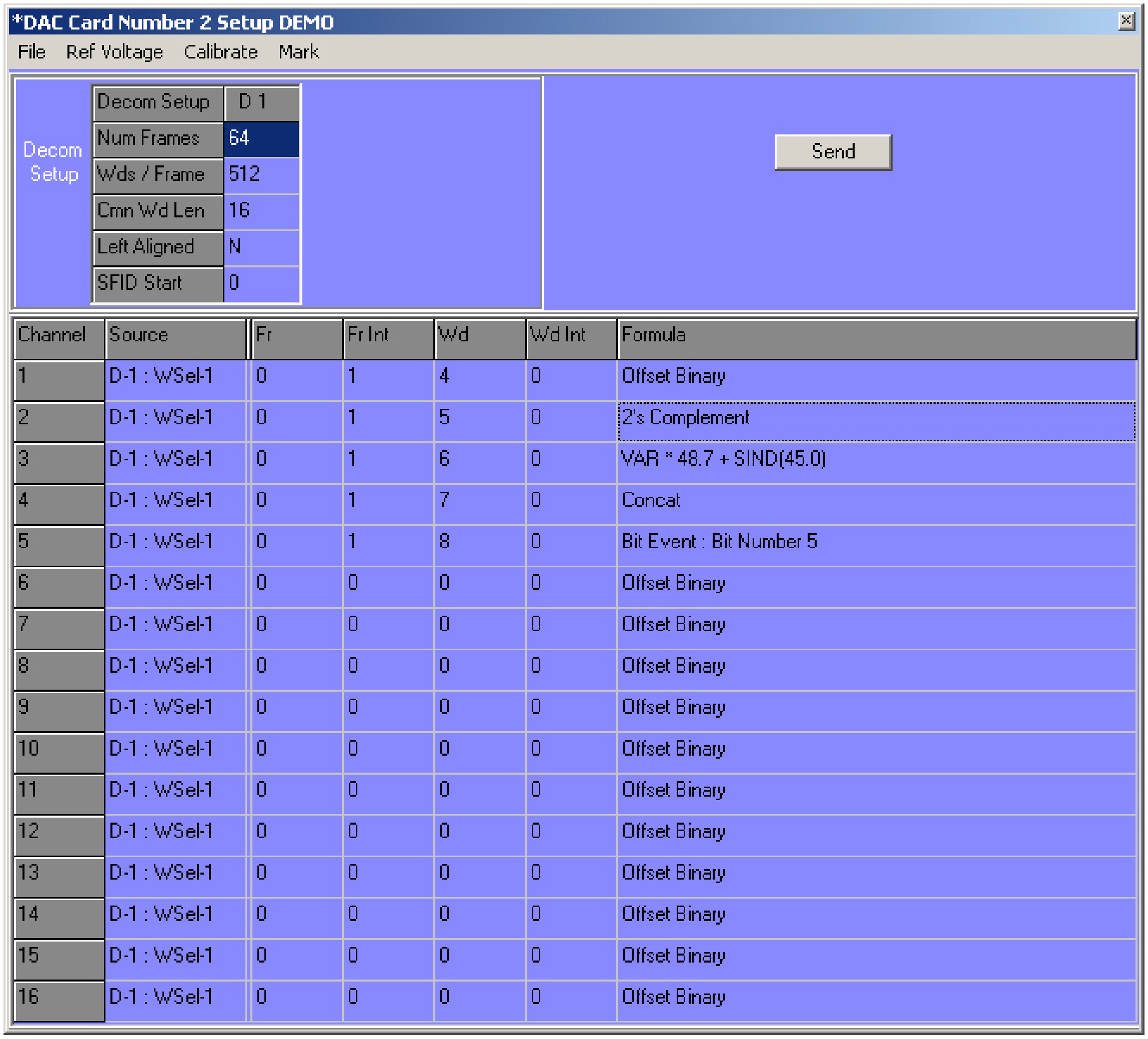
DECOMRAWRECORD.B50
DEMO.B50

Files of type: File Type (*.B50)

Figure 3-74 Decom Setup

The parameters displayed in the decom setup area include:

- Num Frames – Set to the number of minor frames per major frame.
- Words Per Frame – Set to the number of words per minor frame.
- Common Word Length – Set to the number of bits in most of the words for the decommutator stream.
- Left Aligned – Indicate (Y/N) here if the output of the decommutator is set for left aligned data
- SFID Start – The SFID counts from 0 or 1



Decom Setup	D 1
Num Frames	64
Wds / Frame	512
Cmn Wd Len	16
Left Aligned	N
SFID Start	0

Channel	Source	Fr	Fr Int	Wd	Wd Int	Formula
1	D-1 : WSel-1	0	1	4	0	Offset Binary
2	D-1 : WSel-1	0	1	5	0	2's Complement
3	D-1 : WSel-1	0	1	6	0	VAR * 48.7 + SIND(45.0)
4	D-1 : WSel-1	0	1	7	0	Concat
5	D-1 : WSel-1	0	1	8	0	Bit Event : Bit Number 5
6	D-1 : WSel-1	0	0	0	0	Offset Binary
7	D-1 : WSel-1	0	0	0	0	Offset Binary
8	D-1 : WSel-1	0	0	0	0	Offset Binary
9	D-1 : WSel-1	0	0	0	0	Offset Binary
10	D-1 : WSel-1	0	0	0	0	Offset Binary
11	D-1 : WSel-1	0	0	0	0	Offset Binary
12	D-1 : WSel-1	0	0	0	0	Offset Binary
13	D-1 : WSel-1	0	0	0	0	Offset Binary
14	D-1 : WSel-1	0	0	0	0	Offset Binary
15	D-1 : WSel-1	0	0	0	0	Offset Binary
16	D-1 : WSel-1	0	0	0	0	Offset Binary

Figure 3-75 LS-71 Setup Dialog Box

3.8.4.6 Channel Setup

The grid on the lower portion of the LS-71 setup dialog box shown in Figure 3-75 above displays the information the user sets up for each channel on the LS-71 card. The LS-71 channel numbers are listed in the left column. The six parameters along the top of the grid have the following meaning:

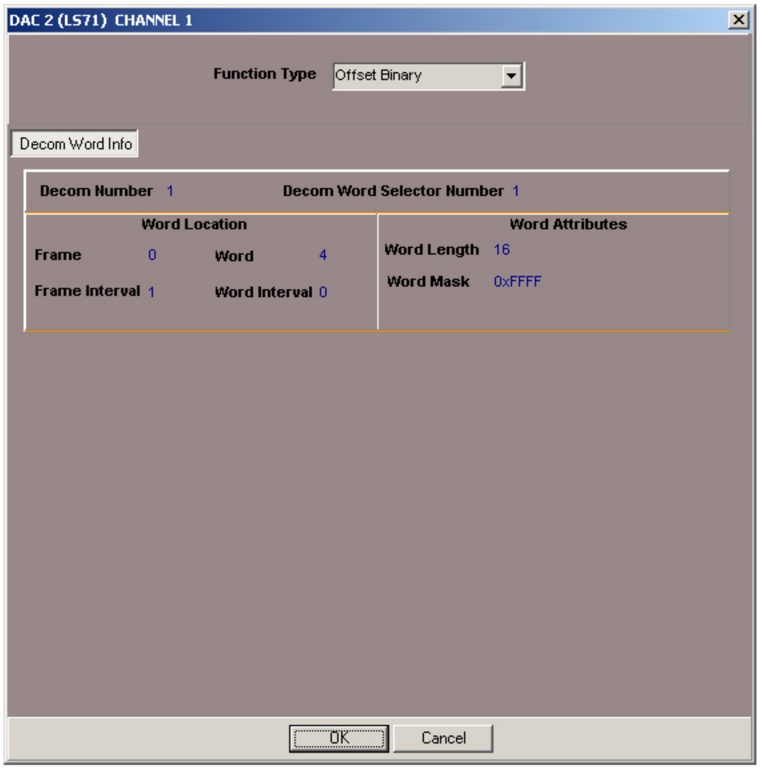
- Source – The source of the data feeding the channel. This is the data coming from the decommutator word(s) and the word selector.
- Fr – The minor frame number of the word driving the LS-71, as configured by the decommutator.
- Fr Int – The minor frame interval of the word driving the LS-71, as configured by the decommutator.
- Wd – The word number of the word driving the LS-71, as configured by the decommutator.

- Wd Int – The word interval of the word driving the LS-71, as configured by the decommutator.
- Formula – This is the formula, or function type applied to the LS-71.

3.8.4.7 Edit Channel Dialog Box

To set up each channel on the LS-71 card with the required configuration, select the channel number on the Setup Display, and double click, or right click on the channel number and click the “Edit Channel” button that results. Either way, a new channel setup window will be displayed as shown in Figure 3-36 on page 179. The Edit Channel dialog box is where the user specifies the source of the LS-71 data, the function to be applied to the channel, the decommutator word selection(s), any masks to be applied, and any formulas to be applied.

NOTE: For all channels on a LS-71 card, a decommutator word can only be used once, for all channels. They cannot be repeated. For instance, the user cannot use Word 1 of Frame 3 in channel 2 and for channel 7.



DAC 2 (LS71) CHANNEL 1

Function Type: Offset Binary

Decom Number 1
Decom Word Selector Number 1

Word Location
Frame 0
Word 4
Frame Interval 1
Word Interval 0

Word Attributes
Word Length 16
Word Mask 0xFFFF

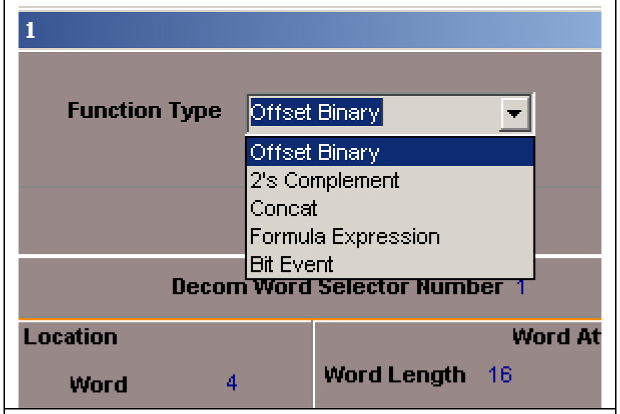
Figure 3-76 Edit Channel Dialog Box

The Edit Channel Dialog Box elements of Figure 3-76 above are as follows:

- **Function Type** – The selected channel can have the data output from the decommutator adjusted by several methods (see paragraph 3.8.4.8 on page 180).
- **Decommutator Word Information** – The user selects a word location from the decommutator that the function type described above will be applied to. For the Concatenate function type (see paragraph 3.8.4.10 on page 181), the user selects two decommutator words. The user will also select the Word Selector number that the data coming into the LS-71 will use. The specific decommutator word information that the user will specify are as follows:
  a) **Frame** – The minor frame number of the selected decommutator word, as defined in the decommutator setup.
  b) **Frame Interval** – The minor frame interval of the selected decommutator word, as defined in the decommutator setup.
  c) **Word** – The word number of the selected decommutator word, as defined in the decommutator setup.
  d) **Word Interval** – The word interval of the selected decommutator word, as defined in the decommutator setup.
  e) **Word Length** – The number of bits in the selected decommutator word, as defined in the decommutator setup.
  f) **Word Mask** – The word mask is applied to the selected decommutator word and is entered as a hex value. Also if a math function will use a bit in the decommutator word, the user should ensure that the bit is also set in the mask. (To use all bits, set the mask to 0xFFFF)

3.8.4.8 Word Function Types

The specified channel can have the data output from the decommutator adjusted by several methods and expressed in several formats as follows:



Function Type: Offset Binary

Decom Word Selector Number 1

Word 4

Word Length 16

Figure 3-77 Word Function Types

1. Offset Binary – Essentially the raw output data from the decommutator applied to the LS-71 after any defined mask is applied.
2. Twos Compliment – The raw output data from the decommutator is expressed in twos compliment format after any defined mask is applied.
3. Concatenate – Two decommutator words are concatenated as follows: ((Word1 & Mask1) << (bits in Word2)) + (Word2 & Mask2)
4. Formula – An extremely powerful capability where the user selects a specific word from the decommutator and applies a formula using the math engine (see

paragraph 3.8.4.11 on page 182). The formula is applied to the decommutator output prior to going to the LS-71.

5. Bit Event – This is a simple method to deflect the channel to ¾ scale if a specific bit is set to TRUE (1).

3.8.4.9 Offset Binary & 2's Complement Word Formats

Offset Binary Representation is a method for representing signed numbers and is popular in A/D and D/A conversions. To represent a binary number as offset binary, begin calculating by assigning half of the largest possible number as the zero value. A positive integer then is represented as the absolute value of the integer added to the zero number. A negative integer is represented as the absolute value of the integer subtracted from the zero number.

Signed Integer	Offset Binary
+5	1000 0101
+4	1000 0100
+3	1000 0011
+2	1000 0010
+1	1000 0001
0	1000 0000
-1	0111 1111
-2	0111 1110
-3	0111 1101
-4	0111 1100
-5	0111 1011

For example:

Largest value for 8-bit integer = 2^8 = 256
Offset binary zero value = 256÷2 =128(decimal) =1000 0000(binary)

The Two's complement representation allows the use of binary arithmetic operations on signed integers, yielding the correct 2's complement results. Positive 2's complement numbers are represented as the simple binary. Negative 2's complement numbers are represented as the binary number that when added to a positive number of the same magnitude equals zero. Note: The most significant (leftmost) bit indicates the sign of the integer; therefore it is sometimes called the sign bit.

Integer		2's Complement
Signed	Unsigned	
5	5	0000 0101
4	4	0000 0100
3	3	0000 0011
2	2	0000 0010
1	1	0000 0001
0	0	0000 0000
-1	255	1111 1111
-2	254	1111 1110
-3	253	1111 1101
-4	252	1111 1100
-5	251	1111 1011

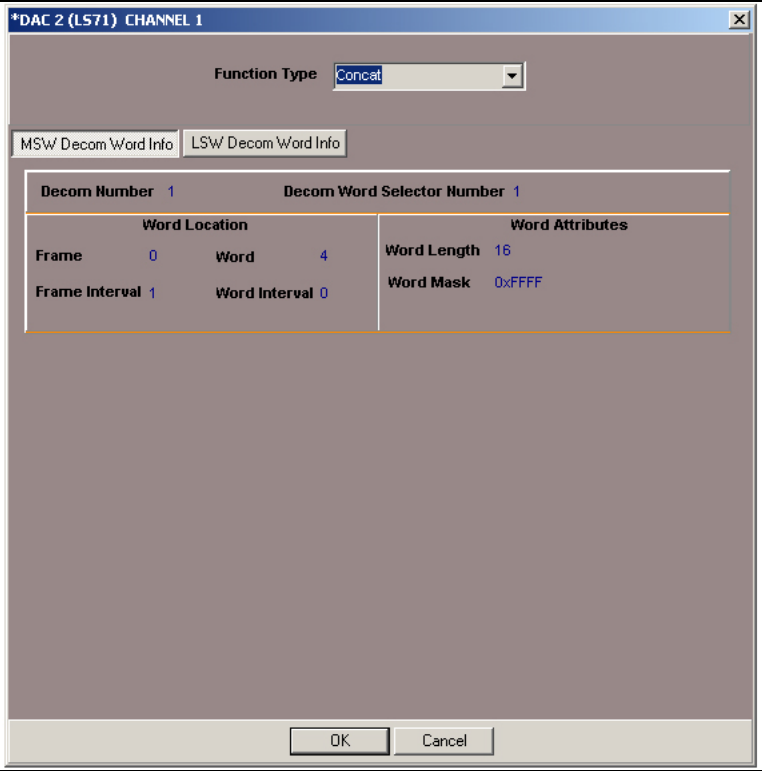
To calculate the 2's complement of an integer, invert the binary equivalent of the number by changing all of the ones to zeroes and all of the zeroes to ones (also called 1's complement), and then add one. If the sign bit is zero, then the number is greater than or equal to zero, or positive. If the sign bit is one, then the number is less than zero, or negative.

3.8.4.10 Concatenate Function

The Concatenate Function dialog box is shown in Figure 3-78 on page 182. The resulting concatenation stores the result in the MSWord or LSWord selected by the user and is then output. The two decommutator words are concatenated as follows:

((Word1 & Mask1) << (bits in Word2)) + (Word2 & Mask2)

Note: “<<” in the above expression denotes left shift N number of bits, where N=word length.



Function Type Concat

Decom Number 1

Decom Word Selector Number 1

Frame 0

Word 4

Frame Interval 1

Word Interval 0

Word Length 16

Word Mask 0xFFFF

Figure 3-78 Concatenate Dialog Box

3.8.4.11 Formula Expression Dialog Box

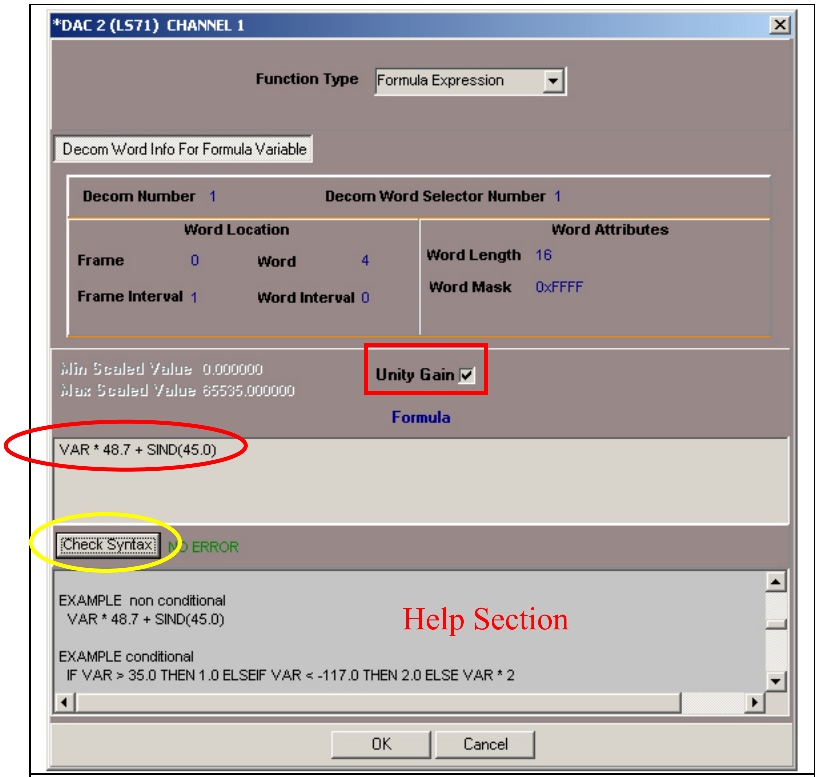
The Formula Expression dialog box is shown in Figure 3-79 on page 183. In addition to the decommutator information (see paragraph 3.8.4.7 on page 179), the user also enters the formula that will be applied to the selected decommutator word before it is output. There is a help window on the bottom to assist the user in entering the formula. In this example, the formula entered is indicated by the red oval.

In specifying the formula, the decommutator word becomes the variable. The variable name is “VAR.” For example, to enter a simple mx + b linear equation, the user would type “VAR * 2.0 + 0.777,” where 2.0 is m, VAR is x, and 0.777 is b. The math engine will also allow the user to enter conditional statements.

To verify the syntax of the formula, click the Syntax Check button (yellow oval in the figure). This will check to ensure that parenthesis and other syntax rules are followed. If the formula is valid, then NO ERROR will be displayed next to the button. If the syntax is invalid, then the error message will be displayed next to the button and the cursor will be placed in the position in the formula the error was encountered.

For certain calculated parameters, the resultants are often compared to minimum and maximum values of the parameter. For example if the decommutator word represents a pitch angle, the user might specify –90.0 degrees for the minimum value. The result of the calculation will produce a minimum deflection on the LS-71 channel when –90.0 or less is solved.

To enter the Min and Max Scaled values, place the mouse cursor next to the labels and right click. Select the “Minimum Scaled Value” or “Maximum Scaled Value” and enter the values in the input box. Note, that the Min and Max Scaled values may not be entered if the “Unity Gain” check box is selected.



*DAC 2 (LS71) CHANNEL 1

Function Type: Formula Expression

Decom Word Info For Formula Variable

Decom Number 1 Decom Word Selector Number 1

Word Location
Frame 0 Word 4
Frame Interval 1 Word Interval 0

Word Attributes
Word Length 16
Word Mask 0xFFFF

Min Scaled Value 0.000000
Max Scaled Value 65535.000000

Unity Gain ☑

Formula

VAR * 48.7 + SIND(45.0)

NO ERROR

EXAMPLE non conditional
VAR * 48.7 + SIND(45.0)

EXAMPLE conditional
IF VAR > 35.0 THEN 1.0 ELSEIF VAR < -117.0 THEN 2.0 ELSE VAR * 2

Figure 3-79 Formula Expression Dialog Box

If the user wishes to normalize the results of the calculation, then the “Unity Gain” check box should be selected (red rectangle in the figure above). When selected, the application will solve for the Max Scaled Value and the Min Scaled Value. In this scenario, the Max Scaled Value will become ($2^{numbits}$) – 1, and the Min Scaled Value will become 0.0.

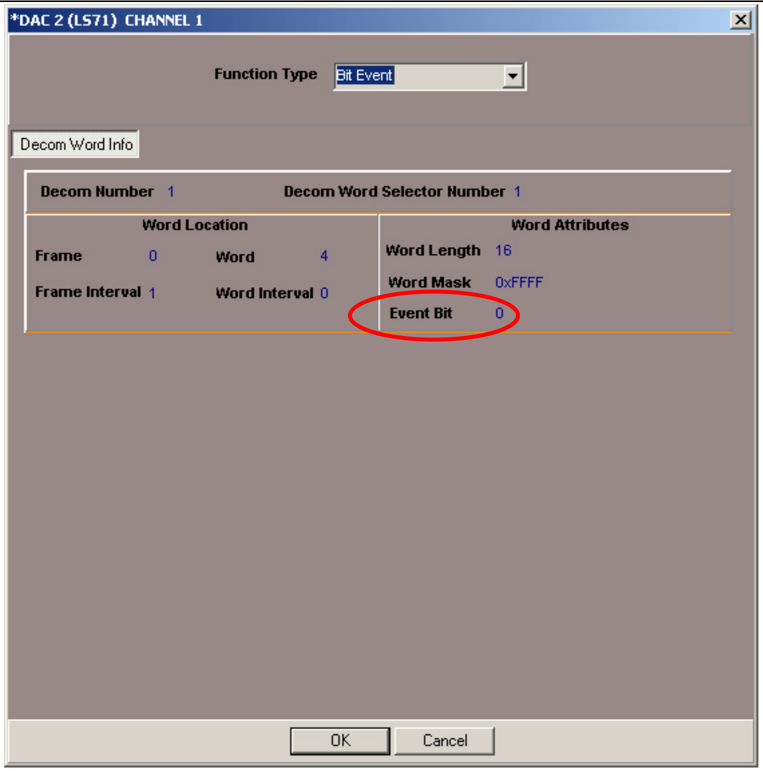
The processing of the channel prior to output to the LS-71 using this function type can be summarized as follows:

1. The selected decommutator word is collected from the decommutator.
2. The selected decommutator word is logically “anded” with the mask bits specified by the user.

3. Prior to the load process of the LS-71, the formula entered by the user is pre-solved by the math engine for every possible bit combination of the raw decommutator word (which is the word length of the word, or 14 bits, which ever is less). The individual results are populated in the look-up table during the load process and during operation. The value derived in step #2 above will drive the look-up table.
4. The result of step #3 above is multiplied by the scale factor determined by the result of Max Scaled Value – Min Scaled Value.
5. The result is output to the LS-71 channel.

3.8.4.12 Bit Event Dialog Box

The Bit Event dialog box is shown in Figure 3-80 below. When the Bit Event function type is selected, the Event Bit field is added to the Word Attributes area as indicated below via the red oval. The Bit Event function is a simple method to deflect the channel to ¾ scale if a specific bit is set to TRUE (1). To select the bit number that will trigger the event, right click in the Word Attributes area and select Event Bit from the list. Enter the bit position in the resulting dialog box and click OK.



*DAC 2 (LS71) CHANNEL 1

Function Type: Bit Event

Decom Word Info

Decom Number 1
Decom Word Selector Number 1

Word Location
Frame 0
Word 4
Frame Interval 1
Word Interval 0

Word Attributes
Word Length 16
Word Mask 0xFFFF
Event Bit 0

Figure 3-80 Bit Event Dialog Box

4 Database Editing in LDPS

Database editing in LDPS consists of look-up tables, and parameter databases. LDPS makes use of three different types of parameter databases. These include IRIG-106 Chapter 4 (or PCM), IRIG-106 Chapter 8, and MIL-STD-1553. Each of database editors is discussed in detail in the following numbered paragraphs.

4.1 Lookup Table Edit

Some parameter databases may require a lookup table to process some or all of the parameters in the database. Each parameter database can only have one look-up table file associated with it. Within the file, there can be up to 48 individual look-up tables, with each look-up table having up to 64 X/Y elements.

4.1.1 Lookup Table Processing

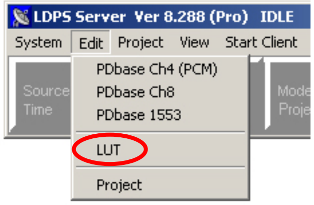
Parameters that use a look-up table have special processing performed on them. There are no standards for defining how look-up table processing should be implemented. For telemetry streams that transmit data from non-linear controls, the look-up table methodology employed in LDPS works very well. The steps involved with look-up table processing of a parameter in LDPS are as follows:

```
    rvalue = BreakoutDiscretes(thedata.Msb, thedata.Lsb, thedata.ParamNumBits,rawvalue);
    rvalue =  GetTwosComp(rvalue, thedata.ParamNumBits);
    //scale it
    result = rvalue * thedata.Euc;
    //extrapolate it
    result = ExtrapolateLut(thelutary[thedata.LutNum].NumElements,
                            thelutary[thedata.LutNum].Xary,
                            thelutary[thedata.LutNum].Yary,
                            result);
    //offset it
    result = result + thedata.PlusB;
```

The ExtrapolateLut() call processes the data as follows:

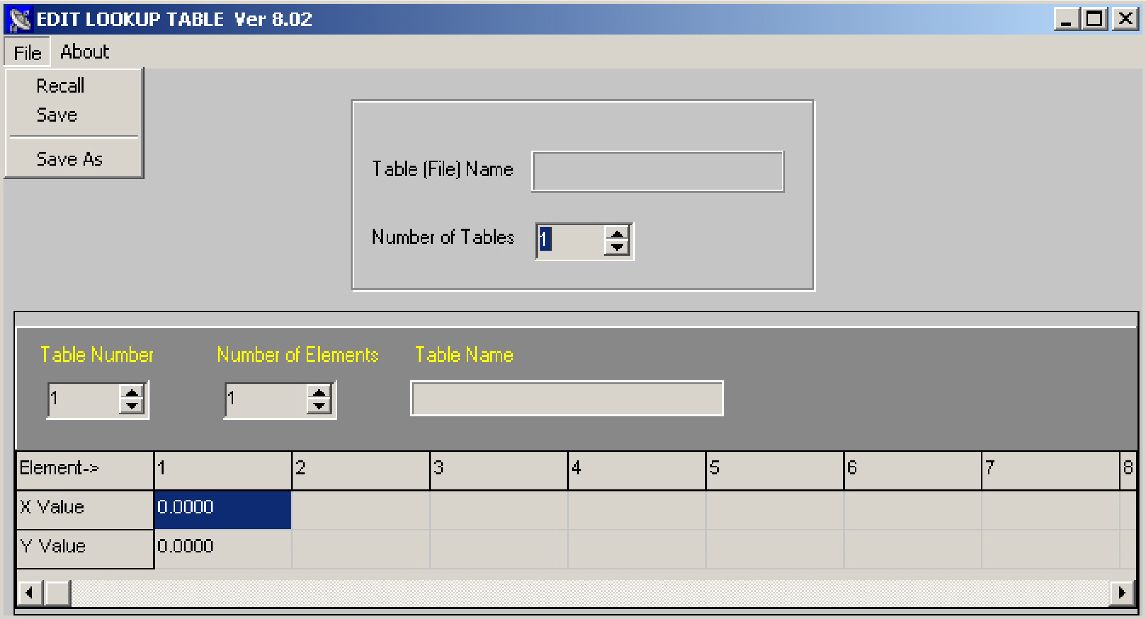
```
  if (xarg <= xd[0])
    result = yd[0];
  else if (xarg >= (xd[len-1]))
    result = yd[len-1];
  else
  {
    for (a=0; a < len; a++)
    {
      if (xd[a+1] > xarg)
      {
        found = true;
        break;
      }
    }
    if (found)
    {
      x3 = xarg;
      x1 = xd[a];
      y1 = yd[a];
      x2 = xd[a+1];
      y2 = yd[a+1];
      y3 = (x3-x2) * (y2-y1) / (x2-x1) + y2;
      result = y3;
    }
    else result = yd[len-1];
  }
```

4.1.2 Editing & Saving Look-Up Tables



To invoke the Look-Up Table (LUT) editor, select LUT from the Edit menu of the LDPS server as shown right. The resulting window is shown in Figure 4-1 below.

The **File** menu allows the user to save and recall look-up table files. The current file being edited is displayed in the Table Name box. If no name is present, then upon saving, the user will be for prompted for a file name. Below the table name is a spin dial that sets the total number of tables in the file. Recall that up to 48 individual look-up tables may be supported by one file. When a parameter is edited that references this file in the database, the look-up table number must correspond with this file, or the processed result will yield a Zero value. See paragraph 4.3.1.2 on page 192 for more details.



Number of Tables: 1

Table Number: 1

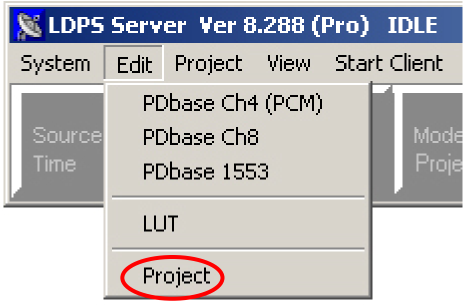
Number of Elements: 1

Element->	1	2	3	4	5	6	7	8
X Value	0.0000							
Y Value	0.0000							

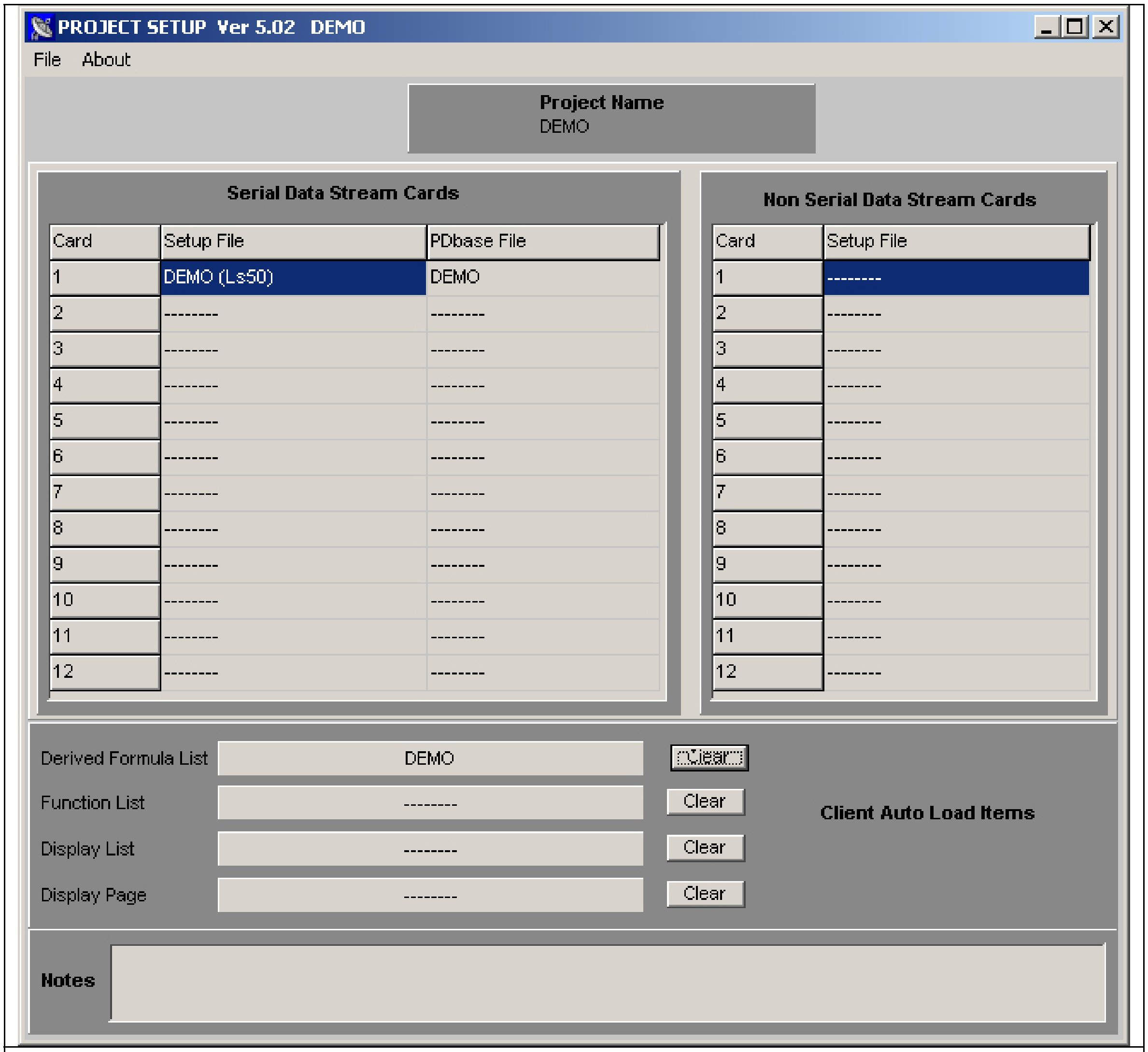
Figure 4-1 Edit Lookup Table Window

In the box below the look-up table file name there are two spin dials used to specify the table number and the number of elements in the table. To the right of the spin dials is the name of the look-up table (not to be confused with the name of the *file* containing the look-up table). At the bottom of the window are the actual the X and Y elements in the table. Here, in the cells of the table, the user enters the decimal values used for each element. Note: The first X value in the table must be the smallest X value in the list, with subsequent X values increasing in value. In other words, sort the X/Y pairs in increasing values of X. Otherwise expect things to fail. Once editing of the look-up tables is complete, make sure and save the file.

4.2 Project Edit



In order for data to be archived, processed and displayed, a project must be loaded. In LDPS, a project is defined as a group of items associated with a particular test. The items may include: Serial and Non-Serial data acquisition cards, parameter database files, configuration & setup files, function and formula lists, display lists, etc. To invoke the project editor, select **Project** from the Edit menu of the LDPS server as shown right. The resulting window is shown in Figure 4-2 below.



Project Name
DEMO

Serial Data Stream Cards

Card	Setup File	PDbase File
1	DEMO (Ls50)	DEMO
2	--------	--------
3	--------	--------
4	--------	--------
5	--------	--------
6	--------	--------
7	--------	--------
8	--------	--------
9	--------	--------
10	--------	--------
11	--------	--------
12	--------	--------

Non Serial Data Stream Cards

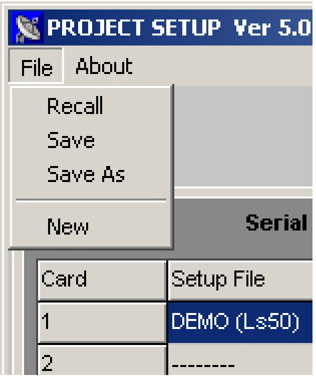
Card	Setup File
1	--------
2	--------
3	--------
4	--------
5	--------
6	--------
7	--------
8	--------
9	--------
10	--------
11	--------
12	--------

Derived Formula List: DEMO [Clear]
Function List: -------- [Clear]
Display List: -------- [Clear]
Display Page: -------- [Clear]

Client Auto Load Items

Notes

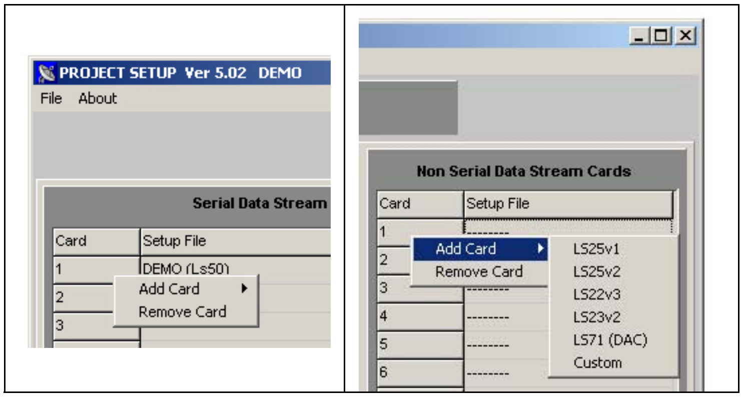
Figure 4-2 Project Setup Window



Card	Setup File
1	DEMO (Ls50)
2	--------

The **File** menu allows the user to save and recall project files. The current project being edited is displayed in the Project Name box. If no name is present, then upon saving, the user will be for prompted for a file name. To start a new project, select the **New** command. This will reset all of the fields in the project editor a blank state.

LDPS can support up to twelve (12) Serial Data Stream Cards, with each card having a configuration/setup file, and a parameter database associated with it. To add a serial data stream card, click in the empty cell to the right of a card number (see below left) and then right click and select "Add Card" from the menu. Select LS50, 1553, or Custom from the list, and then select the setup file from the file list dialog box that appears. After specifying a setup file, the user will be prompted to enter the file name of the parameter database file associated with the serial device. Repeat this procedure for each serial data stream device installed in the system.



Card	Setup File
1	DEMO (Ls50)
2	
3	

Card	Setup File
1	--------
2	
3	--------
4	--------
5	--------
6	--------

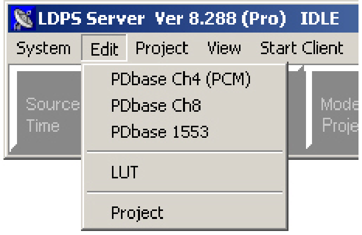
LDPS can also support up to twelve (12) Non-Serial Data Stream Cards, with each card having a configuration/setup file associated with it. To add a non-serial data stream card, click in the empty cell to the right of a card number (see above right) and then right click and select "Add Card" from the menu. Select one of the cards from the list [currently the cards available include: LS25V1, LS25V2, LS22V3, LS23V2, LS71 (DAC)], or Custom from the list, and then select the setup file from the file list dialog box that appears.

To automatically load onto the client the derived formula and DLL function lists when a project is loaded, click the **Derived Formula List** and **Function List** respectively and select the appropriate files from the file list dialog box that appears. Click on the **Clear** button to remove the selection. The **Display List** and **Display Page** can also be automatically loaded onto the client when a project is loaded. Use the appropriate button, or right click and select one of the four list functions from the dropdown menu that appears.

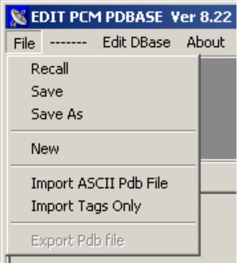
At the bottom of the project setup window is an area intended for the user to enter notes about the project.

Note: In order to archive data, a project must be loaded. At a minimum, a project must contain at least one serial data stream. The serial data stream must contain a hardware setup file and a parameter database file. Finally, the parameter database must contain at least one parameter.

4.3 Parameter Database Editor



The parameter database is the most critical part of the system, and is generally the most difficult to create. If it is not correct, or has errors, the data is meaningless. The server application provides a tool for the manual creation and/or maintenance of the parameter database. To invoke the parameter database editor, click the **Edit** command from the main server menu and then select, **PDbase (Ch4 [PCM], Ch8, or 1553)** as shown right.



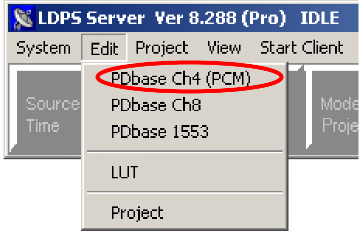
For example, selecting **PDbase Ch4 (PCM)**, will invoke the parameter database editor shown in Figure 4-3 on page 191. Selecting **PDbase Ch8** will invoke the IRIG 106 Chapter 8 parameter database editor shown in Figure 4-13 on page 209. The MIL-STD-1553 parameter database editor, shown in Figure 4-20 on page 219, can be invoked by clicking the **PDbase 1553** command. At the top of the window in Figure 4-3 are two menu items: File and Edit Dbase. The **File** command is used for saving and recalling databases. The database files are stored in two formats, ASCII and binary. When the server and client(s) load a database(s), it is the binary version that is used. To modify the database, use Microsoft Excel or another ASCII text editor to edit the ASCII version of the file. Then **Recall** the ASCII version of the file in the LDPS database editor and **Save** it. This will compile the database and save both the ASCII and the binary versions. The compiled binary version of the file is used when the file recall command is invoked. When the **Import ASCII Pdb File** command is invoked, the ASCII version is recalled.

For those wishing to edit the parameter database in Excel or other ASCII editor, the user may also import just the parameter tags (**Import Tags Only**). The header information is not required for the import. Just have the first line in the ASCII file contain "*.*" and then the rest of the lines are as normal for the ASCII version. Then, use the LDPS parameter editor to set up the header information and save the file. This will yield a compiled version (both binary and ASCII). The user can also export a database to some other format, provided the export DLL assigned in the programs options.

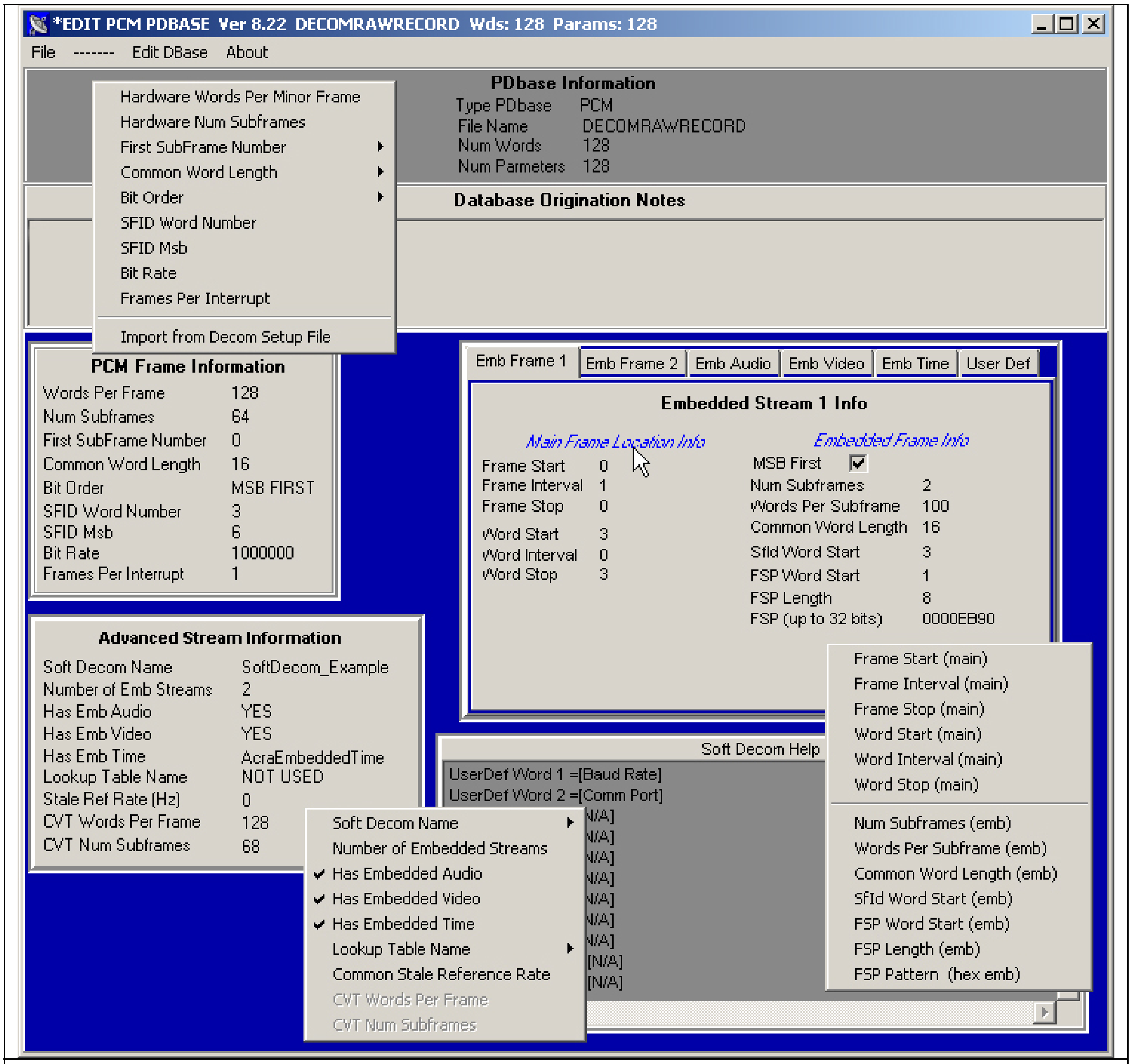
Clicking on the **Edit Dbase** command in Figure 4-3 will result in the Edit PCM Database Window shown in Figure 4-9 on page 200. This window allows editing of the words and parameters in the parameter database.

Below the menu section in Figure 4-3 is information regarding the database that includes: The name of the database loaded for editing (File Name), the count of root words in the database (Num Words), the count of parameters in the database (Num Parameters), and database type (Type Dbase). Included types: PCM, 1553, and IRIG Chapter 8.

4.3.1 IRIG-106 Chapter 4 (PCM) Database



Selecting the **PDbase Ch4 (PCM)** command from the servers' edit menu will invoke the **EDIT PCM PDBASE** window shown in Figure 4-3 below. The major configuration items in this window include: PCM Frame Information, Advanced Stream Information, Setup Tabs for those items enabled in the advanced stream information section, and User Defined Code Words. Each of these configuration items will be described in more detail in the following paragraphs.



PDbase Information

Type PDbase	PCM
File Name	DECOMRAWRECORD
Num Words	128
Num Parameters	128

Database Origination Notes

PCM Frame Information

Words Per Frame	128
Num Subframes	64
First SubFrame Number	0
Common Word Length	16
Bit Order	MSB FIRST
SFID Word Number	3
SFID Msb	6
Bit Rate	1000000
Frames Per Interrupt	1

Embedded Stream 1 Info

Main Frame Location Info

Frame Start	0
Frame Interval	1
Frame Stop	0
Word Start	3
Word Interval	0
Word Stop	3

Embedded Frame Info

MSB First	☑
Num Subframes	2
Words Per Subframe	100
Common Word Length	16
Sfid Word Start	3
FSP Word Start	1
FSP Length	8
FSP (up to 32 bits)	0000EB90

Advanced Stream Information

Soft Decom Name	SoftDecom_Example
Number of Emb Streams	2
Has Emb Audio	YES
Has Emb Video	YES
Has Emb Time	AcraEmbeddedTime
Lookup Table Name	NOT USED
Stale Ref Rate (Hz)	0
CVT Words Per Frame	128
CVT Num Subframes	68

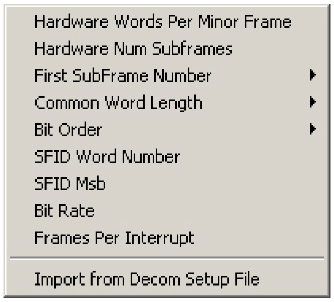
Soft Decom Help

UserDef Word 1 =[Baud Rate]
UserDef Word 2 =[Comm Port]
N/A]
N/A]
N/A]
N/A]
N/A]
N/A]
N/A]
[N/A]
[N/A]

Figure 4-3 Edit Parameter Database Window & Menus (PCM Type)

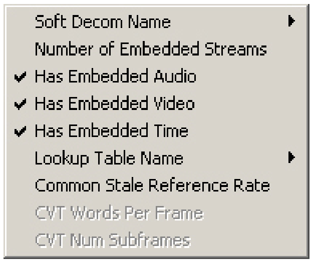
4.3.1.1 PCM Frame Information

These parameters are required to support the CVT and also for some processing. Right click in this area to invoke the menu shown right. The user may either set up the individual parameters required or select **Import from Decom Setup File**. If the latter option is selected, the user will be prompted for the name of the setup file previously created for the LS-50-P decommutator. This option is quite useful, especially if the decommutator word lengths vary (not all the same size).



4.3.1.2 Advanced Steam Information

Right click in this section to invoke the menu shown right. Depending on what is entered here, some advanced tabs will appear where further information will be required.



Select the **Soft Decom Name** command if the stream requires a software decommutator (ordering / manipulation of the main data stream after collection from the device). The user may select from the available software decommutators that have been previously generated. If the selected software decommutator has a help file, it will be displayed below the embedded stream tabs as shown in Figure 4-3 on page 200.

If there are any embedded asynchronous streams in the primary stream (see paragraph 4.3.1.3.1 on page 193), then their quantity is entered by selecting the **Number of Embedded Streams** command. This command is only available if a software decommutator has been selected. If the stream has embedded audio, then check the **Has Embedded Audio** command (see paragraph 4.3.1.3.2 on page 195). This will cause a new tab to appear with additional setup information required. Likewise, if the stream has embedded video or time, then click the **Has Embedded Video** and/or **Has Embedded Time** commands respectively (see paragraph 4.3.1.3.3 on page 196 and paragraph 4.3.1.3.4 on page 197). As before, this will cause new tabs to appear with additional setup information required. For streams with embedded time, the user will be required to enter the name of the time DLL that will support the stream.

If the database requires a look-up table, then the user can select from a list of available look-up tables previously generated by clicking the **Lookup Table Name** command.

The **Common Stale Reference Rate** command sets the rate that most parameters will update at. This command sets the default for all parameters. The state rate is used by the display programs to issue alerts when the data for parameters are not updating at the required rate (based on the time stamp for that parameter).

If a software decommutator is being used, then the **CVT Words Per Frame** and **CVT Num Subframes** values may be adjusted by the user. In both cases, the Number of Embedded Streams cannot be set to be greater than ZERO however. Otherwise, the CVT

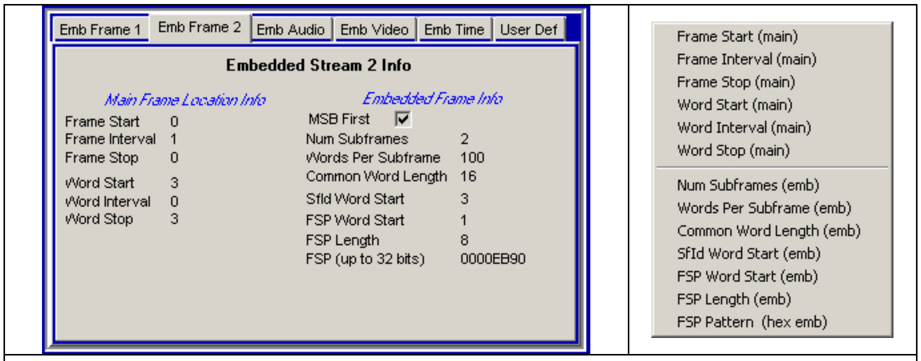
Words Per Frame and CVT Num Subframes values will be filled in by the program. If these parameters are set manually, they cannot be smaller than the PCM frame size.

4.3.1.3 Database Configuration Tabs

Depending on the configuration established in the Advanced Stream Information section described in the previous paragraph, there can be up to seven configuration tabs as shown in Figure 4-3 on page 200. The possible tabs include: configuration for one or two embedded asynchronous streams (see paragraph 4.3.1.3.1 on page 193), configuration setup for embedded audio (see paragraph 4.3.1.3.2 on page 195), video (see paragraph 4.3.1.3.3 on page 196), and time (see paragraph 4.3.1.3.4 on page 197), as well as a tab for user defined code words (see paragraph 4.3.1.3.5 on page 198).

4.3.1.3.1 Embedded Asynchronous Frames Tab

Handling an embedded stream(s) inside a normal PCM stream can sometimes be a complex task. Processing embedded streams can be handled with an extra hardware decommutator. With the LS-50-P decommutator for example, a second hardware decommutator (LS-55-DB) can be used to support one embedded stream. The format of the embedded stream has to be straightforward, with no throw away bits or any processing. If this extra hardware method is chosen, then effectively a second stream will be created for LDPS to process.



Embedded Stream 2 Info

Main Frame Location Info

Frame Start	0
Frame Interval	1
Frame Stop	0
Word Start	3
Word Interval	0
Word Stop	3

Embedded Frame Info

MSB First	☑
Num Subframes	2
Words Per Subframe	100
Common Word Length	16
Sfld Word Start	3
FSP Word Start	1
FSP Length	8
FSP (up to 32 bits)	0000EB90

Frame Start (main)
Frame Interval (main)
Frame Stop (main)
Word Start (main)
Word Interval (main)
Word Stop (main)

Num Subframes (emb)
Words Per Subframe (emb)
Common Word Length (emb)
SfId Word Start (emb)
FSP Word Start (emb)
FSP Length (emb)
FSP Pattern (hex emb)

Figure 4-4 Embedded Asynchronous Frames Tab (PCM Type)

Supporting PCM streams with multiple embedded formats can be accomplished in LDPS by using a “Soft Decommutator.” The software decommutator is as the name implies, it performs the decommutation function in software rather then in hardware. Several soft decoms are supplied in LDPS, but the user if free to develop their own.

On the upper left of the database editor shown in Figure 4-3 on page 191, is the PCM Frame Information section that is primarily used to set up the CVT (current value table). Normally, one sets this information up to match the decommutator settings. With embedded streams however, one typically has to change either the **Words Per Frame** (wpf) or the **Num Subframes** (numsf) or both so that there is enough storage space for the outer PCM frame as well as the embedded frames of data. Embedded streams tend to span multiple PCM outer frames. There must be enough storage for each word in the outer PCM major frame as well as for each word in each of the embedded stream(s) major frame. How the extra storage is created is up to the user (by adjusting the wpf and numsf fields). The wpf and numsf fields must be at least as large as the outer PCM major frame (as set up by the decommutator).

NOTE. The CVT size can only be edited if a Soft Decom is being used. If the Soft Decom is removed, then the CVT size will revert to the size defined in the decommutator settings.

How LDPS handles embedded streams using soft decoms can be a little confusing. To understand this, some background is in order. The server collects raw data from the decommutator, or other serial stream devices, and feeds it to the client, who maintains a copy of the buffer of data. The client processes only the data of interest. Interest is normally assigned by parameters in the display widgets. When the interest list is processed, the client runs through the list, looks at the hardware information for each parameter and sticks the raw data from the buffer it received from the server into the CVT location. The data that is processed comes from the CVT, not from the server buffer. Getting the data into the correct CVT location is done with a soft decom. Note, in the parameter database there are eight hardware fields and several UserDef fields associated with each parameter. For PCM type streams, only six of the hardware fields are used, and none of the UserDef fields are used. These ‘extra’ fields are used by the soft decom to tell where to put the data into the CVT. The six hardware fields used by the soft decom include: Frame Start, Frame Interval, Frame Stop, Word Start, Word Interval, and Word Stop.

These six fields give the soft decom enough information to store the normal outer PCM frame data into the correct CVT location. As to embedded streams, the soft decom knows how to process the data (normal or embedded stream) by using the two extra hardware fields and the eight UserDef fields. The user may choose any method for storing data with these extra fields. In addition to the ‘extra’ eight fields for each parameter, there are an additional sixteen ‘extra’ or UserDef fields in the header of the database. These fields are intended to be used by the soft decom to define things that pertain to how the soft decom will operate.

The IRIG-106 specification states that a Class-II stream may have up to two embedded streams. Unfortunately, the IRIG-106 specification is often deviated from, and in this spirit, LDPS allows the user to have as many embedded streams as they want. However, for the purpose of the discussion, it will be assumed that only two embedded steams will be used. The parameter database editor is set up to assist the user with a database, assuming a two embedded stream limit. If this is not the case the user can still use the database editor, but with some caveats.

On the upper right of Figure 4-3 on page 200 there are a set of tabs with embedded information. LDPS does not use the Emb Frame 1 or the Emb Frame 2 information. The developer of any soft decom is free to use these two tabs as required. If no soft decom is specified from the **Advanced Steam Information** section (see paragraph 4.3.1.2 on page

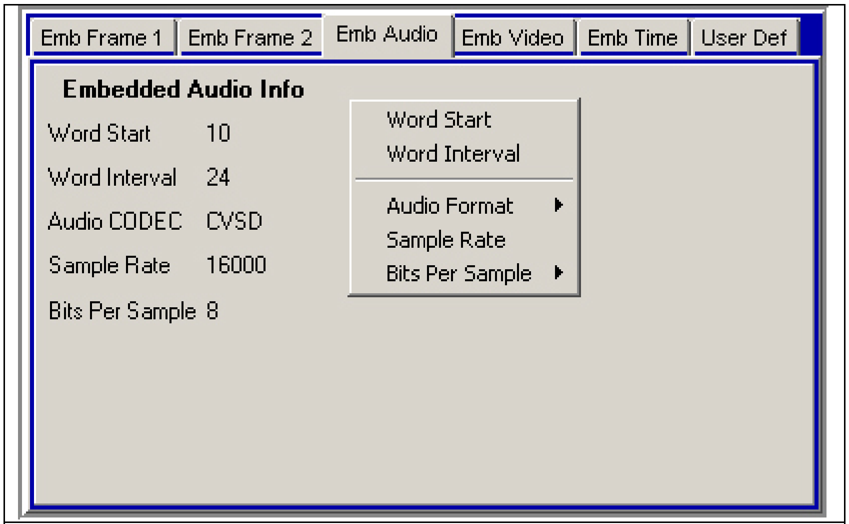
192), then the **Number of Embedded Streams** will be grayed out. When a soft decom is selected, and the number of embedded streams is set to one or more, the parameter database editor will replace the spare hardware fields and the UserDef fields in the editor with text on how a standard embedded stream would use them.

HINT. To help visualize the structure of the CVT, think of the streams, both PCM outer frame and embedded frames, as being separate spreadsheets, one spreadsheet for each stream. Place the outer PCM spreadsheet on top, and align the embedded stream spreadsheet below it so that column #1 of each lines up. If there is a second embedded stream, place that spreadsheet below the first embedded spreadsheet so that all column #1s lines up. Now count the number of rows for all of the spreadsheets. This will become the Num Subframes on the database editor. Next, of all the spreadsheets, set the Words Per frame to the largest number of columns.

The hint above is one suggested method for determining the proper CVT size. Other methods are also possible. Note there are limits to the number of subframes and words per frame (4,096 and 16,384, respectively, and a total of 131,072 cells cannot be exceeded).

4.3.1.3.2 Embedded Audio Tab

The Embedded Audio tab allows the user to set up any embedded audio streams. The **Word Start** indicates the starting word in the primary frame where the embedded audio begins (1 based index).



Embedded Audio Info

Word Start 10

Word Interval 24

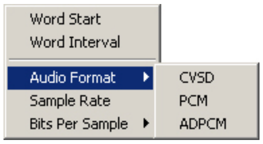
Audio CODEC CVSD

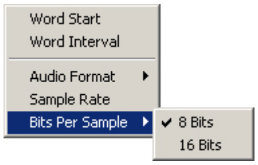
Sample Rate 16000

Bits Per Sample 8

Figure 4-5 Embedded Audio Tab (PCM Type)

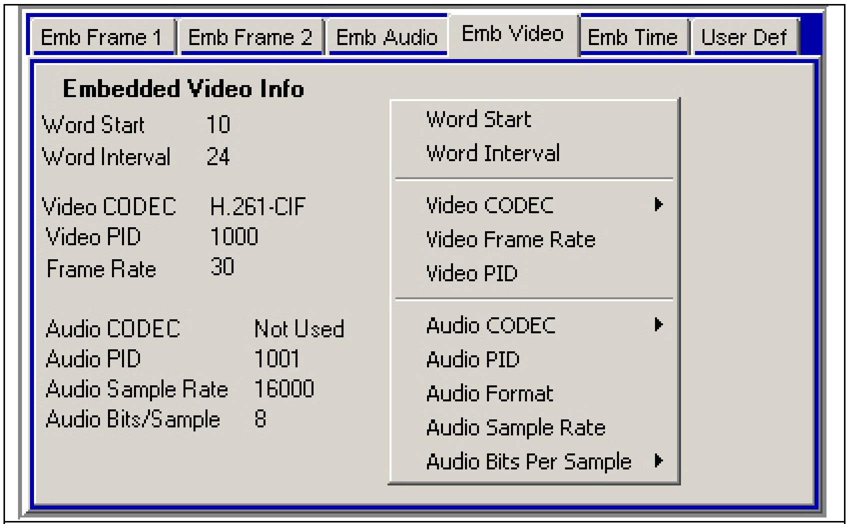
The **Word Interval** indicates how the embedded audio is commutated within the primary frame. The **Audio Format** command selects the encoding format the embedded audio was encoded with (the default is CVSD). Other CODECs include PCM and ADPCM. The **Sample Rate** selects the sample/second that the audio was encoded at (the default is 16 Khz). Eight or sixteen bits/sample may be selected by the user via the **Bits Per Sample** Command.





4.3.1.3.3 Embedded Video Tab

The Embedded Video tab allows the user to set up any embedded video streams. The **Word Start** indicates the starting word in the primary frame where the embedded video begins (1 based index). The **Word Interval** indicates how the embedded video is commutated within the primary frame.



Embedded Video Info

Word Start	10
Word Interval	24
Video CODEC	H.261-CIF
Video PID	1000
Frame Rate	30
Audio CODEC	Not Used
Audio PID	1001
Audio Sample Rate	16000
Audio Bits/Sample	8

Figure 4-6 Embedded Video Tab (PCM Type)

The **Video CODEC** command selects the encoding format the embedded video was encoded with (the default is H.261 - CIF[16]). Some of the other supported formats include: H.264-QCIF,[17] H.264-CIF FEC,[18] MP2[19]-4CIF.[20] The complete list of supported CODECs is shown below left.

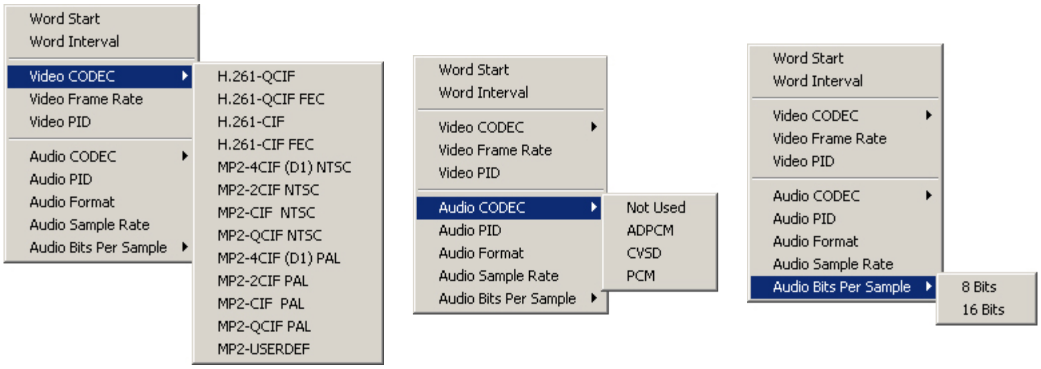
[16] Common Intermediate Format - A standard video format used in videoconferencing. Bit rate at 30 fps is 36.5 Mbps. Resolution is 352 x 288 pixels.

[17] Quarter CIF. Resolution is 176 x 144 pixels. Bit rate at 30 fps is 9.1 Mbps.

[18] Forward Error Coding.

[19] MPEG-2, provides broadcast quality video with resolutions up to 1920x1080.

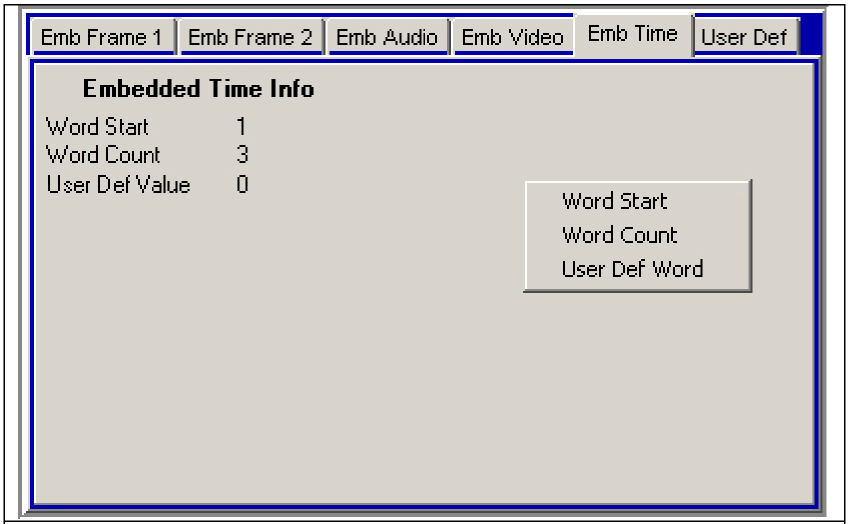
[20] 4CIF (4 x CIF). Resolution is 704 x 576 pixels. Bit rate at 30 fps is 146.0 Mbps.



The **Video Frame Rate** selects the frames/second that the video was encoded at (the default is 30 fps). The **Audio CODEC** command selects the encoding format the audio portion of the embedded video was encoded with (the default is CVSD). Other CODECs include PCM and ADPCM. The **Audio Sample Rate** selects the sample/second that the audio was encoded at (the default is 16 KHz). Eight or sixteen bits/sample may be selected by the user via the **Audio Bits Per Sample** command.

4.3.1.3.4 Embedded Time Tab

For streams that have embedded time, the user can either use the parameter database to collect the time and decode it or use a display widget function. To enable the parameter database to decode the embedded time data, the user must select the appropriate DLL to decode the time. To enable the embedded time functionality in the parameter database, select the **Has Embedded Time** command from the **Advanced Stream Information** menu described in paragraph 4.3.1.2 on page 192. After selecting the file name of the embedded time DLL, a new tab (shown in Figure 4-7 below) will appear.



Embedded Time Info

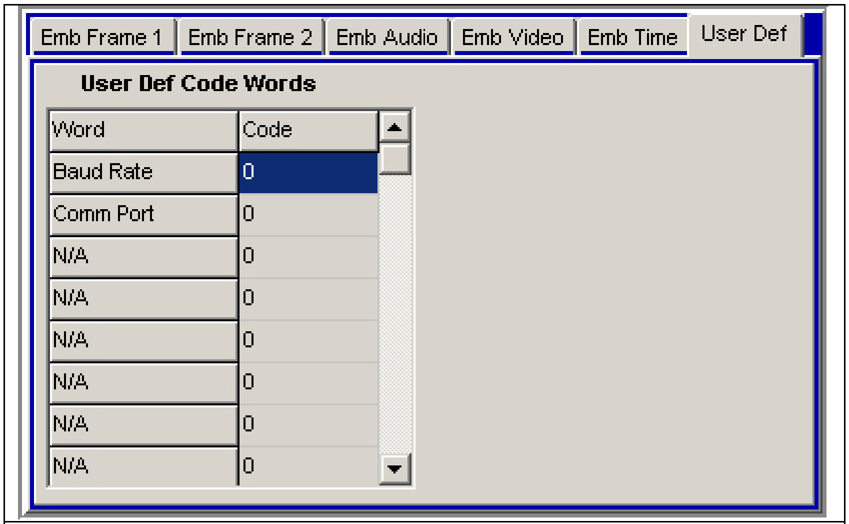
Word Start	1
Word Count	3
User Def Value	0

Figure 4-7 Embedded Time Tab (PCM Type)

The **Word Start** indicates the starting word in the primary frame where the embedded time data begins (1 based index). The **Word Count** indicates how the embedded time is commutated within the primary frame.

4.3.1.3.5 User Defined Code Words Tab

The user-defined code words are integers the user may define as desired. The user-defined code words are not used by LDPS, but instead are intended for use with a software decommutator. For instance, use some of these integers as flags to affect how the software decommutator will treat parameters. If the software decommutator requires the user to enter data, it should also provide a help file that will explain what parameters are to be entered. The help file window is displayed directly below the tabs as shown in Figure 4-3 on page 191. This help file has the extension ".hhlp," and is a simple text file with the same name as the DLL, and located in the same place.



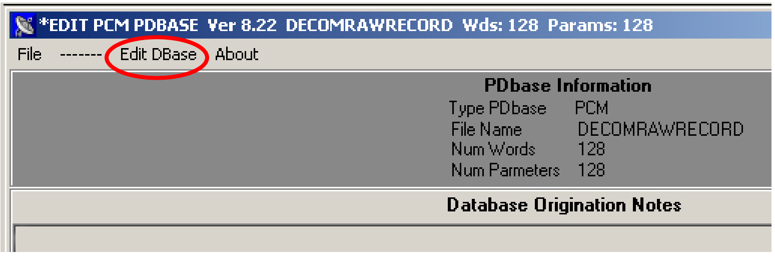
Word	Code
Baud Rate	0
Comm Port	0
N/A	0
N/A	0
N/A	0
N/A	0
N/A	0
N/A	0

Figure 4-8 User Defined Code Words Tab (PCM Type)

The content of the help file is up to the user. For example, if the UserDef code words are to have a label, other than the word number, then the first sixteen lines in the help file must contain the text for the UserDef word number field, surrounded by brackets "[]." For example, if word #1 is to be assigned to the baud rate, then the first line in the help file will contain [Baud Rate]. Any text may be placed before and/or after the brackets for further clarification, but that contained in the brackets will be displayed in the word number label column. If left blank, the word label column will display only the word number.

4.3.1.4 Edit Database Window

To invoke the Edit PCM Database window shown in Figure 4-9 on page 200, click on the **Edit DBase** command in the Edit Parameter Database window shown in Figure 4-3 on page 191 (a cutaway of which is shown below). Note: This editor is intended for simple fixes or parameter additions, not for hand typing hundreds or thousands of parameters.


*EDIT PCM PDBASE Ver 8.22 DECOMRAWRECORD Wds: 128 Params: 128
File

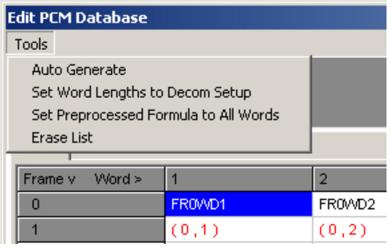
Edit DBase
About

PDbase Information

Type PDbase PCM
File Name DECOMRAWRECORD
Num Words 128
Num Parmeters 128

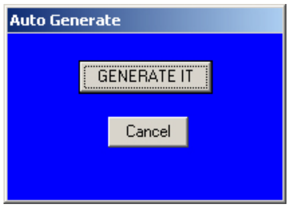
Database Origination Notes

Across the top of the Edit PCM Database window shown right is the main menu, which currently contains the **Tools** command. The database tools include: Auto Generate, Set Word Lengths to Decom Setup, Set Preprocessed Formula to All Words, and Erase List.


Edit PCM Database
Tools
Auto Generate
Set Word Lengths to Decom Setup
Set Preprocessed Formula to All Words
Erase List

Frame v Word >	1	2
0	FR0WD1	FR0WD2
1	(0,1)	(0,2)

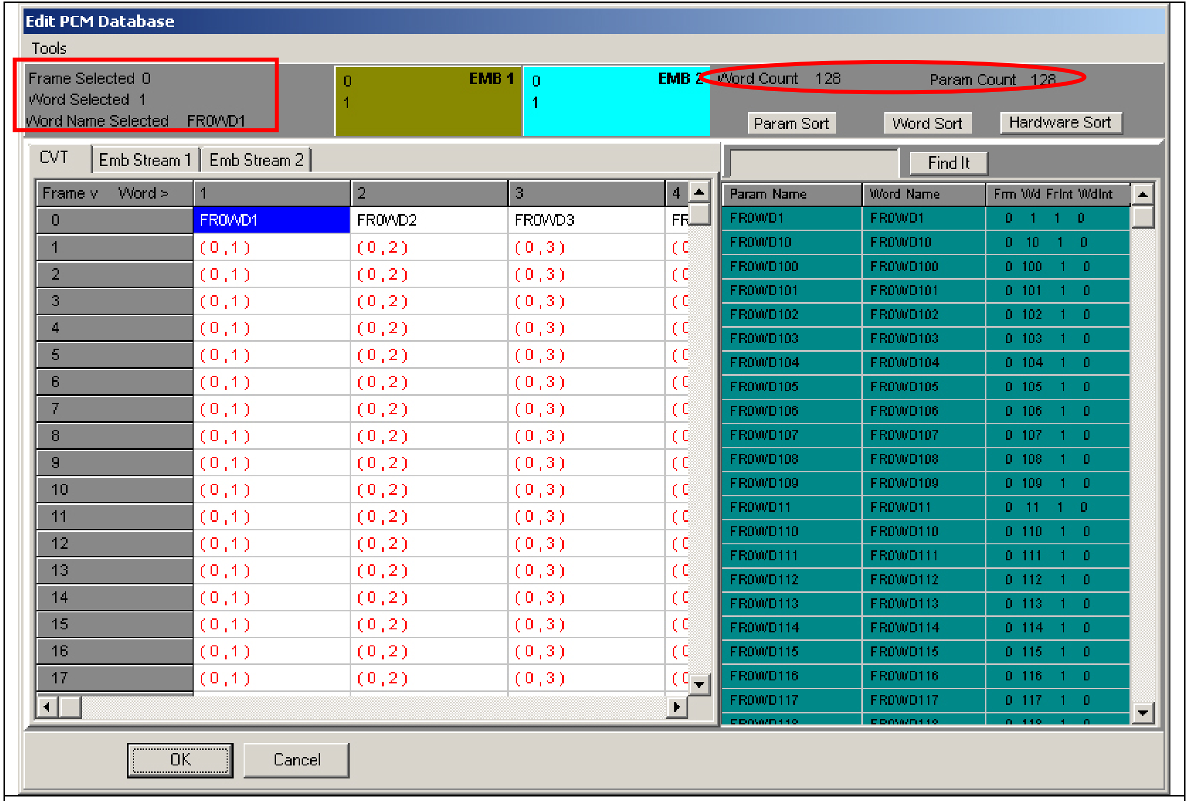
The **Auto Generate** command is a very handy tool for quickly creating a database. Once invoked (click the *Generate It* button shown below left), the user will be prompted to enter a frame interval, after which the program will generate a parameter for each word in a minor frame, and set the frame interval for those words to the frame interval entered by the user. The name of each parameter and each word are the same, with each one starting with FR0WD followed by the word number. NOTE. If a database is generated, and subsequently a word is edited to be super-commutated, it will NOT overwrite the other super-commutated words. Each of the subsequent super-commutated words will have to be individually deleted, which will then be replaced by a red parenthesis indicating where the root super-commutated word is located.



After selecting the **Set Word Lengths to Decom Setup** command, the user will be prompted to enter the name of a decommutator setup file. This will go thru the database and set the word lengths in the database to the corresponding word lengths specified in the decommutator setup file.

The **Set Pre processed Formula to All Words** command allows the user to apply a Pre processed formula to all existing words in the database. This includes a blank Pre processed formula, which will set the formula used to none. Selecting this command will result in the formula setup window shown in Figure 4-11 on page 204.

To start over again, without changing the header info, select the **Erase List** command.



Frame Selected 0
Word Selected 1
Word Name Selected FR0WD1

Word Count 128 Param Count 128

Frame v Word >	1	2	3	4
0	FR0WD1	FR0WD2	FR0WD3	FR
1	(0,1)	(0,2)	(0,3)	(C
2	(0,1)	(0,2)	(0,3)	(C
3	(0,1)	(0,2)	(0,3)	(C
4	(0,1)	(0,2)	(0,3)	(C
5	(0,1)	(0,2)	(0,3)	(C
6	(0,1)	(0,2)	(0,3)	(C
7	(0,1)	(0,2)	(0,3)	(C
8	(0,1)	(0,2)	(0,3)	(C
9	(0,1)	(0,2)	(0,3)	(C
10	(0,1)	(0,2)	(0,3)	(C
11	(0,1)	(0,2)	(0,3)	(C
12	(0,1)	(0,2)	(0,3)	(C
13	(0,1)	(0,2)	(0,3)	(C
14	(0,1)	(0,2)	(0,3)	(C
15	(0,1)	(0,2)	(0,3)	(C
16	(0,1)	(0,2)	(0,3)	(C
17	(0,1)	(0,2)	(0,3)	(C

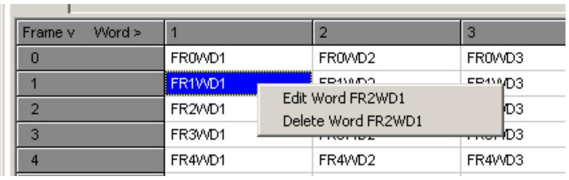
Param Name	Word Name	Frm Wd FrInt WdInt
FR0WD1	FR0WD1	0 1 1 0
FR0WD10	FR0WD10	0 10 1 0
FR0WD100	FR0WD100	0 100 1 0
FR0WD101	FR0WD101	0 101 1 0
FR0WD102	FR0WD102	0 102 1 0
FR0WD103	FR0WD103	0 103 1 0
FR0WD104	FR0WD104	0 104 1 0
FR0WD105	FR0WD105	0 105 1 0
FR0WD106	FR0WD106	0 106 1 0
FR0WD107	FR0WD107	0 107 1 0
FR0WD108	FR0WD108	0 108 1 0
FR0WD109	FR0WD109	0 109 1 0
FR0WD11	FR0WD11	0 11 1 0
FR0WD110	FR0WD110	0 110 1 0
FR0WD111	FR0WD111	0 111 1 0
FR0WD112	FR0WD112	0 112 1 0
FR0WD113	FR0WD113	0 113 1 0
FR0WD114	FR0WD114	0 114 1 0
FR0WD115	FR0WD115	0 115 1 0
FR0WD116	FR0WD116	0 116 1 0
FR0WD117	FR0WD117	0 117 1 0

Figure 4-9 Edit PCM Database Window

Below the main menu and above the frame grid in the figure above is an information area (red oval) displaying the number of parameters (Param Count) and the number of words (Word Count) currently in the database. On the left of this area is the frame number and word number and word name currently being editing (red rectangle).

To the right of the frame grid area is the list of parameters currently in the database. Double clicking on a parameter will invoke the Parameter Definition Window shown in Figure 4-12 on page 206. There is no option to change the Word Information or create/delete parameters with this window. The Parameter Definition window is strictly an edit function of existing parameters.

The frame grid area is where a portion of the editing of the database takes place. To edit a word, select a word via a single click, and once selected, right click to edit or delete the word as shown below. Selecting to edit will result in the Edit PCM Word window shown in Figure 4-10 on page 202.



Frame v Word >	1	2	3
0	FR0WD1	FR0WD2	FR0WD3
1	FR1WD1	FR1WD2	FR1WD3
2	FR2WD1		D3
3	FR3WD1	FR3WD2	FR3WD3
4	FR4WD1	FR4WD2	FR4WD3

Edit Word FR2WD1
Delete Word FR2WD1

By definition, a parameter is a group of bits put together to display data. A parameter is what is used to present data in a display widget, for example. When creating a display, only the parameter name is entered, with the rest of the information being provided by the parameter database. The bits that make up a parameter can all be part of one word, or parts of two words. A word on the other hand is defined as a contiguous set of bits, of a specific length, that make up a PCM frame and is described in the decommutator setup file. Normally all words are of the same length (16 bits, 8 bits, etc), but there can be PCM streams that have words of varying lengths. In either case, as far as the parameter database is concerned, a word is defined as it was in the decommutator setup file, even if the actual word lengths in the PCM stream vary. If the word is sub- or super-commutated, only the first occurrence of the word in the major frame is identified (unless one wishes to display each occurrence as a different parameter). The frame grid portion of Figure 4-9 on page 200, is setup like a spreadsheet. Across the top is the word number for each cell (1 based). Along the left side is the frame number for each cell (0 based). Left click on the corresponding frame/word cell to select the word to edit. This will invoke the Edit PCM Word window shown in Figure 4-10 on page 202.

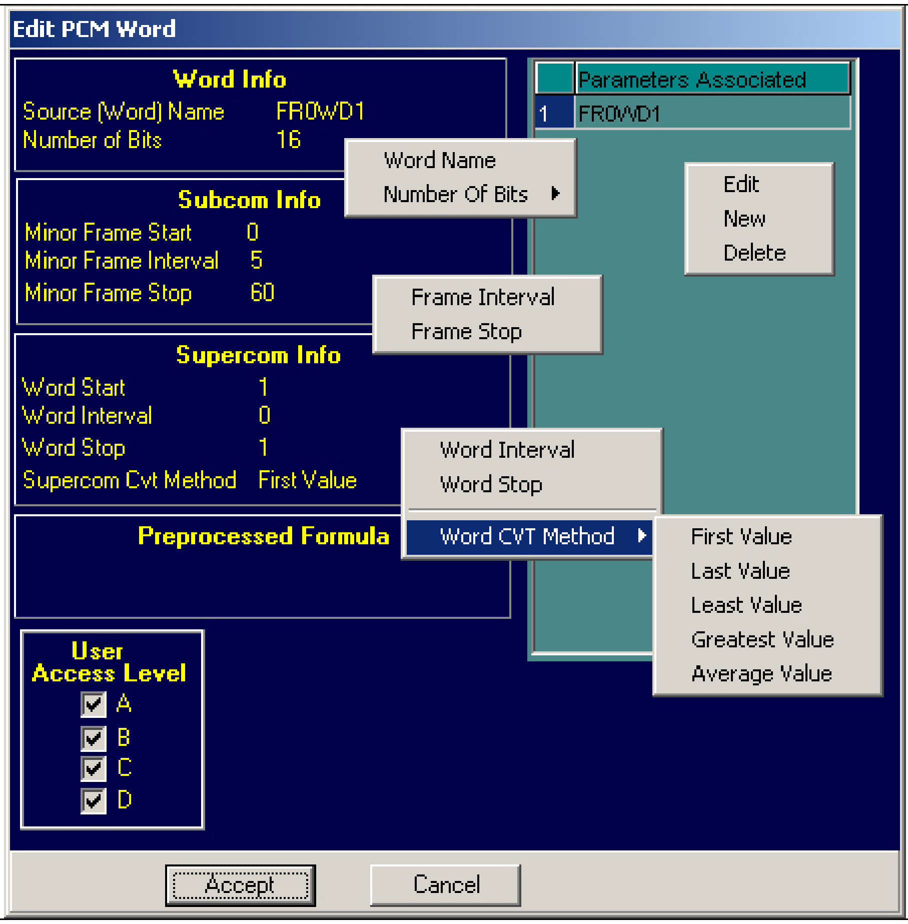
The frame grid will display all the words defined for the database. If the word is super- or sub-commutated, the first appearance of the word will have the word name in the cell. The other cells that the word name belongs to will have a set of red parenthesis that will have the frame number and word number of the originating word in it. For example, a word, starting in frame 0 and word 7 that is sub-commutated every minor frame, will have the actual word name (like FR0WD7 if the database was auto generated) in the cell at frame 0, word 7. All the rest of the cells in the word 7 column will have a "(0,7)" in red, indicating that space in the major frame is used by some other word. This representation makes it easy to see how full the frame is with the data dictionary.

⚠	Since a parameter is made up from bits within a word (or words), the words must be defined first. **ALL WORDS MUST BE CREATRED FIRST.** Failure to do so may result in some of the parameters being set back to defaults, which may or may not be desirable.

Once a word is selected in the grid, right click and a popup menu will allow one to add a word if one doesn't exist, or to edit or delete a word if one has been defined. If a word is deleted, all of the corresponding red parenthesis will disappear as well as the word name. If the **Add New Word** or **Edit Word** command are selected, the Edit PCM Word window described in paragraph 4.3.1.5 will appear to allow the user to edit the word attributes, as well as invoking the parameter definition editor described in paragraph 4.3.1.6 on page 204. If the Add New Word command was selected, the editor will auto name the word, based on the frame and word number. It will also generate a parameter using all the bits in the word. If a word is deleted, all associated parameters with that word are also deleted.

4.3.1.5 Edit PCM Word

The Edit PCM Word window is shown in Figure 4-10 below. There are six sections to configure in the edit word window. The configuration items include: Word Info, Subcom Info, Supercom Info, Preprocessed Formula, Associated Parameters, and User Access Level. Each configuration item is discussed in the numbered paragraphs that follow.



Edit PCM Word

Word Info
Source (Word) Name FR0WD1
Number of Bits 16

Subcom Info
Minor Frame Start 0
Minor Frame Interval 5
Minor Frame Stop 60

Supercom Info
Word Start 1
Word Interval 0
Word Stop 1
Supercom Cvt Method First Value

Preprocessed Formula

User Access Level
☑ A
☑ B
☑ C
☑ D

	Parameters Associated
1	FR0WD1

Figure 4-10 Edit PCM Word Window & Menus

Once the configuration information has been entered, click the **Accept** button. To discard any changes, press the **Cancel** button, which will return the user back to the frame grid (see Figure 4-9 on page 200).

4.3.1.5.1 Word Info

This allows the user to change the name and the number of bits in the selected word from the initial configuration defined in the decommutator setup.

4.3.1.5.2 Subcom Info

This allows the user to change the frame interval, and the stop frame for the selected word. The stop frame value will be entered automatically after the frame interval is entered, based on the number of minor frames set up in the header.

4.3.1.5.3 Supercom Info

This allows the user to change the word interval and the stop word for the selected word. The stop word value will be entered automatically after the word interval is entered, based on the number of words per frame set up in the header. If the selected word is super-commutated, then the user must select the method used to store the value in the CVT.

4.3.1.5.4 Preprocessed Formula

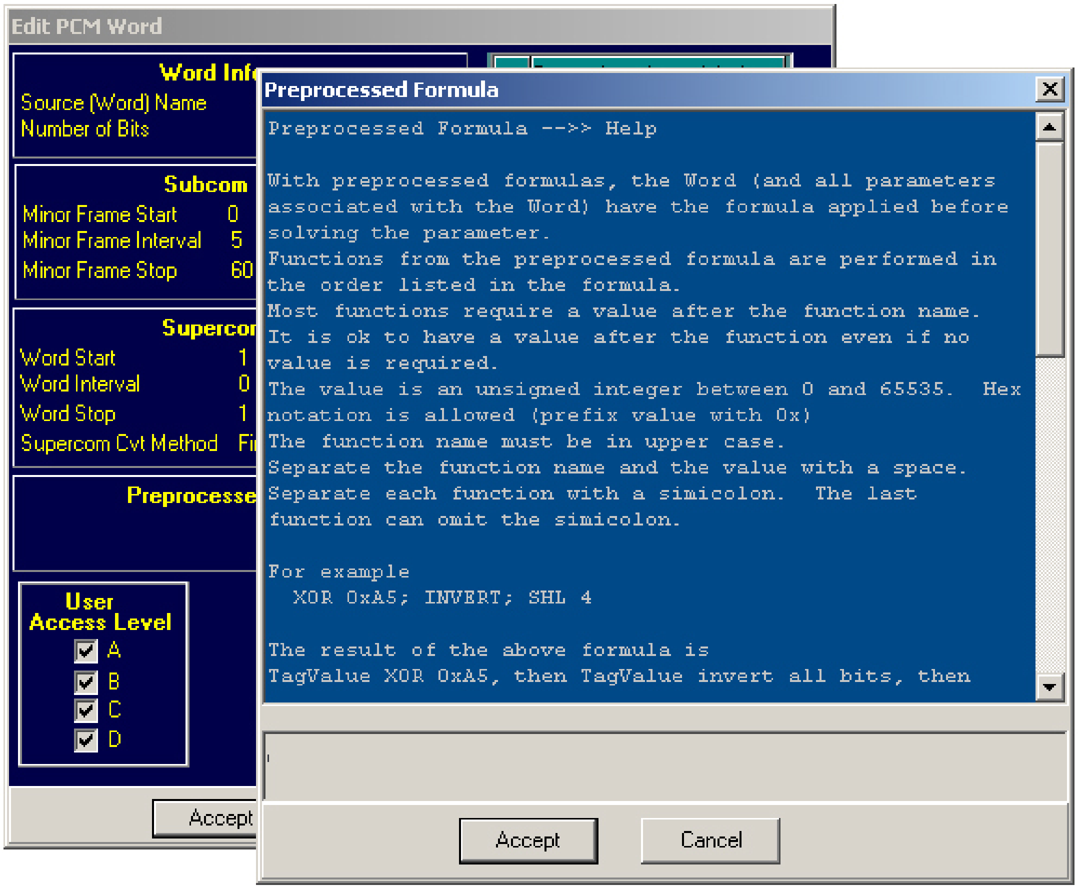
This allows the user to define a preprocessed formula for the selected word, prior to any processing of the parameters. The entered formula will be an integer function and every parameter associated with the word will have the same function(s) applied prior to solving for the parameter. Multiple functions applied to the selected word is supported by separating each function with a semicolon. To invoke the preprocessed formula, right click in the area and select “Preprocessed Formula.” The resulting window is shown in Figure 4-11 on page 204. Enter the formula in the lower portion of the window, guided by the help file info in the upper portion of the window.

4.3.1.5.5 Associated Parameters

On the right of the word editor shown in Figure 4-10 on page 202, is a list of defined parameters that are associated with the selected word. The minimum number of parameters that may be associated with a word is 1. The maximum is 256. Right click in this area to Add, Edit, or Delete a parameter. To Edit or Delete a parameter, first select it by left clicking the parameter. Editing or adding a new parameter will invoke the parameter definition window shown in Figure 4-12 on page 206.

4.3.1.5.6 User Access Level

This allows the user to define the user access level. This function is for system administrators that want to control the distribution of the database to different user groups, where certain groups cannot access the data. It is recommended that all four boxes be checked, unless great care is taken.



Source (Word) Name
Number of Bits

Minor Frame Start 0
Minor Frame Interval 5
Minor Frame Stop 60

Word Start 1
Word Interval 0
Word Stop 1
Supercom Cvt Method Fi

☑ A
☑ B
☑ C
☑ D

```
Preprocessed Formula -->> Help

With preprocessed formulas, the Word (and all parameters
associated with the Word) have the formula applied before
solving the parameter.
Functions from the preprocessed formula are performed in
the order listed in the formula.
Most functions require a value after the function name.
It is ok to have a value after the function even if no
value is required.
The value is an unsigned integer between 0 and 65535.  Hex
notation is allowed (prefix value with 0x)
The function name must be in upper case.
Separate the function name and the value with a space.
Separate each function with a simicolon.  The last
function can omit the simicolon.

For example
  XOR 0xA5; INVERT; SHL 4

The result of the above formula is
TagValue XOR 0xA5, then TagValue invert all bits, then
```

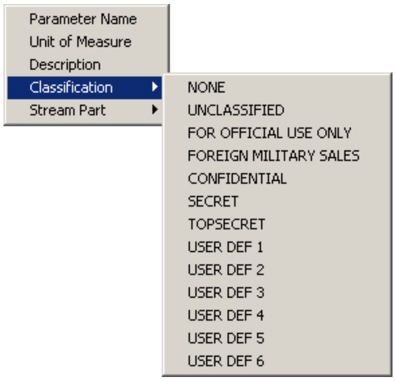
Figure 4-11 Edit PCM Word Window - Preprocessed Formula

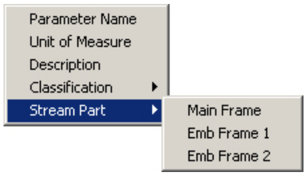
4.3.1.6 Parameter Definition

The parameter definition window is shown in Figure 4-12 on page 206, and is common for all database types (Chapter 4, Chapter 8, and 1553). There are five sections to configure in the parameter definition window. The configuration items include: Parameter ID, Process Definition, Advanced Process, Super Advanced Info, and Default Display Settings. Each configuration item is discussed in the numbered paragraphs that follow.

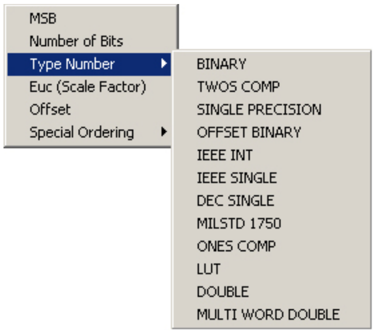
4.3.1.6.1 Parameter ID

This allows the user to define parameter identification and classification level information (text information). The information entered here is optional (but useful). Shown below left, the **Classification Level** is useful for the client program to display the highest classification of all parameters processed for the display page. **The Stream Frame Part** field identifies what stream (primary or embedded) the selected parameter comes from. Other parameter identification fields include **Parameter Name**, **Units of Measure**, and **Description**. The only part LDPS does not use is STREAM PART. A soft decom would, but LDPS does not. The rest of the fields in this block are used by LDPS.



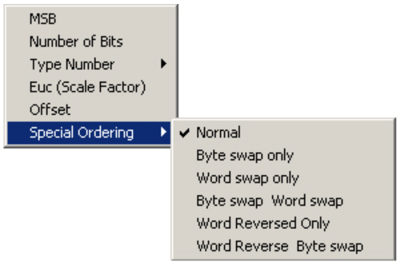


4.3.1.6.2 Process Definition

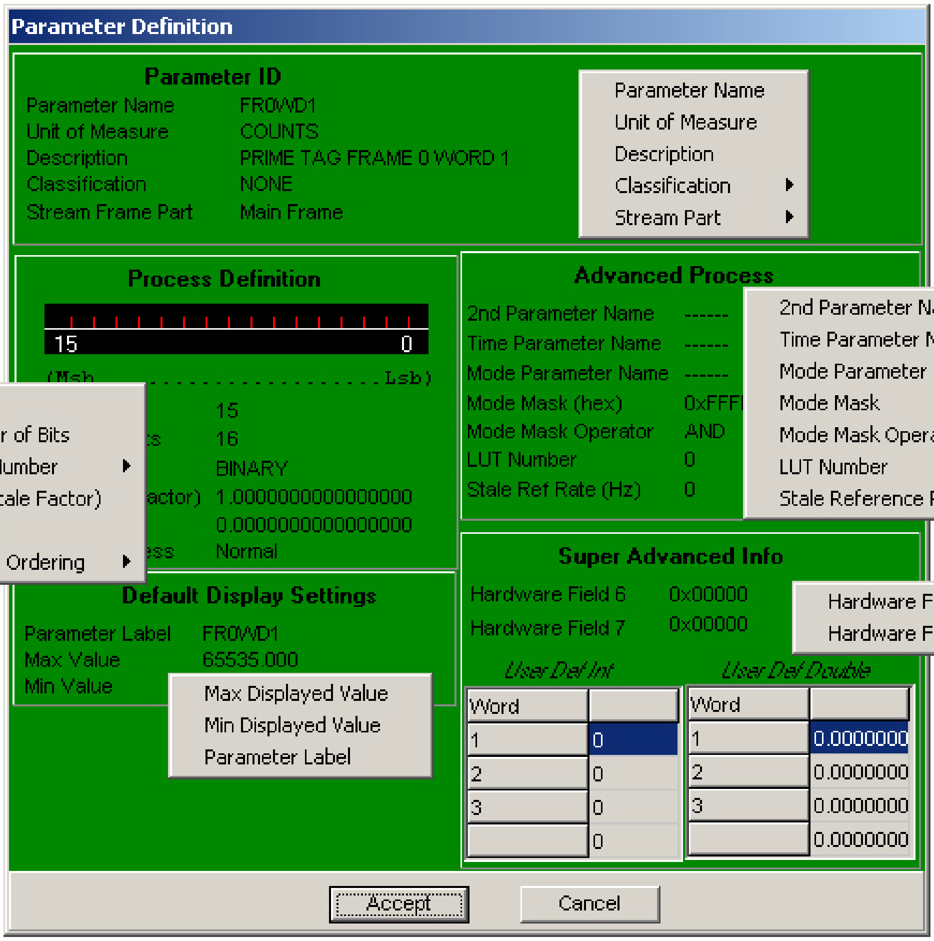


This allows the user to define how the parameter is extracted from the selected word, what kind of data the parameter represents, and how to convert the parameter to its real meaning. Shown right, the parameter definition menu has six items including: MSB, Number of Bits, Type Number, EUC Scale Factor, Offset, and Special Ordering. Recall that all parameters go through a linearization process function called "Mx + b." To define the "x" value, select **EUC (Scale Factor)** and enter the value. Then, define the "+ b" value by selecting **Offset**, and enter the value. For most non-analog streams, the offset value is set to 0.0.

At the top of the process definition box in Figure 4-12 is a little window showing where the parameter bits are within the selected word. Note: if the parameter requires more bits than are defined in the source word, then a second parameter name must be assigned (see paragraph 4.3.1.6.3), unless the bits are contiguous in the stream. If the **Number of Bits** in the parameter is set to be more than there are in the word, then the user will be prompted to answer (Yes/No) if the bits are contiguous within the stream. If they are not contiguous, the number of bits will be set to match the word length. If they are contiguous, then the **MSB** value is set to match the word length minus one (–1), the LSB is set to 0, and the number of bits is set as entered by the user. The label next to the "Number of Bits" will indicate the number followed by "(contig bits)" if the bits are contiguous.



Finally, to address big Endean/little Endean issues, there is the **Special Ordering** command (shown left). If the parameter is 16, 32, or 64 bits, and it is in Big Endean form, the user can have the raw data byte swapped followed by a word swapped to convert the Big Endean form to Little Endean form for PCs. Note: Big Endean swapping will not occur unless the number of parameters is exactly 16, 32, or 64.



Parameter ID

Parameter Name FROWD1
Unit of Measure COUNTS
Description PRIME TAG FRAME 0 WORD 1
Classification NONE
Stream Frame Part Main Frame

Process Definition

15
s 16
BINARY
actor) 1.0000000000000000
0.0000000000000000
ess Normal

Default Display Settings

Parameter Label FROWD1
Max Value 65535.000
Min Value

Advanced Process

2nd Parameter Name ------
Time Parameter Name ------
Mode Parameter Name ------
Mode Mask (hex) 0xFFFF
Mode Mask Operator AND
LUT Number 0
Stale Ref Rate (Hz) 0

Super Advanced Info

Hardware Field 6 0x00000
Hardware Field 7 0x00000

User Def Int

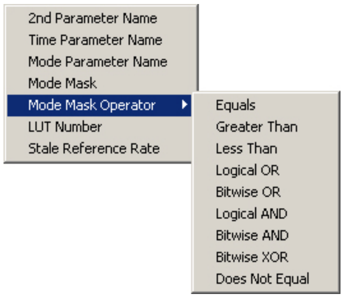
Word	
1	0
2	0
3	0
	0

User Def Double

Word	
1	0.0000000
2	0.0000000
3	0.0000000
	0.0000000

Figure 4-12 Parameter Definition Window & Menus

4.3.1.6.3 Advanced Process



Some databases parameters are more complex and require additional information. This information is configured in the advanced process section. Shown right, the advanced process menu has seven items including: 2nd Parameter Name, Time Parameter Name, Mode Parameter Name, Mode Mask, Mode mask Operator, LUT Number, and Stale Reference Rate

If the selected parameter requires another parameter to be concatenated with, prior to applying the "Mx + b" linearization, then enter the second parameter name by invoking the **2nd Parameter Name** command. The **Time Parameter Name** command is used to enter the name of the time stamp parameter associated with the selected parameter.

If the selected parameter can only be updated if another parameter (operator[21]) a certain value, then enter the other parameter name by invoking the **Mode Parameter Name** command. The **Mode Mask** command is used in conjunction with the Mode Parameter Name. This allows the user to assign the value that the Mode Parameter Name is supposed to 'mask with operator' with before the selected parameter is updated. The value entered must be entered in hex. The **Mode Mask Operator** command allows the user to choice the operator involved in checking the mode mask value. The options include:

- "Equals", //value == mask
- "Greater Than", //value > mask
- "Less Than", //value < mask
- "Logical OR", //value || mask > 0
- "Bitwise OR", //value | mask > mask
- "Logical AND", //value && mask > 0
- "Bitwise AND", //value & mask >= mask
- "Bitwise XOR" //value ^ mask > 0 .
- "Does Not Equal", //value != mask

An example of Mode Parameter Usage follows. Assume a parameter named "LAUNCH." The Launch parameter is not supposed to be updated unless the parameter named "MASTERARM" is equal to a value of 7. In this scenario, the user will set the Mode Parameter Name to MASTERARM, the Mode Mask to 0x7, and the Mode Mask Operator to Equals. Then during processing, if the MASTERARM parameter does not equal 7, then the LAUNCH parameter will retain its last value, and will not be updated.

If the selected parameter requires a look-up table (LUT) to be solved, then enter the look-up table number to use by invoking the **LUT Number** command. The Advanced Steam Information section (see paragraph 4.3.1.2 on page 192) identifies the look-up table file name to use. The Process Definition Type Number (see paragraph 4.3.1.6.2 on page 205) identified the parameter as requiring a look-up table to solve.

If the reference rate for the selected parameter is not the same as the Common Stale Reference Rate (see paragraph 4.3.1.2 on page 192) defined in the Header for most parameters, then invoke the **Stale Reference Rate** command to change the rate for the selected parameter.

4.3.1.6.4 Super Advanced Info

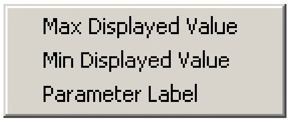
If a software decommutator is not in use, the Super Advanced Information section will not be available. There are eight (8) hardware fields used in the parameter database (numbered 0 through 7). Depending on the database type, not all are used. For the PCM type (Chapter 4), fields 6 and 7 are not used. This allows the user to place any integer value in these

[21] Operator- Equals, is Greater Than, is Less Than, etc.

fields (max value is 0xFFFFF). Normally, these two fields are blank. However, one may need to pass special information to a software decommutator via these fields. In addition to the spare hardware fields (fields 6 and 7), there are four user-defined integers and four user defined double precision variables that may be entered. Again, the LDPS application does not use this information, but a software decommutator may.

If the software decommutator requires the user to enter data, it should also provide a help file that will explain what parameters are to be entered. The help file window is displayed directly below the tabs as shown in Figure 4-3 on page 191. This help file has the extension ".phlp," and is a simple text file with the same name as the DLL, and located in the same place. There are ten fields available for use in the super advanced section. The first two are for hardware fields 6 and 7, which for Chapter 4 PCM streams aren't used in the parameter database. The next four are integer type fields, with the final four being double type fields. The help file can have the labels assigned for all ten of these fields instead of having generic labels. The first ten lines of the help file are scanned for brackets "[]," and the text within those brackets are used for the labels.

4.3.1.6.5 Default Display Settings

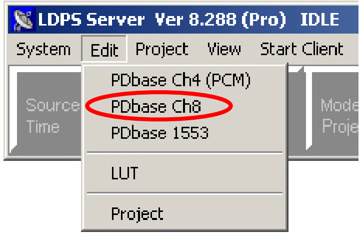


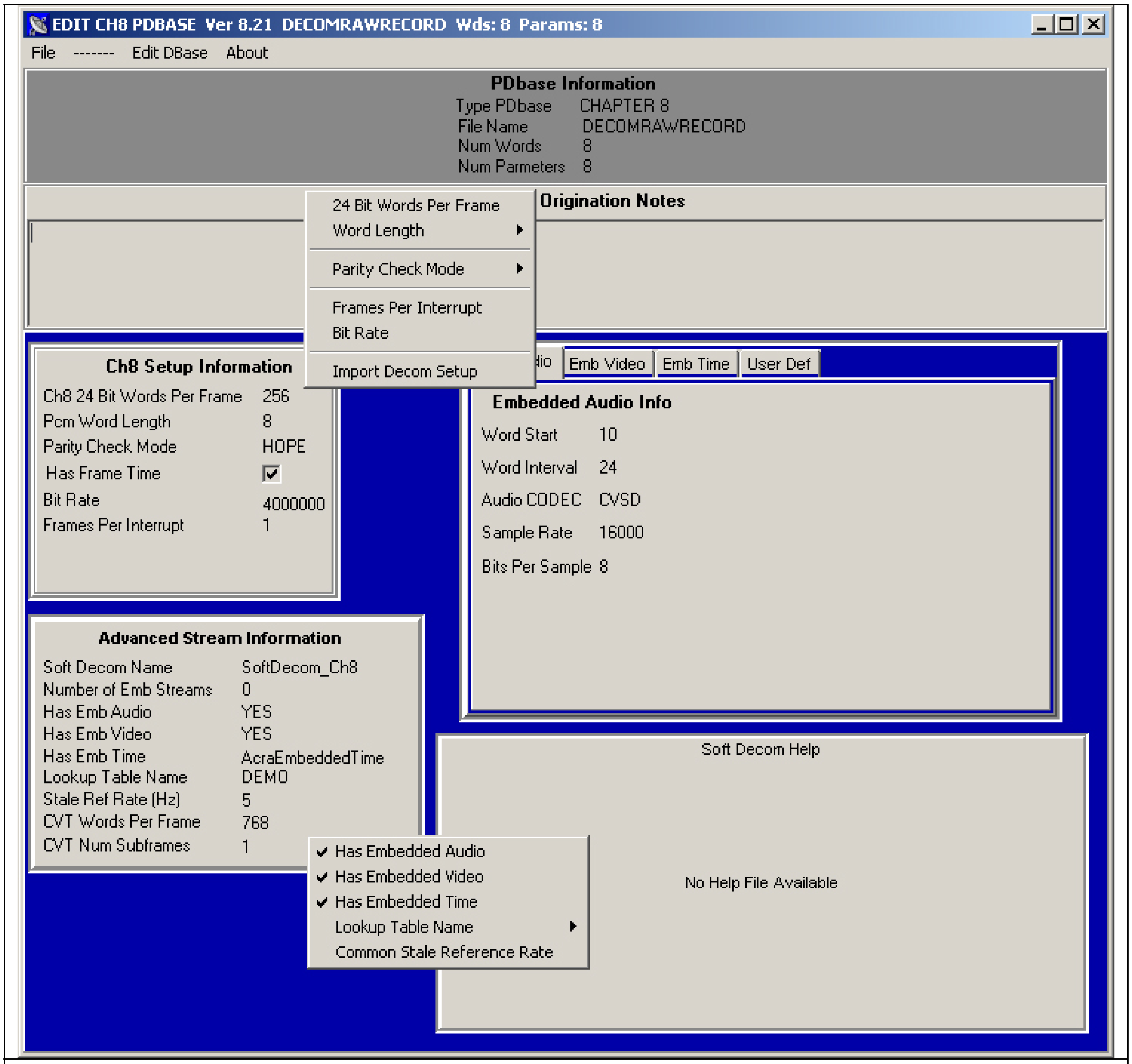
Shown right, the default display settings menu has three items: Max Displayed Value, Min Displayed Value, and Parameter Label. On the client side, the label that goes on a widget is automatically set to the same name as the parameter name (if the user selects to automatically get values from the database – see Part-2 of this manual). By invoking the **Parameter Label** command, an alternate name for the selected parameter may be displayed instead. For example, assume a parameter name called "INPLAT" is selected, but the word "Latitude" needs to be displayed on the widget. The user would enter the alternate name via this command. If the parameter label field is left blank, then the parameter name will be used.

Some display widgets feature maximum and minimum display limits (like a needle gauge). Use the **Max Displayed Value** and **Min Displayed Value** commands to enter the min and max values for the selected parameter (again assuming the inputs to the widgets automatically come from the database).

4.3.2 IRIG-106 Chapter 8 Database Type

Selecting the **PDbase Ch8** command from the servers' edit menu will invoke the **EDIT CH8 PDBASE** window shown in Figure 4-13 below. The major configuration items in this window include: Chapter 8 Setup Information, Advanced Stream Information, and Setup Tabs for those items enabled in the advanced stream information section. The Chapter 8 database tab edit controls are much the same as the PCM database tab edit controls described in paragraph 4.3.1.3 on page 193.





PDbase Information

Type PDbase	CHAPTER 8
File Name	DECOMRAWRECORD
Num Words	8
Num Parmeters	8

Ch8 Setup Information

Ch8 24 Bit Words Per Frame	256
Pcm Word Length	8
Parity Check Mode	HOPE
Has Frame Time	☑
Bit Rate	4000000
Frames Per Interrupt	1

Embedded Audio Info

Word Start	10
Word Interval	24
Audio CODEC	CVSD
Sample Rate	16000
Bits Per Sample	8

Advanced Stream Information

Soft Decom Name	SoftDecom_Ch8
Number of Emb Streams	0
Has Emb Audio	YES
Has Emb Video	YES
Has Emb Time	AcraEmbeddedTime
Lookup Table Name	DEMO
Stale Ref Rate (Hz)	5
CVT Words Per Frame	768
CVT Num Subframes	1

Soft Decom Help

No Help File Available

Figure 4-13 Edit Parameter Database Window & Menus (Chapter 8 Type)

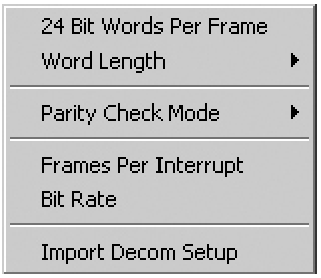
At the top of the window in Figure 4-13 above are two menu items: File and Edit Dbase. The **File** command is used for saving and recalling databases. The database files are stored in two formats, ASCII and binary. When the server and client(s) load a database(s), it is the binary version that is used. To modify the database, use Microsoft Excel or another ASCII text editor to edit the ASCII version of the file. Then **Recall** the ASCII version of the file in the LDPS database editor and **Save** it. This will compile the database and save both the ASCII and the binary versions. The compiled binary version of the file is used when the file recall command is invoked. When the **Import ASCII Pdb File** command is invoked, the ASCII version is recalled.

For those wishing to edit the parameter database in Excel or other ASCII editor, the user may also import just the parameter tags (**Import Tags Only**). The header information is not required for the import. Just have the first line in the ASCII file contain "*.*" and then the rest of the lines are as normal for the ASCII version. Then, use the LDPS parameter editor to set up the header information and save the file. This will yield a compiled version (both binary and ASCII). The user can also export a database to some other format, provided the export DLL assigned in the programs options.

Clicking on the **Edit Dbase** command in Figure 4-13 will result in the Edit Ch8 Database Window shown in Figure 4-14 on page212. This window allows editing of the words and parameters in the parameter database.

Below the menu section in Figure 4-13 is information regarding the database that includes: The name of the database loaded for editing (File Name), the count of root words in the database (Num Words), the count of parameters in the database (Num Parameters), and database type (Type Dbase). Included types: PCM, 1553, and IRIG Chapter 8.

4.3.2.1 Chapter 8 Setup Information



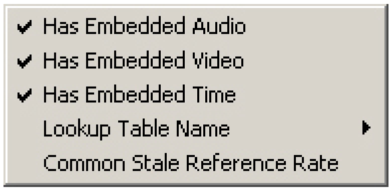
Right click in this area to invoke the menu shown right. As defined in the IRIG-106 document, Chapter 8 words are 24-bits in length. However, the Lumistar LS-50-P decommutator cannot handle word lengths greater than 16 bits. Because of this, the user must configure the word length to be either 8 or 12 bits via the **Word Length** Command. This results in the corresponding words-per-frame being multiplied by 3 or 2. For archiving purposes, the 12-bit word length uses less disk space. The IRIG-106 specification also states the number of Chapter 8 words to number between 128 and 256. Selecting the **24 Bit Words Per Frame** command allows the user to enter the number of Chapter 8, 24-bit words. The number of PCM/CVT words will be updated according to this number.

The **Parity Check** mode defaults to the "Hope for the best" mode. As there is no standardized response to a parity error, the software decommutator implements four modes for parity checking: 1) None – Do nothing, don't even check for parity error. 2) Hope – Check for parity error and count them, but do nothing about them. 3) Word – Don't process the word. The word will not be stored in the buffer. 4) Reset Decoder – The decoder for the specific bus will be reset.

The user may either set up the individual parameters required or select **Import from Decom Setup File**. If the latter option is selected, the user will be prompted for the name of the setup

4.3.2.2 Advanced Information

Right click in this section to invoke the menu shown right. Depending on what is entered here, some advanced tabs will appear where further information will be required.



At the top the box, the file name of the software decommutator being used in displayed. The user may not edit this field. If the soft decom has a help file, it will be displayed next to this box. The number of embedded streams is also displayed, and cannot be edited. The current Chapter 8 software decommutator does not support embedded streams.

If the stream has embedded audio, then check the **Has Embedded Audio** command (see paragraph 4.3.2.5 on page 215). This will cause a new tab to appear with additional setup information required. Likewise, if the stream has embedded video or time, then click the **Has Embedded Video** and/or **Has Embedded Time** commands respectively (see paragraph 4.3.2.6 on page 216 and paragraph 4.3.2.7 on page 217). As before, this will cause new tabs to appear with additional setup information required. For streams with embedded time, the user will be required to enter the name of the time DLL that will support the stream. If the database requires a look-up table, then the user can select from a list of available look-up tables previously generated by clicking the **Lookup Table Name** command.

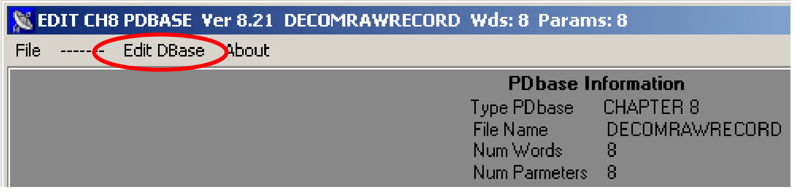
The **Stale Reference Rate** command sets the rate that most parameters will update at. This command sets the default for all parameters. The state rate is used by display programs to issue alerts when data for parameters are not updating at the required rate (based on the time stamp for that parameter).

The **CVT Words Per Frame** and **CVT Num Subframes** values are displayed and cannot be changed here. The CVT words-per-frame is based on the Chapter 8, 24-Bit Words Per Frame and the Parameter Word Length settings in the Chapter 8 Setup Information section (see paragraph 4.3.2.1). The CVT Number of Subframes is fixed at 1 (per the IRIG-106 Chapter 8 specification).

If the database requires a look-up table, then the user can select from a list of available look-up tables previously generated by clicking the **Lookup Table Name** command.

4.3.2.3 Edit Chapter 8 Database Window

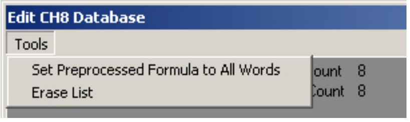
To invoke the Edit Chapter 8 Database window shown in Figure 4-14 below, click on the **Edit DBase** command in the Edit Parameter Database window shown in Figure 4-13 on page 209 (a cutaway of which is shown below).

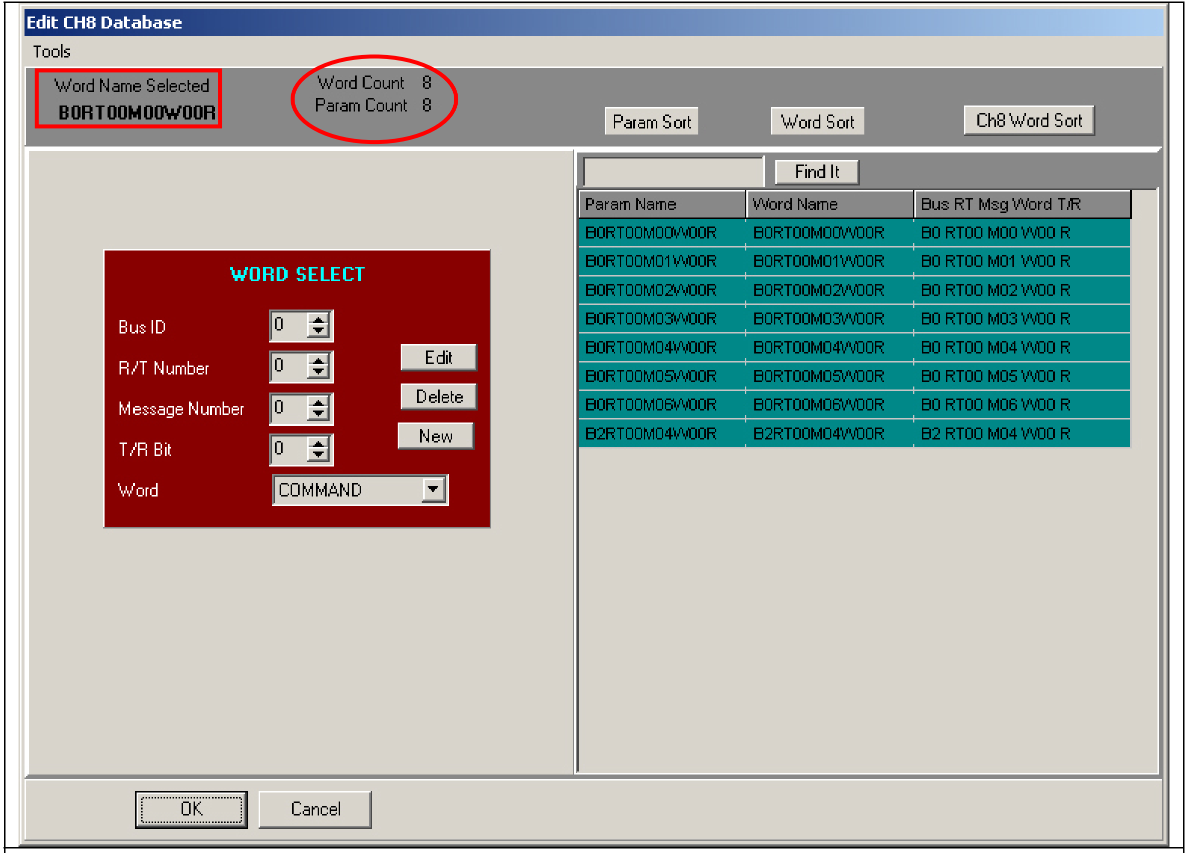


PDbase Information

Type PDbase	CHAPTER 8
File Name	DECOMRAWRECORD
Num Words	8
Num Parmeters	8

Across the top of the Edit Chapter 8 Database window shown right is the main menu, which currently contains the **Tools** command. The database tools include: Set Preprocessed Formula to All Words, and Erase List.





Word Name Selected
B0RT00M00W00R

Word Count 8
Param Count 8

Bus ID	0
R/T Number	0
Message Number	0
T/R Bit	0
Word	COMMAND

Param Name	Word Name	Bus RT Msg Word T/R
B0RT00M00W00R	B0RT00M00W00R	B0 RT00 M00 W00 R
B0RT00M01W00R	B0RT00M01W00R	B0 RT00 M01 W00 R
B0RT00M02W00R	B0RT00M02W00R	B0 RT00 M02 W00 R
B0RT00M03W00R	B0RT00M03W00R	B0 RT00 M03 W00 R
B0RT00M04W00R	B0RT00M04W00R	B0 RT00 M04 W00 R
B0RT00M05W00R	B0RT00M05W00R	B0 RT00 M05 W00 R
B0RT00M06W00R	B0RT00M06W00R	B0 RT00 M06 W00 R
B2RT00M04W00R	B2RT00M04W00R	B2 RT00 M04 W00 R

Figure 4-14 Edit Database Window (Chapter 8 Type)

The **Set Pre processed Formula to All Words** command allows the user to apply a preprocessed formula to all existing words in the database. This includes a blank preprocessed formula, which will set the formula used to none. Selecting this command will result in the formula setup window shown in Figure 4-11 on page 204.

To start over again, without changing the header info, select the **Erase List** command.

⚠	**WARNING**. If there are parameters defined in the database, and adjustments to the header information was subsequently made such that the CVT size is less than when the parameters were defined, then the parameters will be DELETED. The user must enter the header information correctly before editing any parameters.

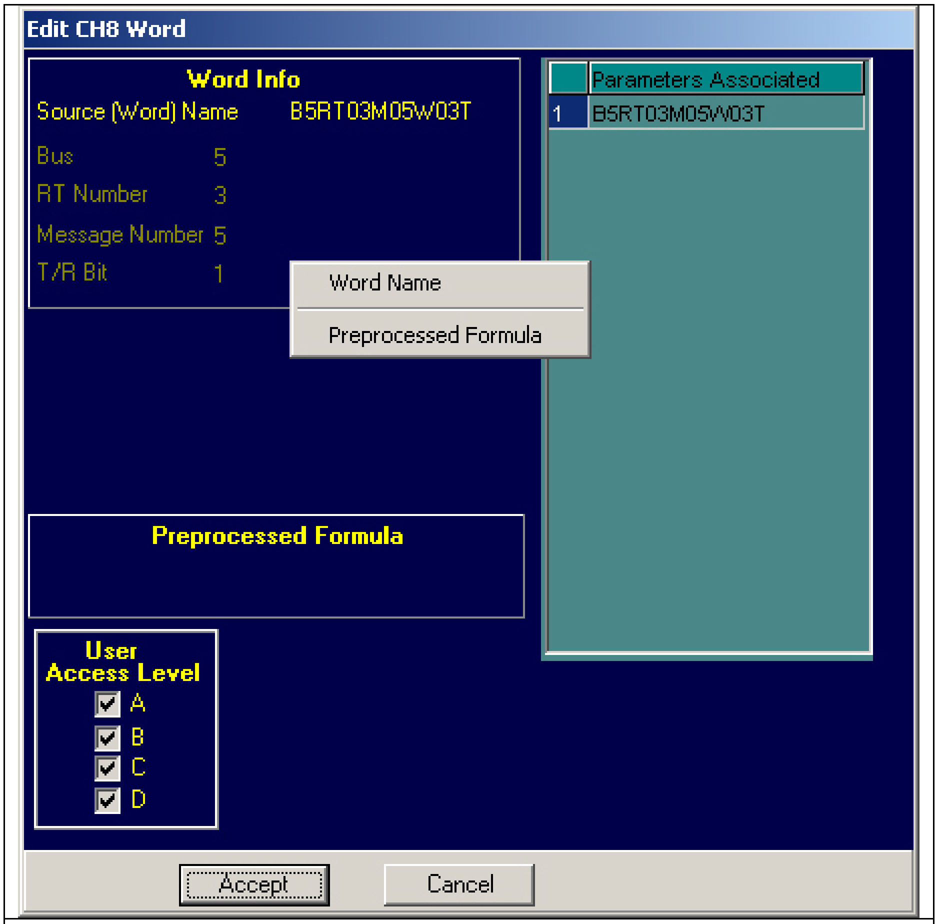
Below the main menu in the figure above is an information area (red oval) displaying the number of parameters (Param Count) and the number of words (Word Count) currently in the database. On the left of this area is the word name currently being editing (red rectangle).

To the right of the word select area is the list of parameters currently in the database. Double clicking on a parameter will invoke the Parameter Definition Window shown in Figure 4-12 on page 206. There is no option to change the Word Information or create/delete parameters with this window. The Parameter Definition window is strictly an edit function of existing parameters. The parameter definition function is common among all database types (Chapter 4, Chapter 8, and 1553)

The Word Select Area allows the user to add, edit, or delete word information. Here the user selects the bus number, RT number, message number, T/R bit position, and Ch8 word number for the selected database word. If a database word is added or edited, the user will be taken to the word editor described in paragraph 4.3.2.4 on the following page, where the user may add a preprocessed formula, set user access levels, and add additional parameters to that root word.

4.3.2.4 Chapter 8 Edit Word

The Chapter 8 word editor is shown in Figure 4-15 below. There are four sections to configure in the Chapter 8 edit word window. The configuration items include: Word Info, Preprocessed Formula, Associated Parameters, and User Access Level. Each configuration item is discussed in the paragraphs that follow.



Edit CH8 Word

Word Info

Source (Word) Name B5RT03M05W03T

Bus 5

RT Number 3

Message Number 5

T/R Bit 1

	Parameters Associated
1	B5RT03M05W03T

Preprocessed Formula

User Access Level

☑ A
☑ B
☑ C
☑ D

Figure 4-15 Edit Chapter 8 Word Window & Menus

The word info section displays the name of the source word, the bus, RT and message numbers, and the R/T Bit position.

The User Access Level allows the user to define various levels of access to the database. This function is for system administrators that want to control the distribution of the database to different user groups, where certain groups cannot access the data. It is recommended that all four boxes be checked, unless great care is taken.

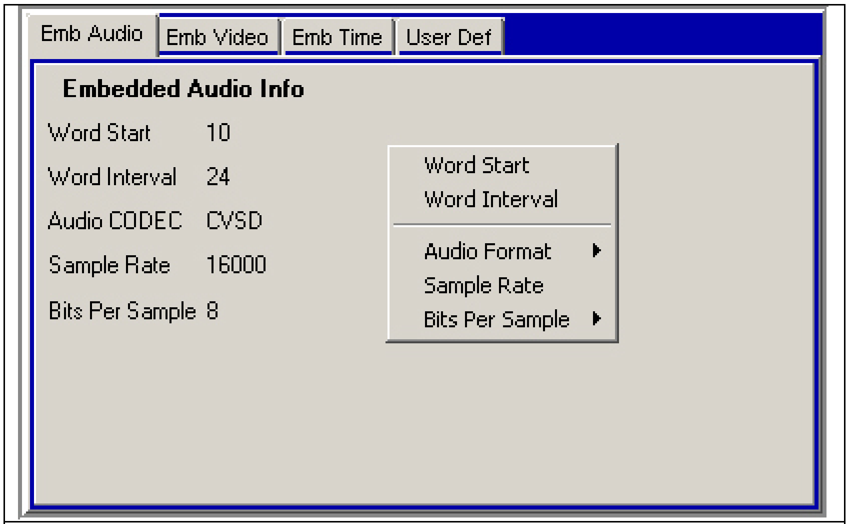
The Preprocessed Formula command allows the user to define a preprocessed formula for the selected word, prior to any processing of the parameters. The formula entered will be an integer function and every parameter associated with the word will have the same function(s) applied prior to solving for the parameter. Multiple functions may be applied to the selected word by separating each function with a semicolon. To invoke the preprocessed formula, right click in the area and select “Preprocessed Formula.”

On the right of the word editor shown in Figure 4-15 on page 214, is a list of defined parameters that are associated with the selected word. The minimum number of parameters that may be associated with a word is 1. The maximum is 256. Right click in this area to Add, Edit, or Delete a parameter. To Edit or Delete a parameter, first select it by left clicking on the parameter. Editing or adding a new parameter will invoke the parameter definition window shown in Figure 4-12 on page 206. The parameter definition function is common among all database types (Chapter 4, Chapter 8, and 1553)

Once the configuration information has been entered, click the **Accept** button. To discard any changes, press the **Cancel** button, which will return the user back to the edit Chapter 8 database window (see Figure 4-14 on page 212).

4.3.2.5 Chapter 8 Embedded Audio Tab

The Embedded Audio tab allows the user to set up any embedded audio streams. To enable the embedded audio functionality in the parameter database, select the **Has Embedded Audio** command from the **Advanced Stream Information** menu described in paragraph 4.3.2.2 on page 211. The **Word Start** indicates the starting word in the primary frame where the embedded audio begins (1 based index).


Embedded Audio Info

Word Start 10

Word Interval 24

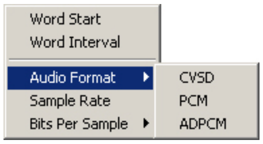
Audio CODEC CVSD

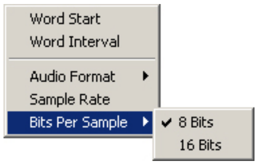
Sample Rate 16000

Bits Per Sample 8

Figure 4-16 Embedded Audio Tab (Chapter 8)

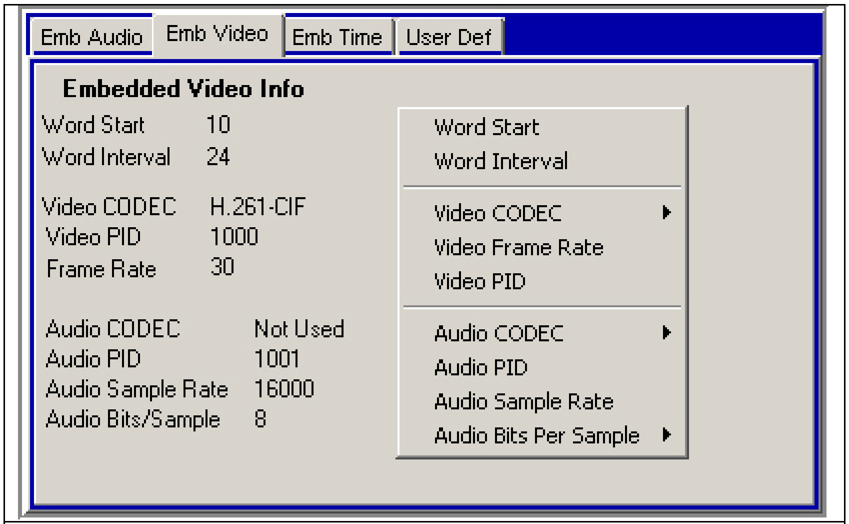
The **Word Interval** indicates how the embedded audio is commutated within the primary frame. The **Audio Format** command selects the encoding format the embedded audio was encoded with (the default is CVSD). Other CODECs include PCM and ADPCM. The **Sample Rate** selects the sample/second that the audio was encoded at (the default is 16 Khz). Eight or sixteen bits/sample may be selected by the user via the **Bits Per Sample** Command.





4.3.2.6 Chapter 8 Embedded Video Tab

The Embedded Video tab allows the user to set up any embedded video streams. To enable the embedded video functionality in the parameter database, select the **Has Embedded Video** command from the **Advanced Stream Information** menu described in paragraph 4.3.2.2 on page 211. The **Word Start** indicates the starting word in the primary frame where the embedded video begins (1 based index). The **Word Interval** indicates how the embedded video is commutated within the primary frame.

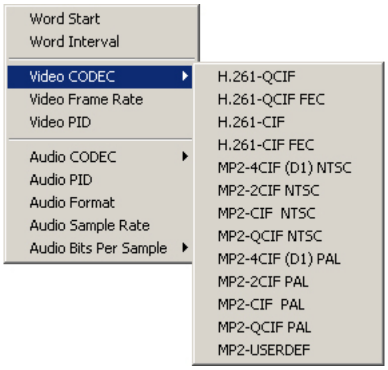


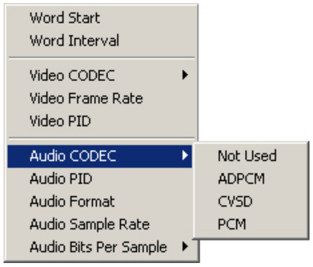
Embedded Video Info

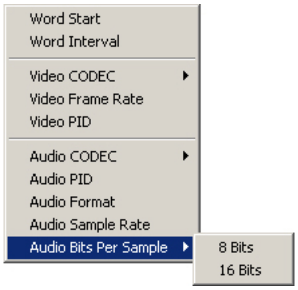
Word Start	10
Word Interval	24
Video CODEC	H.261-CIF
Video PID	1000
Frame Rate	30
Audio CODEC	Not Used
Audio PID	1001
Audio Sample Rate	16000
Audio Bits/Sample	8

Figure 4-17 Embedded Video Tab (Chapter 8)

The **Video CODEC** command selects the encoding format the embedded video was encoded with (the default is H.261 - CIF). Some of the other supported formats include: H.264-QCIF, H.264-CIF FEC, MP2-4CIF. The complete list of supported CODECs is shown below left.



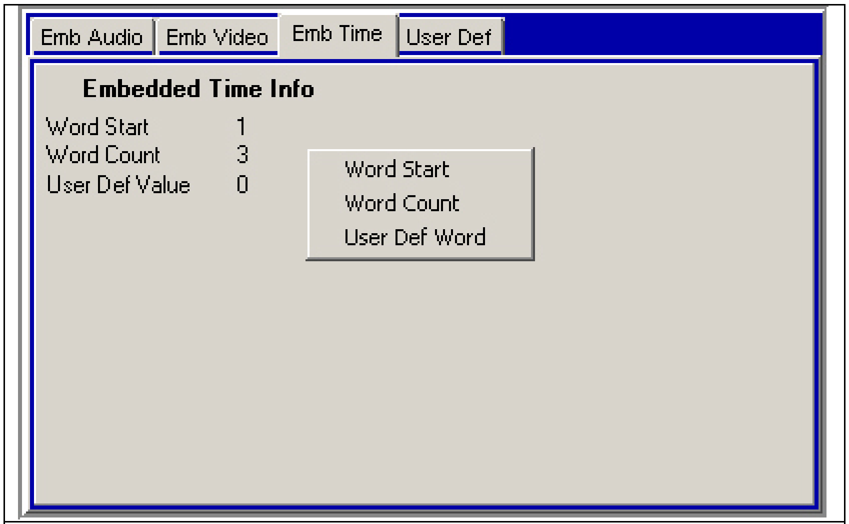




The **Video Frame Rate** selects the frames/second that the video was encoded at (the default is 30 fps). The **Audio CODEC** command selects the encoding format the audio portion of the embedded video was encoded with (the default is CVSD). Other CODECs include PCM and ADPCM. The **Audio Sample Rate** selects the sample/second that the audio was encoded at (the default is 16 KHz). Eight or sixteen bits/sample may be selected by the user via the **Audio Bits Per Sample** command.

4.3.2.7 Chapter 8 Embedded Time Tab

For Chapter 8 streams that have embedded time, the user can either use the parameter database to collect the time and decode it or use a display widget function. To enable the parameter database to decode the embedded time data, the user must select the appropriate DLL to decode the time. To enable the embedded time functionality in the parameter database, select the **Has Embedded Time** command from the **Advanced Stream Information** menu described in paragraph 4.3.2.2 on page 211. After selecting the file name of the embedded time DLL, a new tab (shown in Figure 4-18 below) will appear.



Embedded Time Info

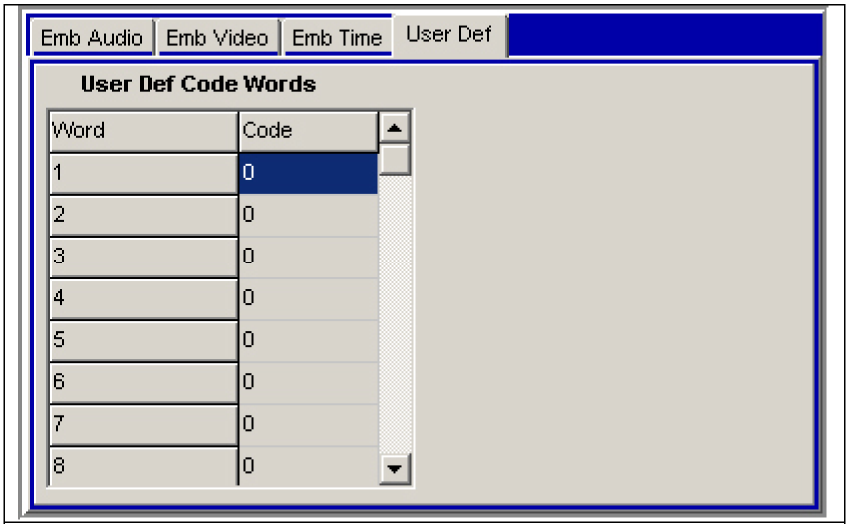
Word Start 1
Word Count 3
User Def Value 0

Figure 4-18 Embedded Time Tab (Chapter 8)

The **Word Start** indicates the starting word in the primary frame where the embedded time data begins (1 based index). The **Word Count** indicates how the embedded time is commutated within the primary frame.

4.3.2.8 Chapter 8 User Defined Tab

The user-defined code words are integers the user may define as desired. The user-defined code words are not used by LDPS, but instead are intended for use with a software decommutator. For instance, use some of these integers as flags to affect how the software decommutator will treat parameters. For the Chapter 8 software decommutator, there is no help file, and these words are used for debug purposes.

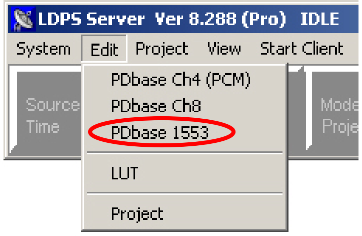


User Def Code Words

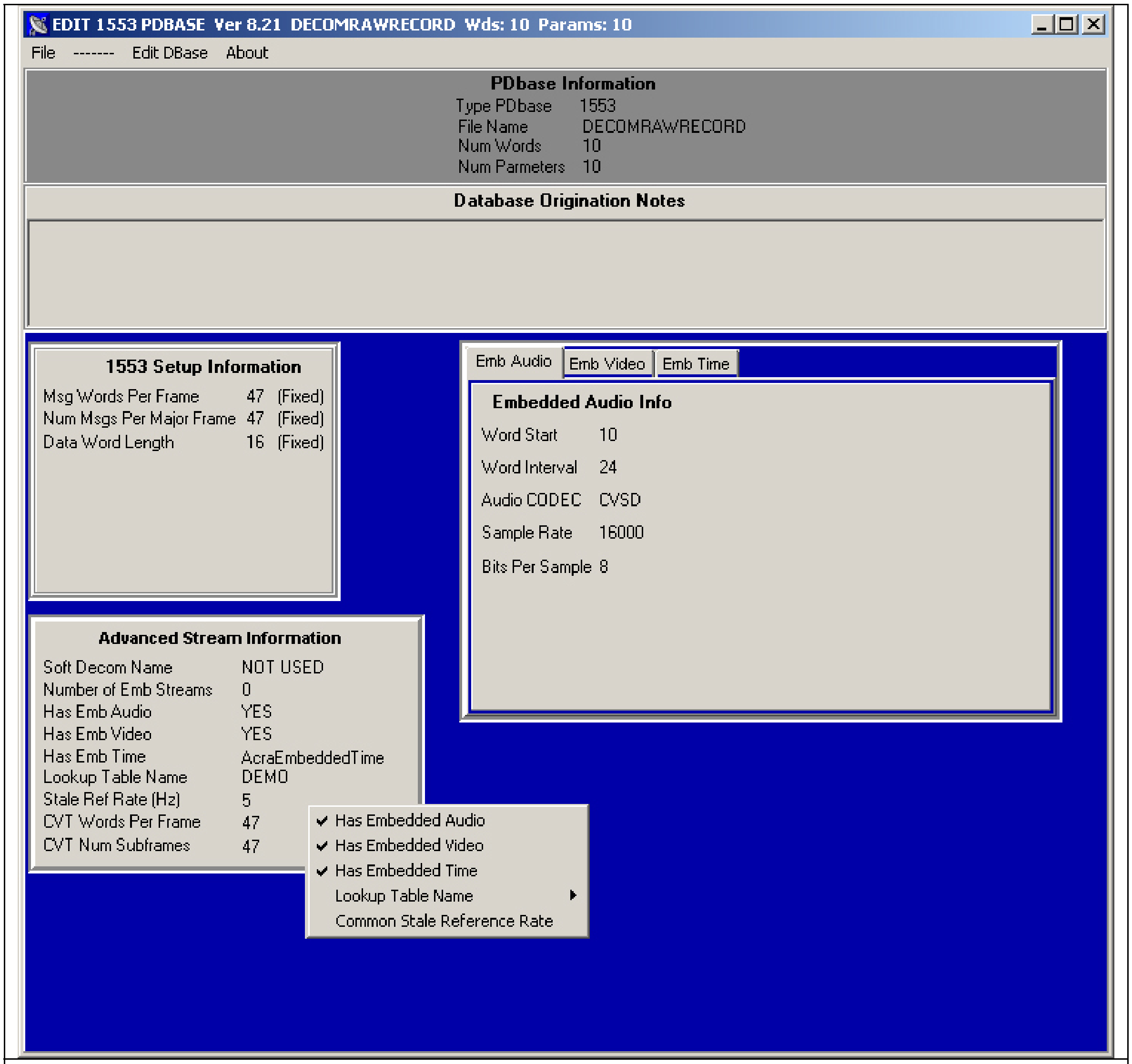
Word	Code
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0

Figure 4-19 User Defined Tab (Chapter 8)

4.3.3 MIL-STD-1553 Database Type



Selecting the **PDbase 1553** command from the servers' edit menu will invoke the **EDIT 1553 PDBASE** window shown in Figure 4-20 below. The major configuration items in this window include: 1553 Setup Information, Advanced Stream Information, and Setup Tabs for those items enabled in the advanced stream information section. The 1553 database tab edit controls are much the same as the PCM database tab edit controls described in paragraph 4.3.1.3 on page 193.



PDbase Information

Type PDbase 1553
File Name DECOMRAWRECORD
Num Words 10
Num Parmeters 10

Database Origination Notes

1553 Setup Information

Msg Words Per Frame	47	(Fixed)
Num Msgs Per Major Frame	47	(Fixed)
Data Word Length	16	(Fixed)

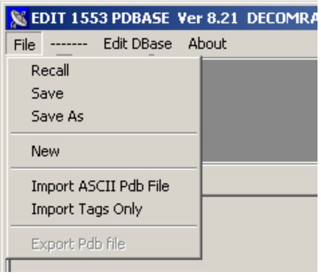
Embedded Audio Info

Word Start	10
Word Interval	24
Audio CODEC	CVSD
Sample Rate	16000
Bits Per Sample	8

Advanced Stream Information

Soft Decom Name	NOT USED
Number of Emb Streams	0
Has Emb Audio	YES
Has Emb Video	YES
Has Emb Time	AcraEmbeddedTime
Lookup Table Name	DEMO
Stale Ref Rate (Hz)	5
CVT Words Per Frame	47
CVT Num Subframes	47

Figure 4-20 Edit Parameter Database Window & Menus (1553 Type)



At the top of the window in Figure 4-20 above are two menu items: File and Edit Dbase. The **File** command is used for saving and recalling databases. The database files are stored in two formats, ASCII and binary. When the server and client(s) load a database(s), it is the binary version that is used. To modify the database, use Microsoft Excel or another ASCII text editor to edit the ASCII version of the file. Then **Recall** the ASCII version of the file in the LDPS database editor and **Save** it. This will compile the database and save both the ASCII and the binary versions. The compiled binary version of the file is used when the file recall command is invoked. When the **Import ASCII Pdb File** command is invoked, the ASCII version is recalled.

For those wishing to edit the parameter database in Excel or other ASCII editor, the user may also import just the parameter tags (**Import Tags Only**). The header information is not required for the import. Just have the first line in the ASCII file contain "*.*" and then the rest of the lines are as normal for the ASCII version. Then, use the LDPS parameter editor to set up the header information and save the file. This will yield a compiled version (both binary and ASCII). The user can also export a database to some other format, provided the export DLL assigned in the programs options.

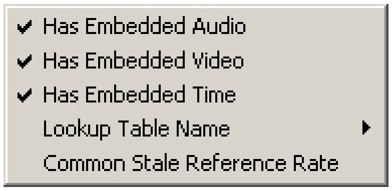
Clicking on the **Edit Dbase** command in Figure 4-20 will result in the Edit 1553 Database Window shown in Figure 4-21 on page 222. This window allows editing of the words and parameters in the parameter database.

Below the menu section in Figure 4-20 is information regarding the database that includes: The name of the database loaded for editing (File Name), the count of root words in the database (Num Words), the count of parameters in the database (Num Parameters), and database type (Type Dbase). Included types: PCM, 1553, and IRIG Chapter 8.

4.3.3.1 MIL-STD-1443 Setup Information

The 1553 Setup Information box in Figure 4-20 is only visible if the database type is 1553. MIL-STD-1553 words are 20 bits long, but only 16 bits have meaning to the user. The 1553 setup information box displays the number of message words per frame, and the number of messages per major frame. The number of words in each message is 41, and there are 128 messages buffered up before LDPS acquires the new data. In general, MIL-STD-1553 data is asynchronous, yet LDSP acquires this data a fixed rate. The update rate for 1553 data is relatively slow. The update rate in LDSP is calculated based on a 100 Hz rate (much faster than 1553 can handle). The archiving of 1553 data is also asynchronous, but in fixed block sizes of the 128 messages.

4.3.3.2 Advanced Stream Information



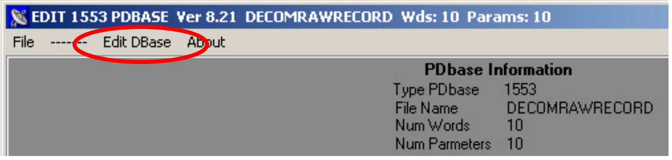
Right click in this section to invoke the menu shown right. Depending on what is entered here, some advanced tabs will appear where further information will be required.

If the stream has embedded audio, then check the **Has Embedded Audio** command (see paragraph 4.3.3.5 on page 224). This will cause a new tab to appear with additional setup information required. Likewise, if the stream has embedded video or time, then click the **Has Embedded Video** and/or **Has Embedded Time** commands respectively (see paragraph 4.3.3.6 on page 225 and paragraph 4.3.3.7 on page 226). As before, this will cause new tabs to appear with additional setup information required. For streams with embedded time, the user will be required to enter the name of the time DLL that will support the stream. If the database requires a look-up table, then the user can select from a list of available look-up tables previously generated by clicking the **Lookup Table Name** command.

The **Stale Reference Rate** command sets the rate that most parameters will update at. This command sets the default for all parameters. The state rate is used by display programs to issue alerts when data for parameters are not updating at the required rate (based on the time stamp for that parameter).

4.3.3.3 Edit 1553 Database Window

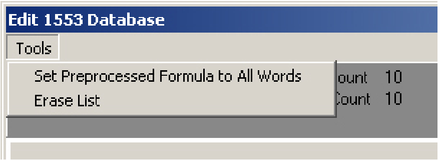
To invoke the Edit 1553 Database window shown in Figure 4-21 below, click on the **Edit DBase** command in the Edit Parameter Database window shown in Figure 4-20 on page 219 (a cutaway of which is shown below).



EDIT 1553 PDBASE Ver 8.21 DECOMRAWRECORD Wds: 10 Params: 10

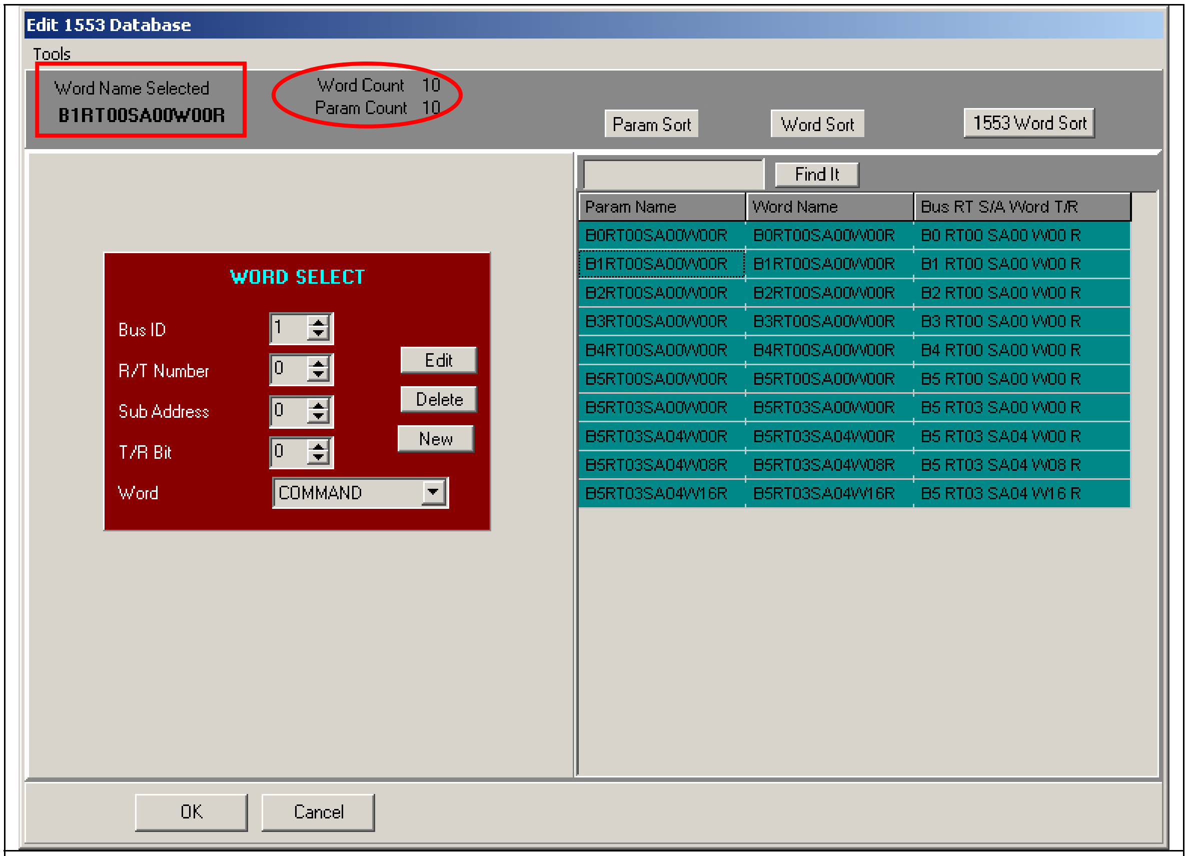
PDbase Information
Type PDbase 1553
File Name DECOMRAWRECORD
Num Words 10
Num Parmeters 10

Across the top of the Edit 1553 Database window shown right is the main menu, which currently contains the **Tools** command. The database tools include: Set Preprocessed Formula to All Words, and Erase List.



The **Set Pre processed Formula to All Words** command allows the user to apply a preprocessed formula to all existing words in the database. This includes a blank preprocessed formula, which will set the formula used to none. Selecting this command will result in the formula setup window shown in Figure 4-11 on page 204.

To start over again, without changing the header info, select the **Erase List** command.



Word Name Selected B1RT00SA00W00R

Word Count 10
Param Count 10

Bus ID 1
R/T Number 0
Sub Address 0
T/R Bit 0
Word COMMAND

Param Name	Word Name	Bus RT S/A Word T/R
B0RT00SA00W00R	B0RT00SA00W00R	B0 RT00 SA00 W00 R
B1RT00SA00W00R	B1RT00SA00W00R	B1 RT00 SA00 W00 R
B2RT00SA00W00R	B2RT00SA00W00R	B2 RT00 SA00 W00 R
B3RT00SA00W00R	B3RT00SA00W00R	B3 RT00 SA00 W00 R
B4RT00SA00W00R	B4RT00SA00W00R	B4 RT00 SA00 W00 R
B5RT00SA00W00R	B5RT00SA00W00R	B5 RT00 SA00 W00 R
B5RT03SA00W00R	B5RT03SA00W00R	B5 RT03 SA00 W00 R
B5RT03SA04W00R	B5RT03SA04W00R	B5 RT03 SA04 W00 R
B5RT03SA04W08R	B5RT03SA04W08R	B5 RT03 SA04 W08 R
B5RT03SA04W16R	B5RT03SA04W16R	B5 RT03 SA04 W16 R

Figure 4-21 Edit Database Window (1553 Type)

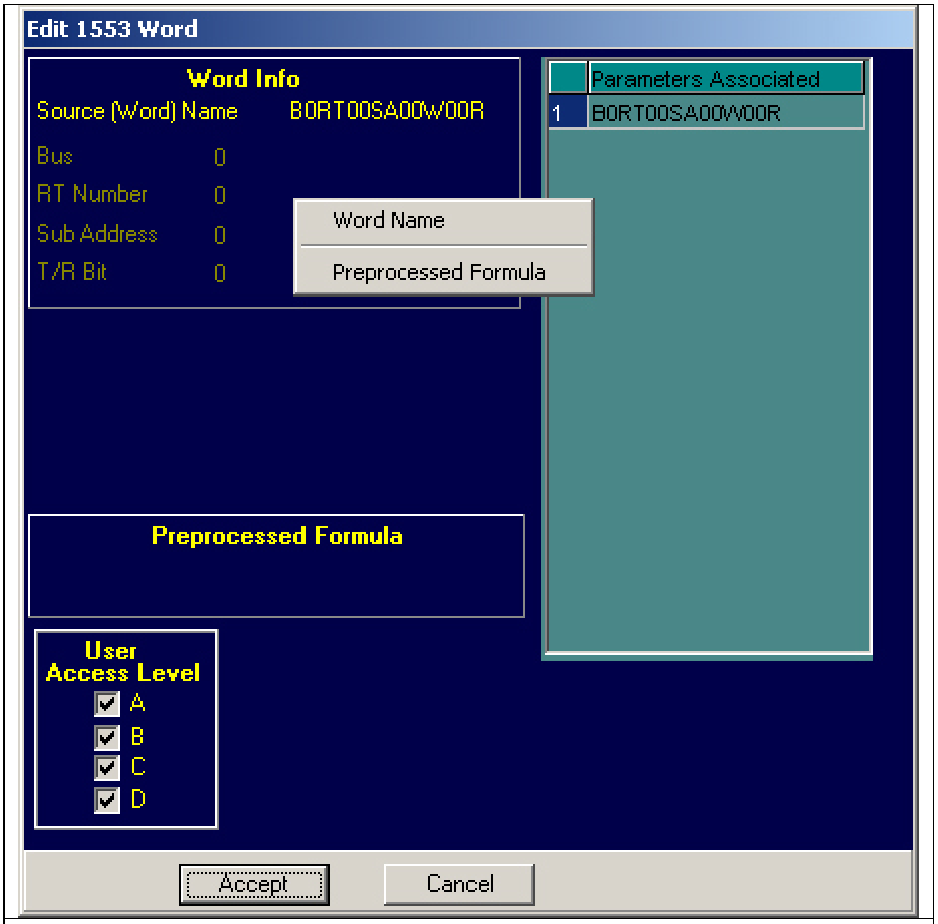
Below the main menu in the figure above is an information area (red oval) displaying the number of parameters (Param Count) and the number of words (Word Count) currently in the database. On the left of this area is the word name currently being editing (red rectangle).

The Word Select Area allows the user to add, edit, or delete word information. Here the user selects the bus ID, RT number, Sub address, T/R bit position, and 1553 word number for the selected database word. If a database word is added or edited, the user will be taken to the word editor described in paragraph 4.3.3.4 on the following page, where the user may add a preprocessed formula, set user access levels, and add additional parameters to that root word.

To the right of the word select area is the list of parameters currently in the database. Double clicking on a parameter will invoke the Parameter Definition Window shown in Figure 4-12 on page 206. There is no option to change the Word Information or create/delete parameters with this window. The Parameter Definition window is strictly an edit function of existing parameters. The parameter definition function is common among all database types (Chapter 4, Chapter 8, and 1553)

4.3.3.4 Edit 1553 Word

The 1553 word editor is shown in Figure 4-22 below. There are four sections to configure in the 1553 edit word window. The configuration items include: Word Info, Preprocessed Formula, Associated Parameters, and User Access Level. Each configuration item is discussed in the paragraphs that follow.



Edit 1553 Word

Word Info

Source (Word) Name B0RT00SA00W00R

Bus 0

RT Number 0

Sub Address 0

T/R Bit 0

	Parameters Associated
1	B0RT00SA00W00R

Preprocessed Formula

User Access Level

☑ A
☑ B
☑ C
☑ D

Figure 4-22 Edit 1553 Word Window & Menus

The Word Info section displays the name of the source word, the Bus & RT number, the Sub address, and the R/T Bit position.

The User Access Level allows the user to define various levels of access to the database. This function is for system administrators that want to control the distribution of the database to different user groups, where certain groups cannot access the data. It is recommended that all four boxes be checked, unless great care is taken.

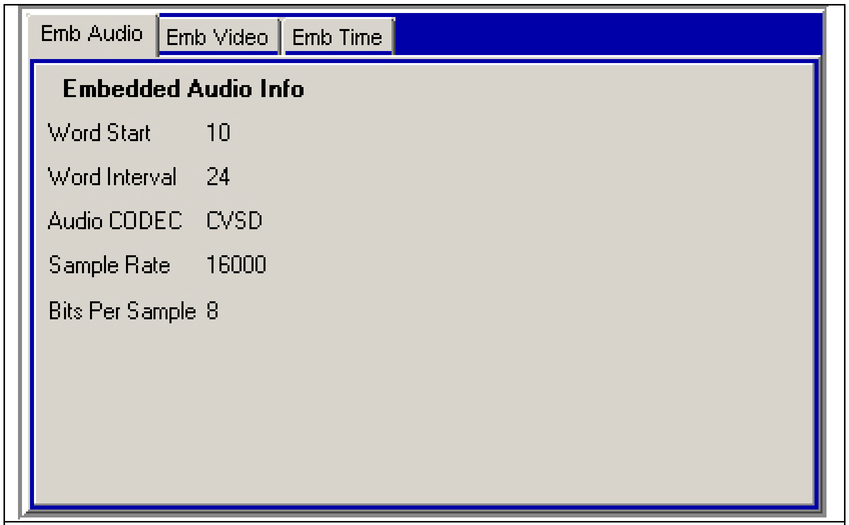
The Preprocessed Formula command allows the user to define a preprocessed formula for the selected word, prior to any processing of the parameters. The formula entered will be an integer function and every parameter associated with the word will have the same function(s) applied prior to solving for the parameter. Multiple functions may be applied to the selected word by separating each function with a semicolon. To invoke the preprocessed formula, right click in the area and select “Preprocessed Formula.”

On the right of the word editor shown in Figure 4-22 on page 223, is a list of defined parameters that are associated with the selected word. The minimum number of parameters that may be associated with a word is 1. The maximum is 256. Right click in this area to Add, Edit, or Delete a parameter. To Edit or Delete a parameter, first select it by left clicking on the parameter. Editing or adding a new parameter will invoke the parameter definition window shown in Figure 4-12 on page 206. The parameter definition function is common among all database types (Chapter 4, Chapter 8, and 1553)

Once the configuration information has been entered, click the **Accept** button. To discard any changes, press the **Cancel** button, which will return the user back to the edit 1553 database window (see Figure 4-21 on page 222).

4.3.3.5 1553 Embedded Audio Tab

The Embedded Audio tab allows the user to set up any embedded audio streams. To enable the embedded audio functionality in the parameter database, select the **Has Embedded Audio** command from the **Advanced Stream Information** menu described in paragraph 4.3.3.2 on page 221. The **Word Start** indicates the starting word in the primary frame where the embedded audio begins (1 based index).



Embedded Audio Info

Word Start 10

Word Interval 24

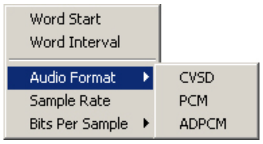
Audio CODEC CVSD

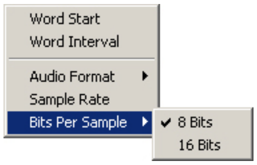
Sample Rate 16000

Bits Per Sample 8

Figure 4-23 Embedded Audio Tab (MIL-STD-1553)

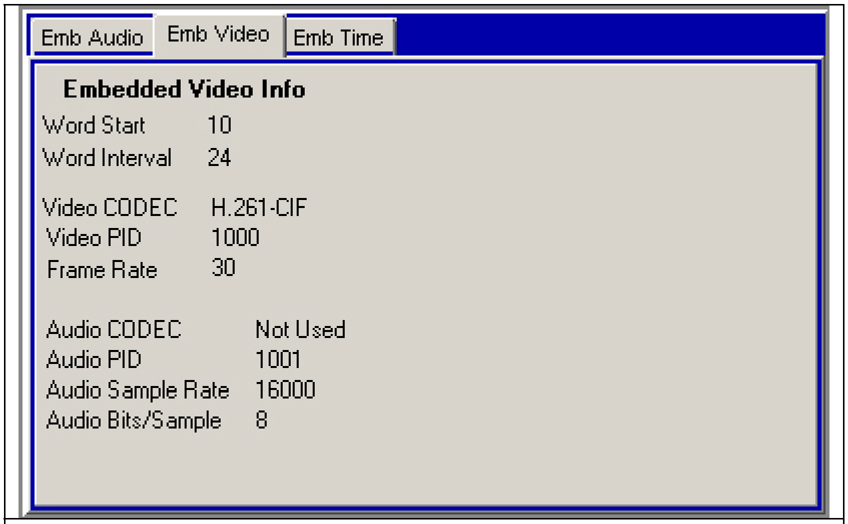
The **Word Interval** indicates how the embedded audio is commutated within the primary frame. The **Audio Format** command selects the encoding format the embedded audio was encoded with (the default is CVSD). Other CODECs include PCM and ADPCM. The **Sample Rate** selects the sample/second that the audio was encoded at (the default is 16 KHz). Eight or sixteen bits/sample may be selected by the user via the **Bits Per Sample** Command.





4.3.3.6 1553 Embedded Video Tab

The Embedded Video tab allows the user to set up any embedded video streams. To enable the embedded video functionality in the parameter database, select the **Has Embedded Video** command from the **Advanced Stream Information** menu described in paragraph 4.3.3.2 on page 221. The **Word Start** indicates the starting word in the primary frame where the embedded video begins (1 based index). The **Word Interval** indicates how the embedded video is commutated within the primary frame.



Embedded Video Info

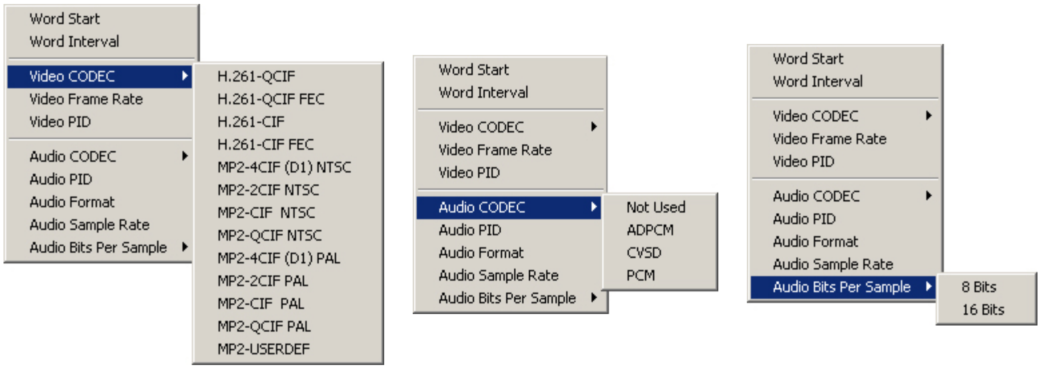
Word Start 10
Word Interval 24

Video CODEC H.261-CIF
Video PID 1000
Frame Rate 30

Audio CODEC Not Used
Audio PID 1001
Audio Sample Rate 16000
Audio Bits/Sample 8

Figure 4-24 Embedded Video Tab (MIL-STD-1553)

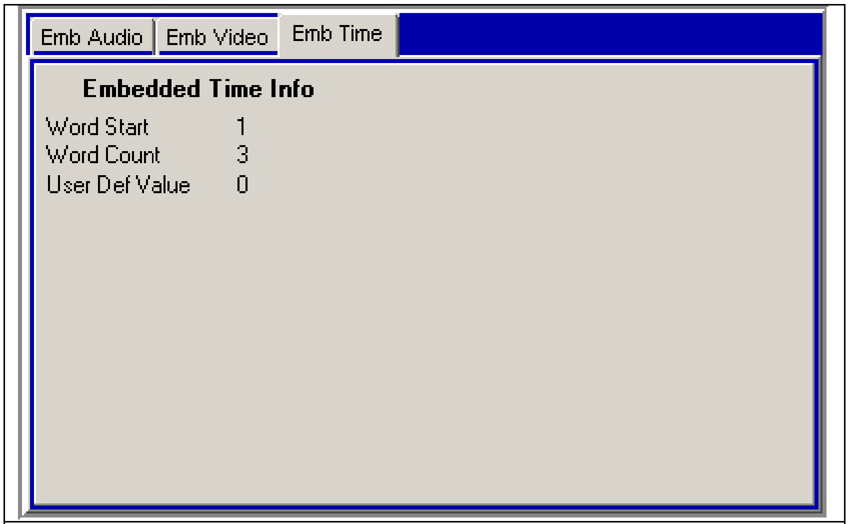
The **Video CODEC** command selects the encoding format the embedded video was encoded with (the default is H.261 - CIF). Some of the other supported formats include: H.264-QCIF, H.264-CIF FEC, MP2-4CIF. The complete list of supported CODECs is shown below left.



The **Video Frame Rate** selects the frames/second that the video was encoded at (the default is 30 fps). The **Audio CODEC** command selects the encoding format the audio portion of the embedded video was encoded with (the default is CVSD). Other CODECs include PCM and ADPCM. The **Audio Sample Rate** selects the sample/second that the audio was encoded at (the default is 16 KHz). Eight or sixteen bits/sample may be selected by the user via the **Audio Bits Per Sample** command.

4.3.3.7 1553 Embedded Time Tab

For MIL-STD-1553 streams that have embedded time, the user can either use the parameter database to collect the time and decode it or use a display widget function. To enable the parameter database to decode the embedded time data, the user must select the appropriate DLL to decode the time. To enable the embedded time functionality in the parameter database, select the **Has Embedded Time** command from the **Advanced Stream Information** menu described in paragraph 4.3.3.2 on page 221. After selecting the file name of the embedded time DLL, a new tab (shown in Figure 4-25 below) will appear.



Embedded Time Info

Word Start 1
Word Count 3
User Def Value 0

Figure 4-25 Embedded Time Tab (MIL-STD-1553)

The **Word Start** indicates the starting word in the primary frame where the embedded time data begins (1 based index). The **Word Count** indicates how the embedded time is commutated within the primary frame.

5 Software Decommutator

Some data streams are just too complex for a standard decommutator to decode and provide useful data. Certain words, for example, might only get updated if another group of words appearing in a strange sequence have the correct corresponding data values. Streams like the one described in IRIG-106 Chapter 8 are a prime example. For this reason, a software decommutator can be invoked to put the data into a useful format.

When a user defines a project, there is usually at least one serial data stream. Each of these streams can have a software decommutator associated with it. The software decommutators are DLLs written or designed by the user that manipulates raw data in the stream according to a set of rules defined in a database. The process in LDPS that occurs on the server when new data is received from the decommutator is as follows:

One-

- Copy the data from the hardware into system memory.
- Archive the data if required.
- Pass the data on to the software decommutator if required.

Two-
On the client side (with data at a rate selected by the Data Transmit Slide Control), collect the client interest data from:

- Software decommutator if selected (passed to serial CVT).
- Serial CVT data.
- Device CVT data.

Three-
Send the client interest data to the clients.

To write a software decommutator, one must follow the rules given below. The IRIG-106 Chapter 8 software decommutator included with LDPS was written by Lumistar. Once the users' software decommutator has been written, place the resulting DLL into the \LDPS_8x\Bin\SoftDecomDlls subdirectory.

It is most important for the user to debug the DLL fully, with lots of error checking.

5.1 API Calls

There are eleven DLL functions called by LDPS. The users' software decommutator must handle up to four data streams. If it is designed to only handle one, then only the first stream to use the software decommutator will work. In LDPS, the software decommutator is very tightly coupled to the parameter database. (See paragraph 1.3 on page 7). The DLL function calls are as follows:

```
bool SoftApiOpen(int numdecomsrequired);
void SoftApiClose(void);
```

```
       bool SoftApiInit(int decomindex,
                             double commonwordupdateratehz,
                             int pcmheaderwordcount, int minorspermajor,
                             int wdsperframe, int bitsperword, int fpi,
                             bool rightaligned, bool msbfirst, int modeary[]);
       bool SoftApiReset(int decomindex,unsigned short fillwithvalue);
       bool SoftApiStartTask(int decomindex);
       bool SoftApiStopTask(int decomindex);
       bool SoftApiFeedRawData(int decomindex,LPVOID databuf);
       bool SoftApiGetDataValue(int decomindex, int addrary[], unsigned int &value);
       bool SoftApiGetTime(int decomindex, double &toy);
       bool SoftApiGetStatus(int decomindex, int &status);
       bool SoftApiGetLastError(int decomindex, int &ercode, char *erst);
```

Below is the software listing of the Lumistar Chapter 8 software decommutator. The rules are inside each function and must be followed exactly.

```
       //---------------------------------------------------------------------------
       /*
       A single softdecom dll is used for all soft decoms with the same functions,
       i.e. for ch8 soft decom, if there are 2 streams using ch8 soft decom, then
       only the one dll is called. the program must pass in whether it is the first
       ch8 softdecom or the 2nd ch8 decom. NOT TO BE CONFUSED WITH STREAM NUMBER.
       A project could have stream 0 using ch8soft, stream 1 using aim9rsoft,
       stream 2 using ch8soft, and stream3 using jsowsoft. When calling this dll
       for the ch8soft, the decom indexes would be 0 and 1, NOT 0 and 3. It is
       up to the program to know how many streams are using the same softdecom
       and keep the indexes straight. This should probably be done at project
       load time. (keep an array of softdecom names, up to max streams. load
       the required dlls once, and assign the softdecom index for each)
       */
//---------------------------------------------------------------------------
/*
VERSION CHANGES
5/3/01 Ver 1.1
       - added a thread to pull the data out of SoftApiFeedRawData(int decomindex,LPVOID
databuf)
       so could return to the program asap
       Added error log if missed processing a new pcm data event
       It takes about 0.00032 seconds to do a single 512 word frame. Put a Sleep(2)
       in the call after the event is fired to do the ch8 conversion so there
       would be some time to process. Approx 1 msec for 3 fpi, so you really
       don't want to go much more than 30 fpi when setting up the decom, especially
       if you have multiple fast soft decoms
5/8/01 Ver 1.2
       - changed SoftApiReset(int decomindex) to
       SoftApiReset(int decomindex,unsigned short fillwithvalue)
       so could have the decoders set all values to a known point
*/
//---------------------------------------------------------------------------
#undef FORSOFTDECOMDLL_IMPORT
#define FORSOFTDECOMDLL_EXPORT

#include "BorLibPch.h"
#include "CommonUtilsUnit.h"
#pragma hdrstop

#include "Ch8DllInterfaceUnit.h"
#include "SoftDecomDllApiUnit.h"
//---------------------------------------------------------------------------
//local vars
bool ApiOpenCalled = false; //if open has been called
#pragma argsused
//---------------------------------------------------------------------------
BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fwdreason, LPVOID lpvReserved)
{
       if (fwdreason == DLL_PROCESS_ATTACH)
         ApiOpenCalled = false;
       return 1;
}
//---------------------------------------------------------------------------
bool SoftApiOpen(int numdecomsrequired)
//THIS MUST BE THE FIRST CALL MADE BEFORE MAKING ANY OTHER DLL CALLS.
//pass in the number of decoms required
{
AnsiString erst;
        if (ApiOpenCalled)
        {
        LogIt("SoftDecomCh8 was not closed first, must close it first. Open Failed.",1);
        return false;
        }
        if (!DllProcessAttach())
          return false;

        ApiOpenCalled = OnApiOpen(numdecomsrequired,erst);

        if (ApiOpenCalled)
          LogIt("SoftDecomCh8..SoftApiOpen() Success",0);
        else
          LogIt("SoftDecomCh8..SoftApiOpen() Failed " + erst,1);
        return ApiOpenCalled;
}//softapiopen

//---------------------------------------------------------------------------
void SoftApiClose(void)
//the last call made before unloading dll
{
        if (!ApiOpenCalled)
          return;
        DllProcessDetach();
}//softapiclose
//---------------------------------------------------------------------------
bool __fastcall OkToMakeCall(int decomindex, AnsiString callingproc)
//called from api calls, check if ok to proceed
{
AnsiString erst = "";

        if (!ApiOpenCalled)
          erst = "CH8 APIOPEN NOT CALLED";
        else if ((decomindex < 0) || (decomindex >= gNumDecoms))
          erst = "CH8 DECOM INDEX INVALID, MUST BE IN RANGE 0.." + NumToStr(gNumDecoms -
        1,2,0);

        if (erst != "")
        {
         erst = callingproc + " call for ch8decom index " + NumToStr(decomindex,5,0) + "
... "
        + erst;
          LogIt(erst,1);
          return false;
        }

        return true;
        }//OkToMakeCall
//---------------------------------------------------------------------------
bool SoftApiInit(int decomindex,
        double commonwordupdateratehz,
        int pcmheaderwordcount,
        int minorspermajor, int wdsperframe, int bitsperword, int fpi, bool
rightaligned, bool msbfirst,
        int modeary[])
//init the indexed soft decom
//commonwordupdateratehz update rate of most of the data words in the stream
//pcmheaderwordcount number of header words in pcm frame (5 if comming from lumistar
or sbs decoms)
//minorspermajor number of minor frames per major frame from pcm decom setup
//wdsperframe number of words per frame comming from pcm decom setup
//bitsperword number of bits per word comming from pcm decom setup
//fpi minor frames per interrupt from pcm setup (number of minor
frames of data in buffer)
```

```
//rightaligned data from the pcm decom is right aligned (true) or left aligned
(false), rem all words fed are 16 bit, you gotta extract bitsperword
//msbfirst data from pcm decom is msb first..a must for ch8
//modeary[] softdecom mode controls, defined in the database when database
generated
// (should match what soft decom is looking for)
// there are 16 elements in the mode ary, you pick the ones
you want

{
       if (!OkToMakeCall(decomindex,"SoftApiInit"))
         return false;
       DecomAry[decomindex]->ApplyInitData(commonwordupdateratehz, pcmheaderwordcount,
                                  minorspermajor, wdsperframe, bitsperword, fpi,
                                  rightaligned, modeary);
       return DecomAry[decomindex]->InitDecom();
}//softapiinit
//-----------------------------------------------------------------------------
bool SoftApiReset(int decomindex,unsigned short fillwithvalue)
//reset the indexed soft decom, fill the data in with fillwithvalue
{
       if (!OkToMakeCall(decomindex,"SoftApiReset"))
         return false;
       return DecomAry[decomindex]->ResetDecoders(fillwithvalue);
}//softapireset
//-----------------------------------------------------------------------------
bool SoftApiStartTask(int decomindex)
//start the indexed soft decom task
{
unsigned short fillwithvalue = 0;
       if (!OkToMakeCall(decomindex,"SoftApiStartTask"))
         return false;
       //set last errors
       DecomAry[decomindex]->LastError = 0;
       DecomAry[decomindex]->LastErrorStr = "NONE";
       //nothing to do for ch8 cause not using messages except reset the decoders
       return DecomAry[decomindex]->ResetDecoders(fillwithvalue);
}//softapistarttask
//-----------------------------------------------------------------------------
bool SoftApiStopTask(int decomindex)
//stop the indexed soft decom task
{
       if (!OkToMakeCall(decomindex,"SoftApiStopTask"))
         return false;
       //nothing to do for ch8 cause not using messages
       return true;
}//softapistoptask
//-----------------------------------------------------------------------------
bool SoftApiFeedRawData(int decomindex,LPVOID databuf)
//feed raw data buff to the indexed soft decom
//this won't be called unless the data from the pcm decom is valid (frame lock)
//It is your responsibility to return from this routine as quickly as possible
//so the next data can be retrieved. If you take too long, then the
//system will suffer and this along with other streams will lose data.
//Your best bet is to copy the data to a local buffer and run a thread.
{
       if (!OkToMakeCall(decomindex,"SoftApiFeedRawData"))
         return false;

       DecomAry[decomindex]->DoGotNewPcmData((unsigned char *)databuf);
         return true;
}//softapifeedrawdata
//-----------------------------------------------------------------------------
bool SoftApiGetDataValue(int decomindex, int addrary[], unsigned int &value)
//get the indexed soft decom value, based on the addrary passed in
//the addrary passed in is set up in the database when the database is created.
//it should match the softdecoms method to decode which value to get
//there are 8 elements in the addrary, you pick out the ones you want
{
       if (!OkToMakeCall(decomindex,"SoftApiGetDataValue"))
       {
         value = 0;
         return false;
       }
       return (DecomAry[decomindex]->GetWordValue(addrary,value));
}//softapigetdatavalue
//---------------------------------------------------------------------------
bool SoftApiGetTime(int decomindex, double &toy)
//get the indexed soft decom embedded time (time of year in seconds)
{

       if (!OkToMakeCall(decomindex,"SoftApiGetTime"))
       {
         toy = 0.0;
         return false;
       }
       toy = DecomAry[decomindex]->Ch8Decom->GetHpmFrameTime();
       return true;

}//softapigettime

//---------------------------------------------------------------------------
bool SoftApiGetStatus(int decomindex, int &status)
//get the indexed soft decom status
{
       if (!OkToMakeCall(decomindex,"SoftApiGetStatus"))
         return false;

       DecomAry[decomindex]->GetDecomStatus(status);
       return true;

}//softapiGetStatus

//---------------------------------------------------------------------------
bool SoftApiGetLastError(int decomindex, int &ercode, char *erst)
//erst needs to be at least 1024 chars long
//get the indexed soft decom last error code and string
{
AnsiString s;

       if (!OkToMakeCall(decomindex,"SoftApiGetLastError"))
         return false;

       ercode = DecomAry[decomindex]->LastError; //last error received
       s = DecomAry[decomindex]->LastErrorStr;
       if (s.Length() > 1024)
          s = s.SubString(1,1024);
       strcpy(erst, s.c_str());

       return true;

}//softapigetlasterror
```

5.2 IRIG 106 Chapter 8 Software Decommutator

The LDPS software decommutator for IRIG-106 Chapter 8 data follows rules set down by the IRIG standard, as well as from a collection of notes and known attributes of live streams of data from a variety of actual programs (see paragraph 5.4 on page 233 for the notes).

A careful examination of the IRIG-106 standard reveals the need for a large number of options to be included in the software decommutator. These options are set up in the parameter database hardware fields (see paragraph 1.3.4 on page 14), or in the header portion of any parameter database).

The IRIG-106 standard does not overtly specify what to do if a parity error occurs. Therefore, the user has the option to allow the decoder to perform different functions, depending on the parity mode. The parity modes are as follows:

- DONTCARE – Don’t check for parity error.
- HOPE - Process as if no error and hope for the best.
- NOWORD - Don't process the word. If it is a frame time word, then don't process the frame of data. If it is a data word, then count it as a data word but don’t store the data value.
- RESET - Reset the decoder for that bus, don't process any more on that bus until a new command word arrives.

5.3 Decoder Status Values

When the status of the decommutator is retrieved, the following values can be observed:

```
//message complete error codes
#define MSGCOMPLETETIMEOUTERROR      4 //msg wasnt completed within timeout spec..assume
tm
dropout
#define MSGCMDWDTOOSOONERROR         8 //got a new command word before previous command
word finished
#define TOOMANYDATAWORDSERROR        16 //data word count > 33
#define MSGNOGOTSTATUSWORDERROR      128
#define MSGNOGOTALLDATAWORDSERROR    256
#define MSGNOGOTLOITWORDERROR        512
#define MSGNOGOTUSTWORDERROR         1024
#define MSGNOGOTHOITWORDERROR        2048
#define MSGMOREDATATHANEXPECTEDERROR 4096 //data word count exceeded words to follow in
command word
#define MSGDATAWORDPARITYERROR       8192 //got parity error, mode is dont process
except data word
#define MSGNORINGBUFFERROR           16384 //msg ringbuffer is null..program didnt
allocate any memory
#define MSGNORINGBUFFEMPTYERROR      32768 //msg ringbuffer is null..program didnt
allocate any memory
//health info error codes
#define FRAMEUSTTIMEOUTOFORDERERROR  1
#define FRAMELOITTIMEOUTOFORDERERROR 2
#define FRAMEHOITTIMEOUTOFORDERERROR 4
#define FRAMETIMEPARITYERROR         8 //one of the frame time words had parity error
with parity checking on
```

```
#define MSGRINGBUFFFULLERROR          16 //msg ringbuffer is full..user not pulling data
out fast enough
#define USERSTOPPEDTASK               32 //user called stop_hpm_task
#define DECOMNOTINITED                64 //init_decom not been called
//user interface error codes
#define USERERRORFRAMESIZE            1 //user entered invalid pcm frame size
#define USERERRORPARITYMODE           2 //user entered invalid parity error mode..resorts
//to parityhopemode
#define USERERRORNUMINTERESTMSGS      4 //user entered invalid number of msgs to monitor
//in interest table..see max msgs
#define USERERRORMSGTAGNUM            8 //user entered invalid encoded msg to monitor in
//interest table (value > 3FFF)
#define USERERRORNOMSGSINLIST         16 //user sez want interest list, but no messages
```

5.4 Some Notes on Lumistar's Chapter 8 Software Decommutator

This is a collection of notes on how the Lumistar Ch8 soft decom processes **IRIG-106-93** Chapter 8 (all bus) data, and the deviation of this specification used by F18 at China Lake. Lumistar wishes to thank Ms. Teresa Telles and Mr. Darrel Douglas of NAS China Lake for their contributions to this project. The notes are incomplete, but should give enough information to get started on developing a software decommutator for other Chapter 8 applications/implementations.

Chapter 8 data words - For each word (24 bits), the first 8 bits define PARITY, BUS ID, and TYPE WORD.

```
x xxx xxxx xxxx xxxx xxxx xxxx
p bus type
```

For the most part, PARITY is not used for F-18 and AV-8 systems, but it is desirable to have a counter for parity errors. The BUS ID identifies which bus (0..7) the data belongs to. TYPE identifies the type word it is (0..15).

Command Word - When a command word is received, the lower 15 bits complete the addressing and the number of data words to follow. Recall that the first 8 bits contained the bus id and type word. The number of data words to follow cannot be relied upon in the decoding scheme, so it is not advisable to process these bits, unless the message number is 0 or 31. In which case the last 5 bits are discrete, and are called MODE CODE. These bits are the actual command word issued by the bus controller. There times when analysts want to examine the MODE CODE value. Most of the time, there should be no data words following the message if the command word is a mode code. However, if the value of the mode code is less than 16, then no data words follow, otherwise one data word will follow. One should have a way to retrieve the command word value. The bit break out of the command word is as follows:

```
x xxx xxxx xxxxx xxxxx  x  xxxxx
p bus type rt #  msg # t/r datawds
```

where:

p = parity
bus = bus ID (0..7)

type = type word (0..15)
rt # = RT number (0..31)
msg # = message number (sub address) (0..31)
t/r = transmit or receive bit (1=T 0=R) with respect to TERMINAL
datawds = number of data words to follow (or mode code) (0..31)

The t/r bit can be confusing on the aircraft because usually the mission computer is the bus controller. The number of data words to follow a command word is sometimes used, sometimes not (sometimes it's even correct). A general rule is to count the data word as being correct as long as it is less than word 33.

Status Word – One will not get a status word on a broadcast command (when the RT number is 31). When a status word is received, the lower 16 bits identify the RT number and the status of the message sent to the RT. The rest of the addressing is based upon the last valid command word for that bus. The status bits are broken down as follows:

```
x xxx xxxx xxxxx x x x xxx x x x x x
p bus type   rt #      status
                     msb          lsb
```

where:

p = parity
bus = bus ID (0..7)
type = type word (0..15)
rt # = RT number (0..31) communicating with (self if BC-RT or RT-BC, other if RT-RT)
status = status bits (don't need to break out, but if one did, break out as follows:)

```
        msb            msg error
                       instrumentation
                       service request
        reserved
        reserved
        reserved
                       broadcast command received
                       busy
                       subsystem flag
        dynamic bus control acceptance
                       terminal flag
lsb     parity
```

Table 5-1 Chapter 8 Data Word Types		
Number	**Type**	**Definition**
0	Buffer Overflow	The data in the aircraft instrumentation buffer overflowed, therefore the data is corrupted. If this type word is detected, you should wait until a new command word is detected and reset the message decoder.
1	Fill	The data is fill data AAAA. All bus data transmitted is asynchronous but it must be transmitted synchronously. Therefore, fill data is inserted to accommodate that.
2	User Defined 1	
3	User Defined 2	(For F18 and AV8 aircraft, this word contains the voice discretes (hot mic).
4	Response Time	This is the response time of the RT. Some projects do not use this type word.
5	Microsecond Time	This is the BCD time in microseconds of when the bus controller issued the command word, or the microsecond time when the PCM stream was sent.
6	Low Order Time	This is the BCD time in seconds (LSW) of when the bus controller issued the command word, or the time when the PCM stream was sent.
7	High Order Time	This is the BCD time in seconds (MSW) of when the bus controller issued the command word AND the time when the PCM stream was sent.
8	Error Bus B	The all bus system is a dual redundant bus..2 busses, bus A and Bus B (not to be confused with bus id 0..7). This means that if something is wrong with one bus, the controller will switch over to the other bus. If this type word is encountered, the bus controller has detected a hardware failure. The data will be invalid so the decoder should wait for another valid command word before continuing processing for that bus.
9	Data Bus B	The word contains data to be processed based on the last command word received for that bus.
10	Status Bus B	The word contains the status of the message sent.
11	Command Bus B	This word contains the addressing for the words that follow for that bus.
12	Error Bus A	Same as type word 8 but for bus A.
13	Data Bus A	Same as type word 10 but for bus A
14	Status Bus A	Same as type word 10 but for bus A
15	Command Bus A	A Same as type word 11 but for bus A

In IRIG-106 Chapter 8, the amount of data to be processed is quite large. There can be up to eight (8) busses, with each bus containing up to thirty-two (32) RTs. Each RT can contain up to thirty-two messages, with each message containing up to thirty-two words. The message could be a transmit or receive message. Multiplying all this out, one has 8*32*32*32*2 = 524,288 words. This doesn't include the status, command, time, or user defined words. Because memory prices are low, the structure being used will define the "allbus" to have forty (40) words instead of thirty-two (broken out as: command word, low order time, microsecond time, status word, response time, mode code, user def 1, user def 2, and 32 data words). This structure makes decoding faster, but wastes memory. Therefore, the memory requirements will increase to 655,360 words for the data. The word number meaning for this discussion is as follows:

WORD	MEANING
0	command word
1..32	data words
33	status word
34	message time hi time
35	message time low time
36	message time micro
37	user def word 1
38	user def word 2
39	mode code
40	response time

This does NOT cover RT- to-RT messages, where there are two command words, two status words, and optionally two response words, and two sets of message times. The structure would just put the words on top of the previous type word. This may or may not be adequate for a complete decoder.

Decoding the “allbus” data is almost simple. As previously mentioned above, the PCM stream contains 256 words. The first word is the frame sync pattern (FAF320). The next 255 words are the Chapter 8 data words. The first three data words after the frame sync pattern are the PCM frame time (type words 7, 6, and 5). These word meanings are fixed. After that, the data can come in any order, for any bus. One will need to have an array of eight command word latches, one for each bus. When a command word is received, update the latch for that bus. The data that follows will use the addressing in the command word latch for the bus identified in the upper eight bits. Following the command word is the message time. Note that the message time type words are also type 6 and 5 (for low order and micro), just like for the frame time. Establish if the type words (5 and 6) are for frame time, or for message time by determining if the type word is located within the first three words after the frame sync pattern (for PCM frame time). If the type words are after the first three words following the frame sync pattern, then they are for message time. Note, the next two words that follow a command word are the message time words. Fill data can be in between. Also, one may get type word 6 on one frame, and type 5 on the next frame.

Data words have no counter; so knowing where to put the data becomes a problem unless there is a counter. This becomes difficult if there are TM dropouts because one doesn’t know if the command word was updated. For TM dropouts, or some other error, it is recommended that the updating of data words should stop for that bus until a new command word is received.

For analysis purposes, one should employ counters for the various types of data (i.e., count the amount of data processed for bus A versus bus B). Also, bus loading can be computed with the appropriate counters. Each bus has a 1 Megabit maximum capability. Using message time and the amount of data for each bus, the bus loading can be computed.

Parity errors - To determine if there is a parity error, add all the set bits of the 24-bits (including the parity bit). The result should be ODD. Parity errors should be counted and made available to the user. I. e., a 24 bit word. For exmple:

1DFFB9 = 0001 1101 1111 1111 1011 1001 = 17 set bits = odd = good
9DFFB9 = 1001 1101 1111 1111 1011 1001 = 18 set bits=even=bad

The problem with parity errors is, *"What do I do if I get one?"* (other than count it). One should have the option of doing one of the following:

- Process as if no error and hope for the best.
- Don't process the word. If it is a frame time word, then don't process the frame of data.
- Reset the decoder for that bus, don't process any more on that bus until a new command word is received.

When checking the message time to see if it goes backwards, one must allow for some slop (like a millisecond) if the command word decoder for last message time is being used. One could keep a running tab of the message time for each bus, RT, T/R bit, and message instead. The command word decoder message time can be off by that much (based upon real-world observations).

In theory, there are three types of communication that take place on a 1553 bus. These include: RT-to-BC, BC- to-RT, and RT-to-RT. If the T/R bit is set to 0 (Rx BC-to RT), then the order of words is: command, time, data, and status. If the T/R bit is set to 1 (Tx RT-to-BC), then the order of words is: command, time, status, and data. If the message is RT-to-RT, then the order of words is: command, time, command, time, status, data, and status (where the first command and first time and LAST status come from one RT, and the second command, second time, and FIRST status come from the other RT). No status word will be received on a broadcast command (when the RT number is 31).

Looking at actual data from an F18 at China Lake showed that these laws were not followed.

6 Getting Started

Learning to use a new item of high technology for the first time is never an easy endeavor, and no two people will approach it in the same way. This chapter is intended as a place to start the process of learning how to use LDPS.

6.1 Quick Review

By way of quick review, the following summarizes some of the more salient concepts that effect how LDPS operates.

- In LDPS, there are two major architectural entities: the server, and the client(s). Each is a powerful application in their own right and they work together to acquired, archive, process, and present a wide variety of telemetry data.
- The servers' job is to collect data from the hardware, do some manipulation of the data, and distribute the data to the clients and/or the hard disk drive.
- LDPS is a project-oriented application. In order for data to be distributed, a project must be loaded. A project contains information about the streams of data and the hardware that collects the data.
- Each project can contain up to twelve (12) streams of serial data and twelve streams of non-serial data. Each stream consists of a hardware device, data produced by the device, and possibly a serial database associated with the device.
- The processing of data is based on the parameter name. The user only has to know the parameters' name in order to have processing occur on that parameter.
- Each stream contains a current value table (CVT) for the device and a parameter list. Serial devices also have a serial CVT and a parameter list. The device parameter list is fixed and is created by the system (cannot be edited by the user). The user creates the serial parameter list.
- The processing of parameters can take many forms, including the use of a Look-up Table.
- A hardware device can be set up and raw data can be monitored without a project being loaded on the server. In this scenario, none of the clients will receive the data.

6.2 Begin The Process of Using LDPS

To get started with LDPS, follow the steps below in the order listed.

First consider the following questions about the project. Is this a stand-alone machine or are there remote client machines to be connected? What hardware is installed on the machine (decommutators, receivers, etc)? Is the hardware going to provide only a quick look at the data, or is the data also going to be processed? Is the system going to be used for playback or a live operation? What kind of telemetry stream is this (PCM, Chapter 8, etc)? Does a database have to me made from a third party data dictionary? What set ups are required for the device(s) installed in the system?

If remote clients are going to need data, then click on the **Net Admin** menu *(System → NetAdmin.* on the server and fill in the information required to define the network. See paragraph 1.6 on page 35 for more details on network administration.

Configure the hardware devices installed on the machine via the Device Manager on the server *(System → Devices → Manage)*. See paragraph 2.12 on page 63 for more details on machine configuration. For each device installed on the machine, set up the device and then save the configuration in a set up file.

For any serial streams of data that require a look-up table, select Edit → LUT from the server. Create and edit the look-up table using the LUT editor and save the result in a LUT file. See paragraph 4.1.2 on page 186 for more details on creating and editing look-up tables.

Almost without exception, all serial streams need to have a parameter database in order to use them. To create and edit a parameter database for the serial stream(s) in the system, select Edit → PDbase XXX from the server (where XXX =Ch4 (PCM), Ch8, or 1553). Click on the **Edit DBase** command on the respective parameter editor to create/edit the database. Save the resulting parameter database to a file. See paragraph 4.3 on page 190 for more information on creating and editing parameter databases.

If the test involves processing any serial data, then a project must be created. To create a project, select Edit → Project from the server. Assign the related items (hardware devices, parameter databases, LUT, etc.) to the project and save the project to a file. See paragraph 4.2 on page 187 for more information on creating projects.

Next load the project file previously created by selecting Project → Load from the server. When prompted, enter the path and file name of the project file to be opened. Depending on the size of the database(s), this may take a few seconds. Once the project is loaded, a trill will be sounded from the machine speaker. The act of loading a project will start execution of the individual application programs that setup and control the hardware devices in the system. At this point, the data acquisition process has begun. While a project is loaded, certain items cannot be changed and are thus indicated by being grayed out.

At this juncture, one may wish to archive the data. To begin recording data, click on the **Record** button in the Archive Control window (see Figure 2-19 on page 70). To stop the archiving, click the record button again. To close the project, select Project → Close from the server. When the project is closed, the archive will also close. See paragraph 2.15 on page 69 for more information of the archive function in LDPS.

To display the processed data, go to the desktop and start the client program that was installed with the LDPS system. If a project is loaded, then when the client program comes up, it will automatically load the same project and the client program can be operated as described the Part-2 of this manual. The client program can also be launched from the server by clicking on the **Start Client** command. As before, when the client program comes up, it will automatically load the same project.

After the test is complete, unload the project by clicking on Project → Close from the server. This will also stop archiving if it was still recording.

With the test complete and archiving halted, the user may now wish to play back the data (assuming any was recorded). With the project unloaded, click on System → Mode → Playback from the server. Next, select Project → Open from the server. When prompted, enter the name and path for the playback project to be loaded. The project will be loaded and the playback control window (see Figure 2-18 on page 67) will be filled with information about the project and controls for playback. From this point on the operation of the system is the same as if in live mode, except one cannot select any setups for any the devices.

To get out of playback mode, first unload the project *(Project → Close),* and then configure the server for Live operation *(System → Mode → Live).*

7 Appendix

7.1 Frame Synchronization

Definitions:

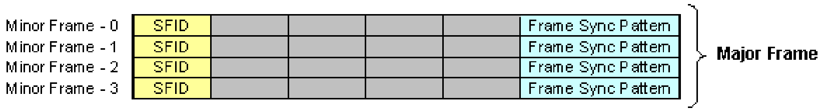
- **Frame Synchronization Pattern** – A unique binary bit pattern used to indicate the beginning of a telemetry minor frame.
- **Frame Synchronizer** – Correlator & State Machine circuitry that recognizes unique bit patterns indicating the beginning of minor frame data. The frame synchronizer typically "searches" for patterns, "checks" for the recurrence of the pattern in the same position for several frame periods, and then "locks" on the pattern.

7.2 Frame Nomenclature

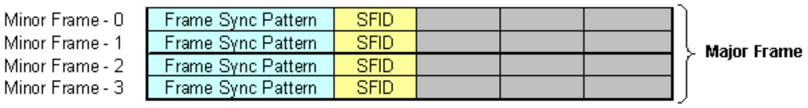
Definitions:

- **Major Frame** – An integer number of minor frames, not to exceed 256 per the IRIG-106 specification. *The LS-50 however can support up to <u>1024</u> minor frames per major frame.*
- **Minor Frame** – A fixed length block of data sub-divided into an integer number of fixed-length words. *The LS-50 can support up to <u>16,383</u> words per minor frame.*

7.3 Subframe Synchronization



Note: The location of the frame synchronization pattern (FSP) may be visualized, as in the examples here, as either being at the beginning of the minor frame, or at the end. The location of the subframe identification (SFID) word(s) is arbitrary.

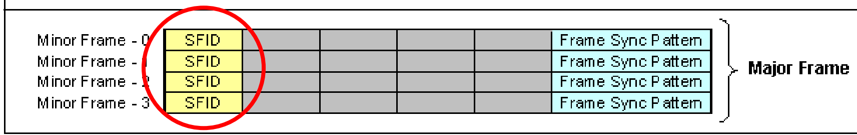


7.3.1 Subframe Identification

Definition:

- **SFID** – The most commonly used subframe synchronization method is called Subframe Identification (SFID).

In this method, the synchronization pattern occupies one or more words in each minor frame. The SFID acts as a counter. The pattern value increments or decrements to a specific value and then resets.

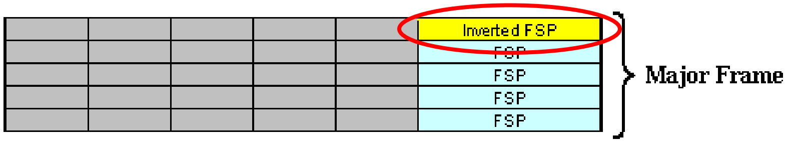


7.3.2 Frame Code Complement (FCC)

Definition:

- **FCC** – Another commonly used subframe synchronization method is called Frame Code Complement (FCC).

In this method, the complement (inverted) of the synchronization pattern is placed in the FSP location in minor frame-0. All other FSPs are not inverted. Because the complement of the frame synchronization pattern exhibits the same correlation properties as the true pattern, frame sync lock will not be compromised. Minimum sync overhead is attained using this method, although it requires longer subframe acquisition time than the SFID method.

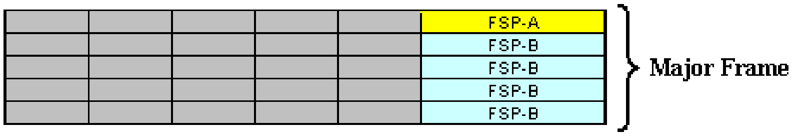


7.3.3 Unique Recycle Code (URC)

Definition:

- **URC** – A less commonly used subframe synchronization method is called Unique Recycle Code (URC).

URC is a slight variation on the FCC method. For URC, the beginning of minor frame-0 is identified by a unique synchronization pattern **NOT** related to the primary synchronization pattern.

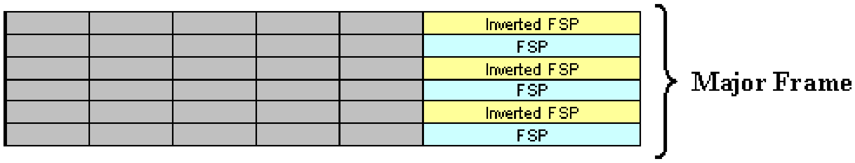


7.3.4 Frame Alternating Complement (FAC)

Definition:

- **FAC** – A less commonly used subframe synchronization method that is a variant of the FCC mode is called Frame Alternating Complement (FAC).

In this method, the frame synchronization pattern is alternated with the complement of the frame synchronization pattern.

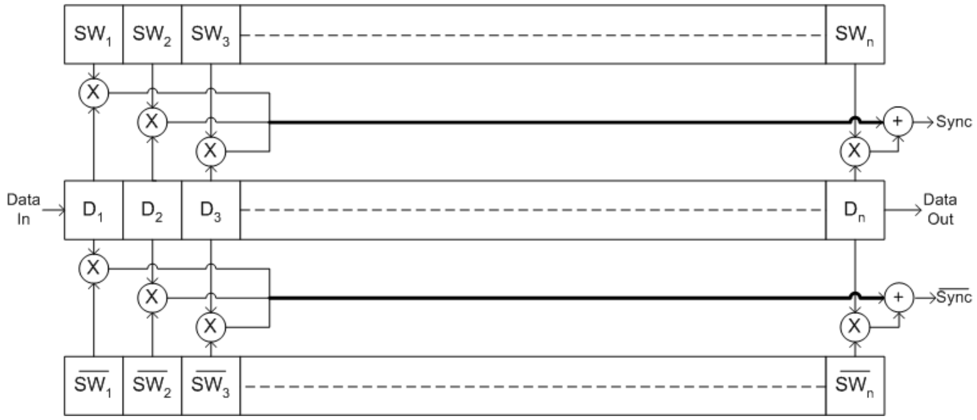


7.4 Finding the Frame Sync Pattern

Definition:

- **Correlateor** – Logic circuit (see below) used to detect the presence of a frame synchronization pattern used to identify the beginning of a minor frame.

The synchronization strategy is to pass the incoming data stream into a correlator which checks each bit of the input stream against a predefined synchronization pattern. In the correlator, the data is passed through a shift register, the contents of which are bitwise compared with the predefined pattern once each bit period. When the summation output of the correlator exceeds a preset threshold, the sync pattern is declared found. Optimal codes for the sync pattern are chosen because they have low correlation unless the code pattern is exactly aligned with the desired pattern.



7.4.1 Correlator Performance Metrics

Statistical Measures – The primary performance measure used in association with the frame synchronizer is; 1) the probability of falsely locking onto a random data pattern and believing it to be the real sync pattern, and 2) the probability of missing a valid sync pattern in the data stream due to an unacceptable number of bit errors.

- The probability of a false lock is only a function of the length of the chosen sync pattern, and NOT a function of the channel bit-error rate.
- The probability of missing a valid pattern is a function of both channel bit-error rate, and pattern length.

7.4.2 Probability of Missing a Sync Pattern

A "Geek" Technical Tidbit: The probability of missing a valid sync pattern in a noisy environment….
The probability of missing a sync pattern in a data stream is directly related to the number of bit errors encountered in the channel. If the correlator allows for a number "k" or fewer bit errors (sync tolerance value) to occur in a sync pattern of length "N" bits, then the probability "P" of missing a sync pattern in a channel with a bit-error-rate of "B" is given by:
$P = \sum_{j=k+1}^{N} \left(\frac{N!}{j!(N-j)!} \right) B^j (1-B)^{N-j}$

7.5 IRIG-106 Telemetry Classes

IRIG-106 "Factoid"

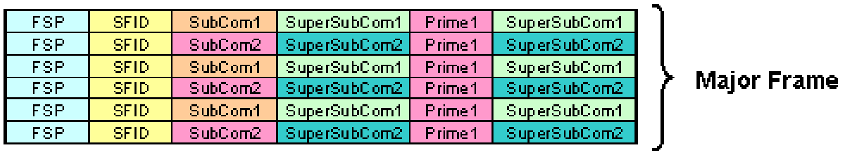
- The IRIG-106 standard defines two variations of the basic telemetry frame structure. These variations are referred to as **Class-I**, and **Class-II** and are summarized below.

Parameter	**Class-I**	**Class-II**
Bits/Minor Frame	≤8192 Bits	≤16,384 Bits
Words/Minor Frame	≤512 Words	>512 Words
Minor Frame Length	Fixed	Variable
Fragmented Words	Not Allowed	Up to 8
Format Changes	Not Allowed	Allowed
Asynchronous Formats	Not Allowed	Allowed
Bit Rates	>10 bps	>5 Mbps
Independent Subframe	Not Allowed	Allowed
SuperCom Spacing	Uniform in Minor Frame	"Anything Goes"
Data Format	Unsigned Binary, Complemented Binary	Others Allowed
Word Length	4 to 16 Bits	16 to 64 Bits

7.6 Telemetry Frame Commutation Schemes

Definitions:

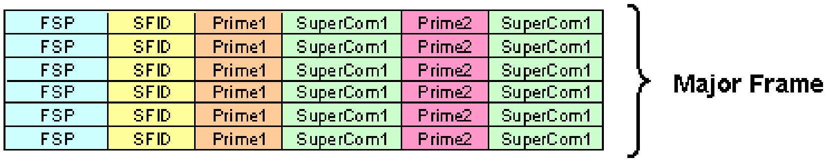
- **Subcommutated** – A parameter sent at a rate less than or equal to the minor frame rate, with each parameter appearing at a fixed subframe location.
- **Subframe** – Corresponds to a column within a major frame.
- **Super-Subcommutated** – A subframe parameter that appears more than once per minor frame.



FSP	SFID	SubCom1	SuperSubCom1	Prime1	SuperSubCom1
FSP	SFID	SubCom2	SuperSubCom2	Prime1	SuperSubCom2
FSP	SFID	SubCom1	SuperSubCom1	Prime1	SuperSubCom1
FSP	SFID	SubCom2	SuperSubCom2	Prime1	SuperSubCom2
FSP	SFID	SubCom1	SuperSubCom1	Prime1	SuperSubCom1
FSP	SFID	SubCom2	SuperSubCom2	Prime1	SuperSubCom2

Definitions:

- **Commutated** – A parameter sent once per minor frame and located in the same location in each minor frame relative to the synchronization marker. (Also called “Prime” Commutated)
- **Supercommutated** – A parameter sent at a sampling rate that is an integer factor greater then the minor frame rate, with each appearance of the parameter at a fixed location relative to the synchronization marker of the minor frame. (Note, the number of appearances of a supercommutated parameter within each minor frame is NOT fixed by the IRIG-106 standard).

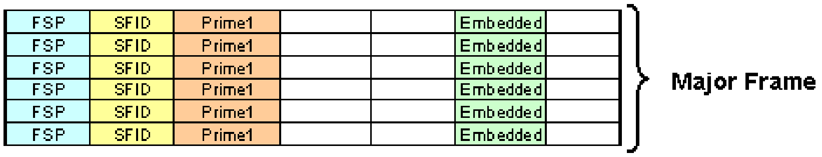


FSP	SFID	Prime1	SuperCom1	Prime2	SuperCom1
FSP	SFID	Prime1	SuperCom1	Prime2	SuperCom1
FSP	SFID	Prime1	SuperCom1	Prime2	SuperCom1
FSP	SFID	Prime1	SuperCom1	Prime2	SuperCom1
FSP	SFID	Prime1	SuperCom1	Prime2	SuperCom1
FSP	SFID	Prime1	SuperCom1	Prime2	SuperCom1

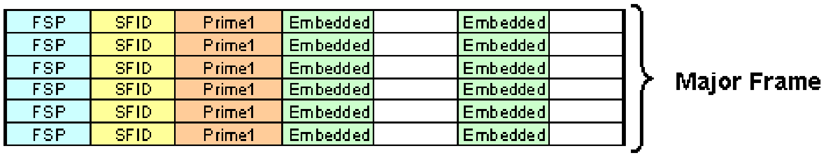
7.7 Embedded Asynchronous Telemetry Streams

Definition:

- **Embedded Asynchronous Frame** – Literally, one telemetry stream embedded within the frame structure of another, where the embedded words are at fixed locations within the primary minor frame. The LS-50 can support multiple embedded asynchronous streams using either a second hardware decommutator (LS-55-DB), and/or a software decommutator (see the LDSP user's manual for more information). The embedded words may be prime commutated, or super-commutated within the minor frame as shown below. The embedded stream is said to be "asynchronous," because there is often no definable temporal relationship between the synchronization marker of the embedded stream and the synchronization marker of the primary minor frame. More specifically, the location of the sync marker and SFID of the embedded frame are often not the same from one major frame to the next. The asynchronous nature of the embedded stream also implies that there is no bit alignment between the words of the embedded stream and the words of the primary stream. For example, bit-1 (leftmost) of the frame sync pattern of the embedded stream could be located in the middle of the second embedded word in the first major frame, and reoccur again in the second to last bit of the fifth embedded word of the next major frame, and so on…



FSP	SFID	Prime1			Embedded	
FSP	SFID	Prime1			Embedded	
FSP	SFID	Prime1			Embedded	
FSP	SFID	Prime1			Embedded	
FSP	SFID	Prime1			Embedded	
FSP	SFID	Prime1			Embedded	

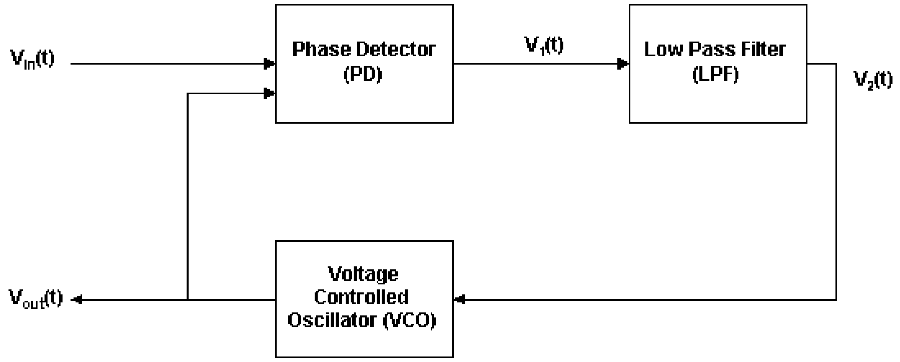


FSP	SFID	Prime1	Embedded		Embedded	
FSP	SFID	Prime1	Embedded		Embedded	
FSP	SFID	Prime1	Embedded		Embedded	
FSP	SFID	Prime1	Embedded		Embedded	
FSP	SFID	Prime1	Embedded		Embedded	
FSP	SFID	Prime1	Embedded		Embedded	

7.8 The Phase Lock Loop (PLL) – The Heart of Every Bit Sync

A "Geek" Technical Tidbit:

At the heart of any modern bit synchronizer is a phase-lock-loop (PLL) circuit. The implementation may be analog, digital, or some combination.



A phase lock loop consists of three basic components: a Phase Detector (PD), a Low Pass Filter (LPD) and a Voltage Controlled Oscillator (VCO). The phase detector produces an output signal, $V_1(t)$ that is a function of the phase difference between the input signal $V_{in}(t)$ and the VOC output signal $V_{out}(t)$. The low pass filter is used to remove the AC component of the signal coming from the phase detector output [$V_1(t)$]. The filtered output signal, $V_2(t)$ is the control signal that is used to change the frequency of the VCO output. The VCO is a special type of oscillator that produces a periodic waveform, the frequency of which may be varied about some free-running frequency, f_0, according to the value of the applied input voltage $V_2(t)$. The frequency of f_0 is the frequency of the VCO output when the applied input voltage $V_2(t)$ is zero.

When used in a bit synchronizer, the PLL configuration may be designed so that it acts as a narrowband tracking filter when the LPF is a narrowband filter. The frequency of the VCO will become that of one of the line components of the input spectrum. As such, the VCO output signal will equal the average frequency of this input signal component. Once the VCO has acquired this frequency component, the frequency of the VCO will track this signal component if it changes slightly. If the bandwidth of the LPF is wider, then the VCO can track the instantaneous frequency of the input signal as it changes. In either case, when the PPL tracks the changes in the input signal, the PPL is said to be "locked" If the applied input signal to the PLL has an initial frequency of f_0, then the PLL will acquire lock and the VCO will track the input signal frequency over some specific range, provided that the input frequency changes slowly. The loop will remain locked only over some finite range of frequency shift. This range is called the lock range. The lock range depends on the overall dc gain of the loop, including the dc gain of the LPF used. If the input signal has an initial frequency that is not equal to f_0, the loop may not lock, even though the input frequency is within the lock range. The frequency range over which the input signal will cause the loop to lock is called the capture range of the loop.

7.9 IRIG-200 Time Codes

IRIG-200 "Factoid"

- IRIG time code formats are used on military test ranges and come in several different formats for differing resolutions. Within the IRIG formats there are two different classes: Class-I (IRIG A through H frame formats), and Class-II (MIL-STD-1553 time format). The timing information within the frame can be either days, hrs, minutes and seconds (in BCD format), or in straight binary seconds format. The basic lengths and rates of the time-code frames as defined in IRIG Standard 200 are shown below:

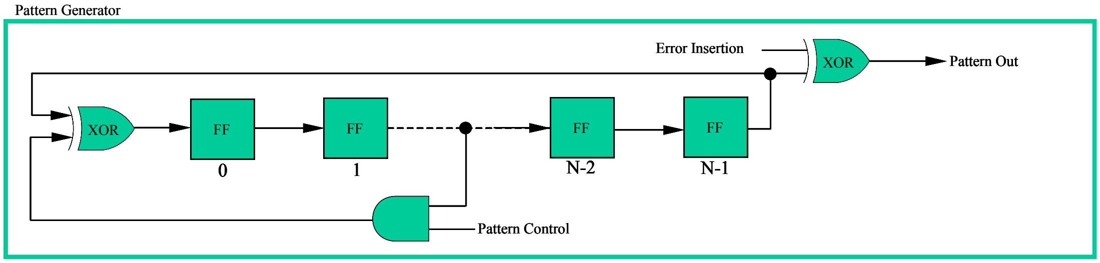
Format	**Bit Rate**	**Frame Rate**	**Bits/Frame**	**Carrier Freq.**
A	1,000 bps	10 f/sec.	78 bitss	10 KHz
B	100 bps	1 f/sec.	74 bits	1 KHz
D	1 bps	1 f/hr.	25 bits	100 Hz or 1 KHz
E	10 bps	6 f/min.	71 bits	100 Hz or 1 KHz
G	10,000 bps	100 f/sec.	74 bits	100 KHz
H	1 bps	1 f/min.	32 bits	100 Hz or 1 KHz

Note: the LS-50-P supports IRIG A,B and G formats.

7.10 The Pseudorandom Noise (PN) Sequence

A "Geek" Technical Tidbit:

The basic performance measure of any digital transmission system, of which a telemetry system is an example, is the probability that any transmitted bit will be received in error. These bit errors when they occur can be introduced in many places along the path the signal flows through. Errors introduced into the transmission are often random in nature and are strongly affected by system parameters such as *signal level*, *noise level*, and *timing jitter*.



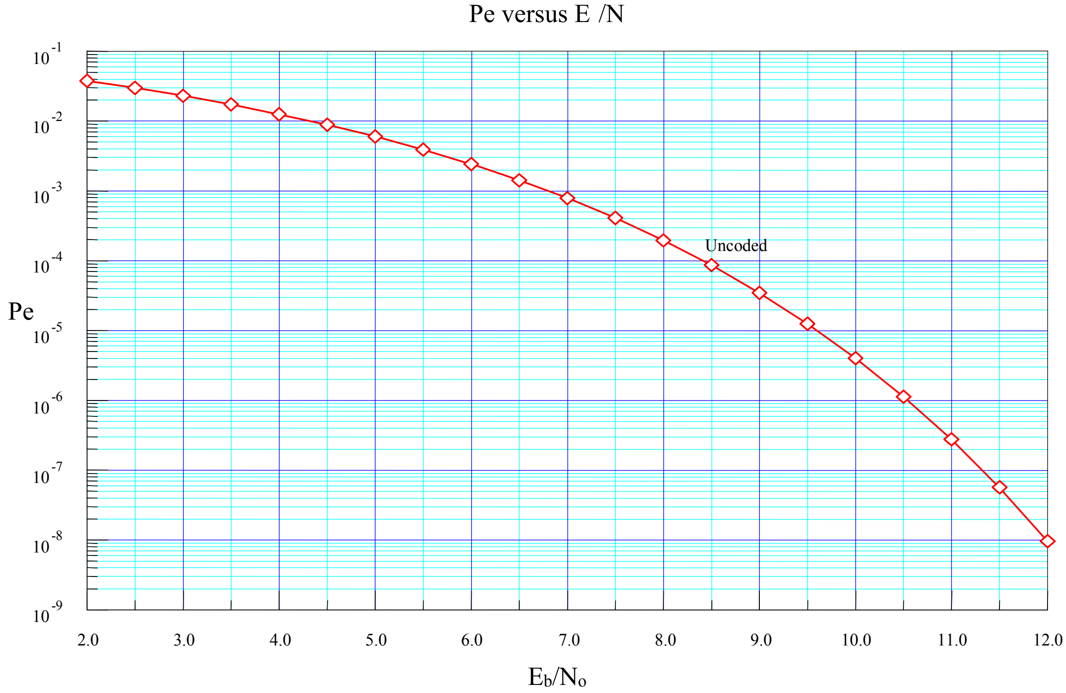
The actual digital test signal generated by the BERT employs a Pseudorandom Noise (PN) sequence to simulate traffic and to examine the transmission system for pattern-dependent tendencies or critical timing effects. An example of such a PN generator is shown above. Selecting the proper PN sequence that will be appropriate for the particular system being tested is important. Some of the key properties of the selected PN sequence that are of importance include: 1) The length of the PN Sequence. 2) The Linear Feedback Shift Register configuration used to implement the PN generator (this defines the binary run properties of the sequence). 3) Spectral line spacing of the sequence (which depends on the bit rate of the sequence). Although there are many, two PN sequence patterns have been standardized by the CCITT[22] for testing digital transmission systems. They are based on 15-stage and 23-stage Linear Feedback Shift Register configurations.

[22] CCITT Rec. 0151, Yellow Book, Vol. 4 Fascicle IV.4 Recommendation 0.151.

7.11 Bit Error Rate Probabilities

A “Geek” Technical Tidbit:

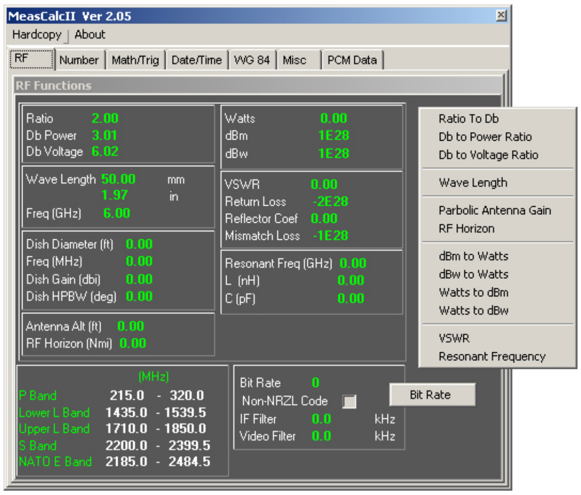
It is often helpful to visualize the BER probability function graphically by using a double log plot of P_e versus E_b/N_0. This type of plot is often referred to as a “waterfall curve.” Such a plot is shown in the figure below.



It is important to understand that this plot represents the theoretical relationship between the BER probability and E_b/N_0. If one were to characterize the actual measured BER performance for various values of E_b/N_0 for the system, a slightly different set of data points would be obtained. For the actual system, for any given value of P_e, the resulting value of E_b/N_0 will always be slightly higher in value than the theoretical. The overall performance of the system is thus compared to the best-case theoretical performance and is expressed in terms of the difference, or deviation from theory. As E_b/N_0 is a dimensionless quantity and is expressed in terms of dB, the performance of the system is often expressed as*, “so many dB from theory”.*

7.12 The Measurement Calculator

The Measurement Calculator is a virtual “Swiss Army Knife” of measurement calculations and offers a smorgasbord of handy numerical routines for a variety of different applications. Each application area has a tab containing many different parameters and functions. The user selects a particular parameter or function by right clicking and selecting the item. The specific menus for each of the tabs are shown in the figures that follow.



Ratio 2.00
Db Power 3.01
Db Voltage 6.02

Wave Length 50.00 mm
1.97 in
Freq (GHz) 6.00

Dish Diameter (ft) 0.00
Freq (MHz) 0.00
Dish Gain (dbi) 0.00
Dish HPBW (deg) 0.00

Antenna Alt (ft) 0.00
RF Horizon (Nmi) 0.00

Watts 0.00
dBm 1E28
dBw 1E28

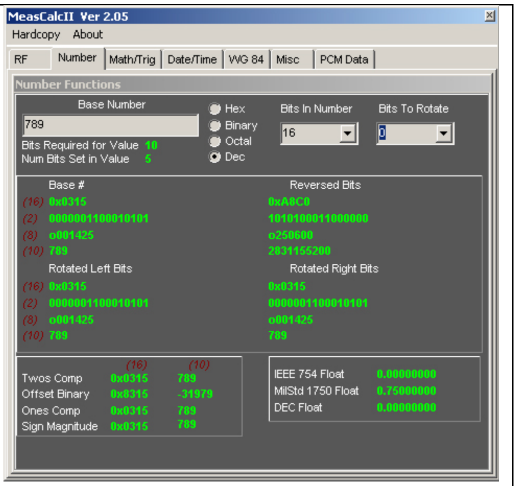
VSWR 0.00
Return Loss -2E28
Reflector Coef 0.00
Mismatch Loss -1E28

Resonant Freq (GHz) 0.00
L (nH) 0.00
C (pF) 0.00

	(MHz)
P Band	215.0 - 320.0
Lower L Band	1435.0 - 1539.5
Upper L Band	1710.0 - 1850.0
S Band	2200.0 - 2399.5
NATO E Band	2185.0 - 2484.5

Bit Rate 0
Non-NRZL Code □
IF Filter 0.0 kHz
Video Filter 0.0 kHz

Figure 7-1 The RF Tab



Base Number 789
Bits Required for Value 10
Num Bits Set in Value 5
○ Hex
○ Binary
○ Octal
● Dec
Bits In Number 16
Bits To Rotate 0

	Base #	Reversed Bits
(16)	0x0315	0xA8C0
(2)	0000001100010101	1010100011000000
(8)	o001425	o250600
(10)	789	2831155200

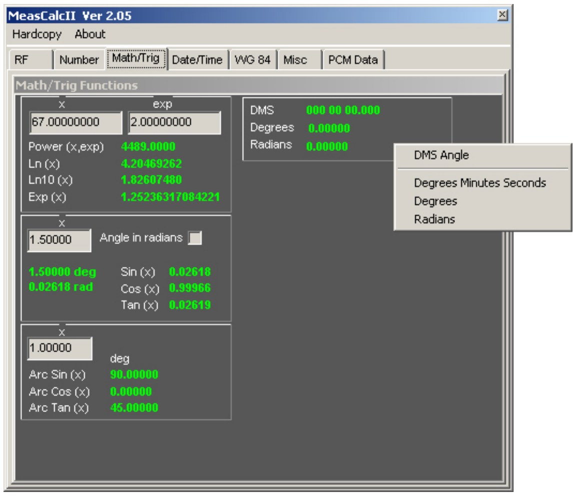
	Rotated Left Bits	Rotated Right Bits
(16)	0x0315	0x0315
(2)	0000001100010101	0000001100010101
(8)	o001425	o001425
(10)	789	789

	(16)	(10)
Twos Comp	0x0315	789
Offset Binary	0x8315	-31979
Ones Comp	0x0315	789
Sign Magnitude	0x0315	789

IEEE 754 Float 0.00000000
MilStd 1750 Float 0.75000000
DEC Float 0.00000000

Figure 7-2 The Numbers Tab

The measurement calculator has seven tabs across the top of the window that include: RF, Number, Math/Trig, Date/Time, WG 84, Misc, and PCM Data. The use of each tab is fairly self-exclamatory and is not described in detail here.



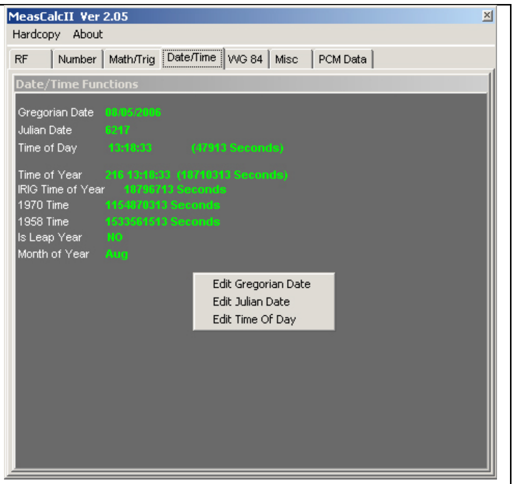
x 67.00000000
exp 2.00000000
Power (x,exp) 4489.0000
Ln (x) 4.20469262
Ln10 (x) 1.82607480
Exp (x) 1.25236317084221

x 1.50000
Angle in radians □
1.50000 deg
0.02618 rad
Sin (x) 0.02618
Cos (x) 0.99966
Tan (x) 0.02619

x 1.00000
deg
Arc Sin (x) 90.00000
Arc Cos (x) 0.00000
Arc Tan (x) 45.00000

DMS 000 00 00.000
Degrees 0.00000
Radians 0.00000

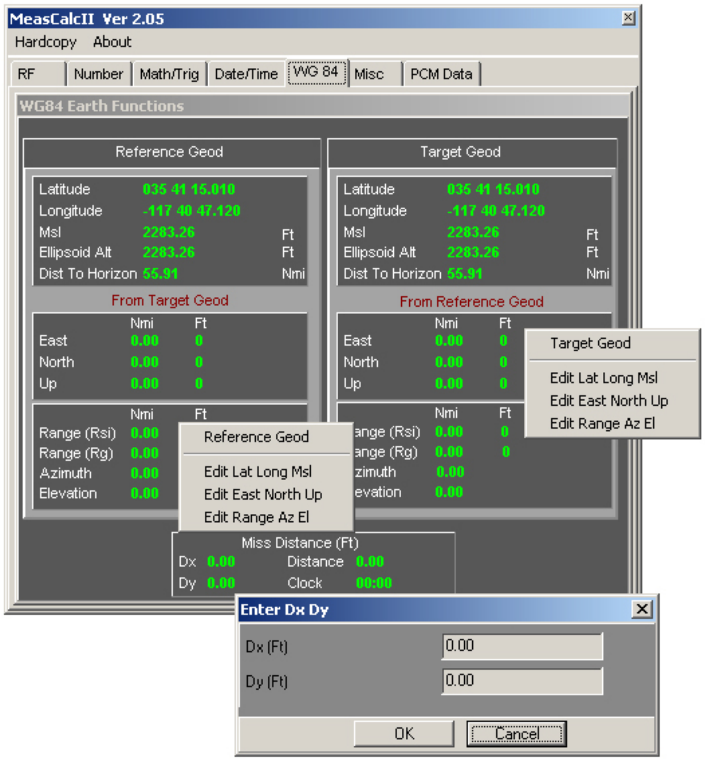
Figure 7-3 The Math/Trig Tab



Gregorian Date 08/05/2006
Julian Date 6217
Time of Day 13:18:33 (47913 Seconds)

Time of Year 216 13:18:33 (18710313 Seconds)
IRIG Time of Year 18796713 Seconds
1970 Time 1154870313 Seconds
1958 Time 1533561513 Seconds
Is Leap Year NO
Month of Year Aug

Figure 7-4 The Time/Date Tab



Reference Geod

Latitude	035 41 15.010	
Longitude	-117 40 47.120	
Msl	2283.26	Ft
Ellipsoid Alt	2283.26	Ft
Dist To Horizon	55.91	Nmi

From Target Geod

	Nmi	Ft
East	0.00	0
North	0.00	0
Up	0.00	0

	Nmi	Ft
Range (Rsl)	0.00	0
Range (Rg)	0.00	0
Azimuth	0.00	
Elevation	0.00	

Target Geod

Latitude	035 41 15.010	
Longitude	-117 40 47.120	
Msl	2283.26	Ft
Ellipsoid Alt	2283.26	Ft
Dist To Horizon	55.91	Nmi

From Reference Geod

	Nmi	Ft
East	0.00	0
North	0.00	0
Up	0.00	0

	Nmi	Ft
ange (Rsl)	0.00	0
ange (Rg)	0.00	0
zimuth	0.00	
evation	0.00	

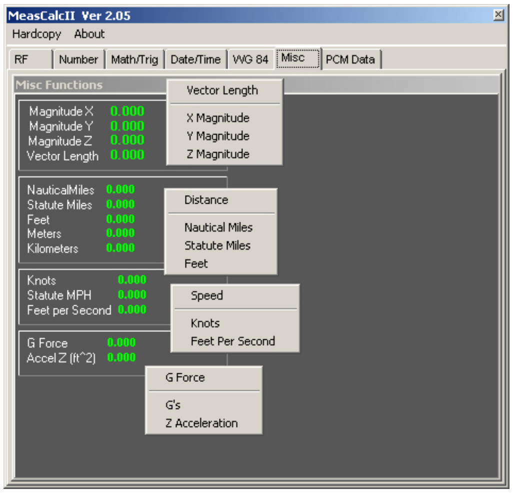
Miss Distance (Ft)

Dx	0.00	Distance	0.00
Dy	0.00	Clock	00:00

Dx (Ft) 0.00

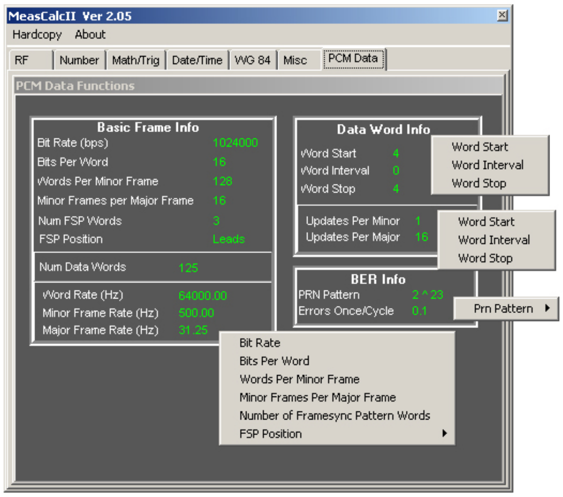
Dy (Ft) 0.00

Figure 7-5 The WG 48 Tab



Magnitude X	0.000
Magnitude Y	0.000
Magnitude Z	0.000
Vector Length	0.000
NauticalMiles	0.000
Statute Miles	0.000
Feet	0.000
Meters	0.000
Kilometers	0.000
Knots	0.000
Statute MPH	0.000
Feet per Second	0.000
G Force	0.000
Accel Z (ft^2)	0.000

Figure 7-6 The Misc Tab



Basic Frame Info

Bit Rate (bps)	1024000
Bits Per Word	16
Words Per Minor Frame	128
Minor Frames per Major Frame	16
Num FSP Words	3
FSP Position	Leads
Num Data Words	125
Word Rate (Hz)	64000.00
Minor Frame Rate (Hz)	500.00
Major Frame Rate (Hz)	31.25

Data Word Info

Word Start	4
Word Interval	0
Word Stop	4
Updates Per Minor	1
Updates Per Major	16

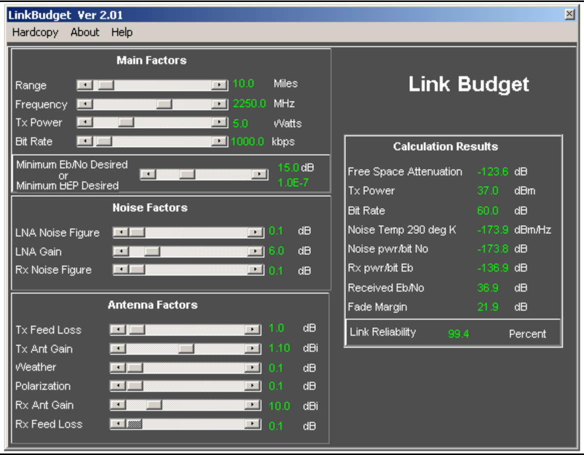
BER Info

PRN Pattern	2 ^ 23
Errors Once/Cycle	0.1

Figure 7-7 The PCM Data Tab

7.13 The Link Budget Calculator

The Link Budget Calculator is a useful tool for performing a link margin analysis for terrestrial or satellite radio links used in telemetry applications. The user enters the fundamental parameters that describe the link, including range, frequency, transmitter power, bit rate, etc., and the budget for the link is calculated as shown below right in Figure 7-8. The link budget is a simple accounting of the gains and losses that the signal experiences during its travel from the source to the destination.



Main Factors

Range	10.0	Miles
Frequency	2250.0	MHz
Tx Power	5.0	Watts
Bit Rate	1000.0	kbps
Minimum Eb/No Desired	15.0	dB
or Minimum BEP Desired	1.0E-7	

Noise Factors

LNA Noise Figure	0.1	dB
LNA Gain	6.0	dB
Rx Noise Figure	0.1	dB

Antenna Factors

Tx Feed Loss	1.0	dB
Tx Ant Gain	1.10	dBi
Weather	0.1	dB
Polarization	0.1	dB
Rx Ant Gain	10.0	dBi
Rx Feed Loss	0.1	dB

Calculation Results

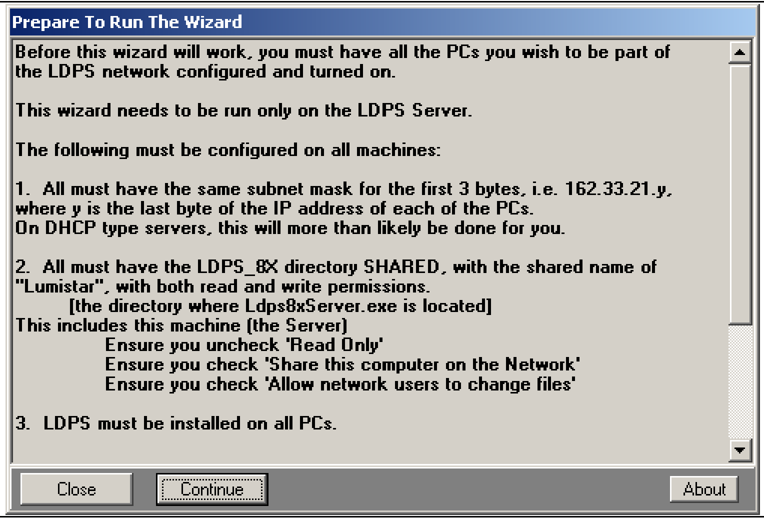
Free Space Attenuation	-123.6	dB
Tx Power	37.0	dBm
Bit Rate	60.0	dB
Noise Temp 290 deg K	-173.9	dBm/Hz
Noise pwr/bit No	-173.8	dB
Rx pwr/bit Eb	-136.9	dB
Received Eb/No	36.9	dB
Fade Margin	21.9	dB
Link Reliability	99.4	Percent

Figure 7-8 Tools → Link Budget Calculator

7.14 The Network Wizard

Network management is very important in LDPS and is necessary for the server-to-client communication mechanism. The client programs must know who to request data from, and also be able to read project files. During operation, the server broadcasts data out to the world on the net, or on the user defined subnet mask, if it was optionally set. The client programs pick up the data from the network and uses it as needed. The clients' configuration options define the network address of the server it accepts data from, as well as the address of the Backup Server, if one has been assigned.

For the mechanisms described above to function correctly, the network environment must be setup and configured correctly. The Network Wizard is a helpful took that aids the user in the network setup process, and walks the user through the steps necessary. The first window that the Network Wizard displays is shown in Figure 7-9 below.



Before this wizard will work, you must have all the PCs you wish to be part of the LDPS network configured and turned on.

This wizard needs to be run only on the LDPS Server.

The following must be configured on all machines:

1. All must have the same subnet mask for the first 3 bytes, i.e. 162.33.21.y, where y is the last byte of the IP address of each of the PCs.
On DHCP type servers, this will more than likely be done for you.

2. All must have the LDPS_8X directory SHARED, with the shared name of "Lumistar", with both read and write permissions.
[the directory where Ldps8xServer.exe is located]
This includes this machine (the Server)
Ensure you uncheck 'Read Only'
Ensure you check 'Share this computer on the Network'
Ensure you check 'Allow network users to change files'

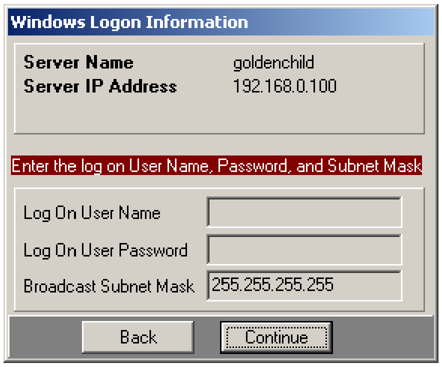
3. LDPS must be installed on all PCs.

Figure 7-9 Network Wizards' Help Screen

The complete text of the help window is shown below:

```
Before this wizard will work, you must have all the PCs you wish to be part of
the LDPS network configured and turned on.

This wizard needs to be run only on the LDPS Server.

The following must be configured on all machines:

1.  All must have the same subnet mask for the first 3 bytes, i.e. 162.33.21.y,
where y is the last byte of the IP address of each of the PCs.
On DHCP type servers, this will more than likely be done for you.

2.  All must have the LDPS_8X directory SHARED, with the shared name of
"Lumistar", with both read and write permissions.
         [the directory where Ldps8xServer.exe is located]
This includes this machine (the Server)
               Ensure you uncheck 'Read Only'
               Ensure you check 'Share this computer on the Network'
               Ensure you check 'Allow network users to change files'

3.  LDPS must be installed on all PCs.

4.  All PCs must have the same log on user name and password.

5.  All firewalls turned off (i.e. McAffey, Symantic, Windows, etc)
If you know what you are doing, you can set the firewalls up to work if you wish.

When all of the above are met, press 'Continue' to begin, or 'Close' to end the
wizard.
```

Clicking the **Continue** button in Figure 7-9 invokes the Windows Login Information dialog box shown below. Enter the log on name and password, and make any changes to the subnet mask required (contact your network administrator for assistance).



Windows Logon Information

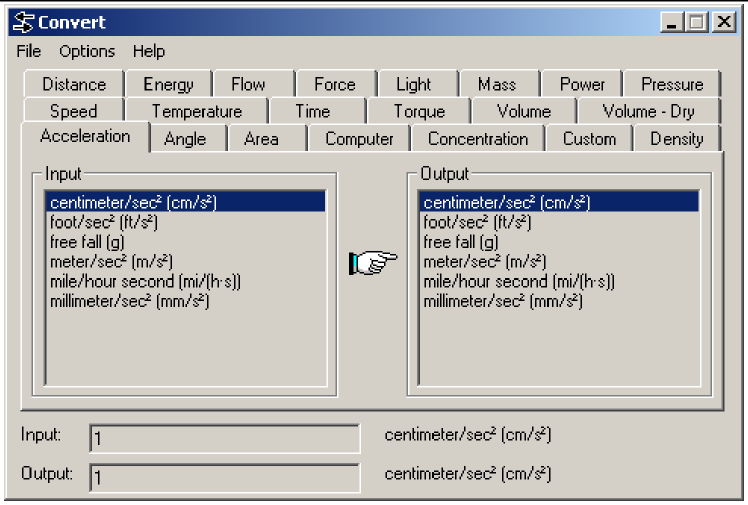
Server Name goldenchild
Server IP Address 192.168.0.100

Enter the log on User Name, Password, and Subnet Mask

Log On User Name
Log On User Password
Broadcast Subnet Mask 255.255.255.255

7.15 The Measurement Converter

The Measurement Converter is another virtual "Swiss Army Knife" of conversion functions for a large variety of physical parameters. Each physical parameter has a tab containing many different units of measure that the user converts to and from. The user selects a particular input and output parameter, and the units of measure for each and then enters the numeric value for the input. The output is automatically calculated as the input value is entered. The specific menus for each of the tabs are shown in the figures that follow.



Convert
File Options Help
Distance Energy Flow Force Light Mass Power Pressure
Speed Temperature Time Torque Volume Volume - Dry
Acceleration Angle Area Computer Concentration Custom Density
Input
Output

- centimeter/sec² (cm/s²)
- foot/sec² (ft/s²)
- free fall (g)
- meter/sec² (m/s²)
- mile/hour second (mi/(h·s))
- millimeter/sec² (mm/s²)

- centimeter/sec² (cm/s²)
- foot/sec² (ft/s²)
- free fall (g)
- meter/sec² (m/s²)
- mile/hour second (mi/(h·s))
- millimeter/sec² (mm/s²)

Input: 1 centimeter/sec² (cm/s²)

Output: 1 centimeter/sec² (cm/s²)

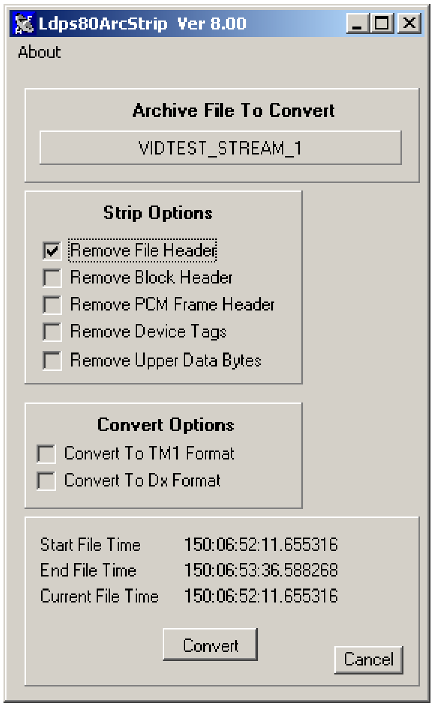
Figure 7-10 Tools → Measurement Converter

7.16 The LDPS Archive Stripper Utility

If the user wishes to employ their own tools for data reduction, it will be necessary to extract data from the LDPS archive files to facilitate this processing. The archive stripper utility shown in Figure 7-11 below should prove useful in this endeavor. The archive file format described in paragraph 2.15.1 on page 72 has four main sections. These include:

- File Header – Contains information about how to read the file.
- Block Header – Contains block time stamp, run number, time source, and a spare byte.
- Frame Header – The decommutator prefixes PCM data with the frame timestamp.
- Device Tags – Data from the devices (device status tags) are written after the PCM data.

The user may elect to remove any or all of the main sections described above. The archive file may also optionally be converted into a TM1[23] archive format.



Ldps80ArcStrip Ver 8.00

About

Archive File To Convert

VIDTEST_STREAM_1

Strip Options

☑ Remove File Header
☐ Remove Block Header
☐ Remove PCM Frame Header
☐ Remove Device Tags
☐ Remove Upper Data Bytes

Convert Options

☐ Convert To TM1 Format
☐ Convert To Dx Format

Start File Time 150:06:52:11.655316
End File Time 150:06:53:36.588268
Current File Time 150:06:52:11.655316

Figure 7-11 Tools → LDPS Archive Stripper

To operate the program, start by clicking on the **Archive File To Convert** bar. Select the archive file to convert. Note: the file must be in the LDPS 8x archive format. Select the

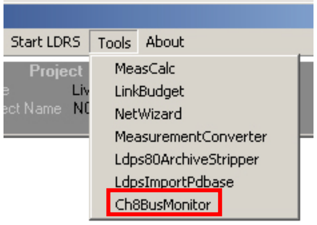
[23] (Tables Manager 1) A multidimensional analysis program for DOS and Windows from Applix, Inc., Westborough, MA (www.applix.com).

strip options check boxes for those elements that are to be removed from the archive file. Finish by clicking the **Convert** button. The process may be aborted by pressing the **Cancel** button. When the conversion is finished, a trill sound will emit from the PC speaker and the Convert button will no longer appear depressed. The converted file is saved in the directory the stripper utility resides in. The converted file will have the same name as the original file with a "BIN" extension. If one elects to keep the file header information in the converted file, then the file header information will be followed by a NULL character before the PCM data begins. (Look for a <CR>, Line Feed, NULL sequence).

7.17 LDPS Import Database

The LDPS parameter database import program is described in detail in paragraph 1.3.8 on page 21. To invoke it, from the server, select Tools, and then LdpsImportPdbase (*Tools → LdpsImportPdbase*). This tool can also be invoked in a stand-alone fashion from the ../User Tools/ subdirectory. This program will not import every type of format likely to be encountered, but it will handle most of them. If it will not support a particular master the user can always write their own custom application, using the supplied LdpsPdbaseConvert.dll.

7.18 Chapter 8 Bus Monitor



The Chapter 8 Bus Monitor is invoked from the tools menu on the LDPS server as shown right. The Chapter 8 Bus Monitor program is an application that monitors IRIG-106, Chapter 8[24] data, and a few statistics. The actual raw PCM data is decommutated by the server program (see paragraph 3.2 on page 79). The embedded Chapter 8 data is then further processed by the Chapter 8 software decommutator DLL running on the Client. The resulting Chapter 8 data is placed in shared memory, where the bus monitor application can access it. Note: the bus monitor application is useless without the LDPS client running a Chapter 8 decommutator as one of the streams.

To invoke the Chapter 8 Bus Monitor display, click on the Tools menu in the server and select the **Ch8BusMonitor** command (*Tools → Ch8BusMonitor*). The resulting window is shown in Figure 7-12 on page 261. There are five main sections of the GUI. At the top left are two commands: Debug, and About. Clicking the **Debug** command invokes the Chapter 8 Software Decommutator Logging Options Window shown in Figure 7-13 on page 263. The **About** command allows the user to view the software version number and the error log for this Chapter 8 Bus Monitor.

Below the two commands, in the upper left corner of the window is the Chapter 8 software decommutator information. This indicates if the soft decom is alive or not, and

[24] Defines MIL-STD-1553 data that has been embedded within a standard PCM telemetry stream defined by Chapter 4 of the IRIG-106 document.

which streams are available for monitoring. If there is more than one Chapter 8 stream loaded on the server, the user may select which stream to monitor via the click wheel.

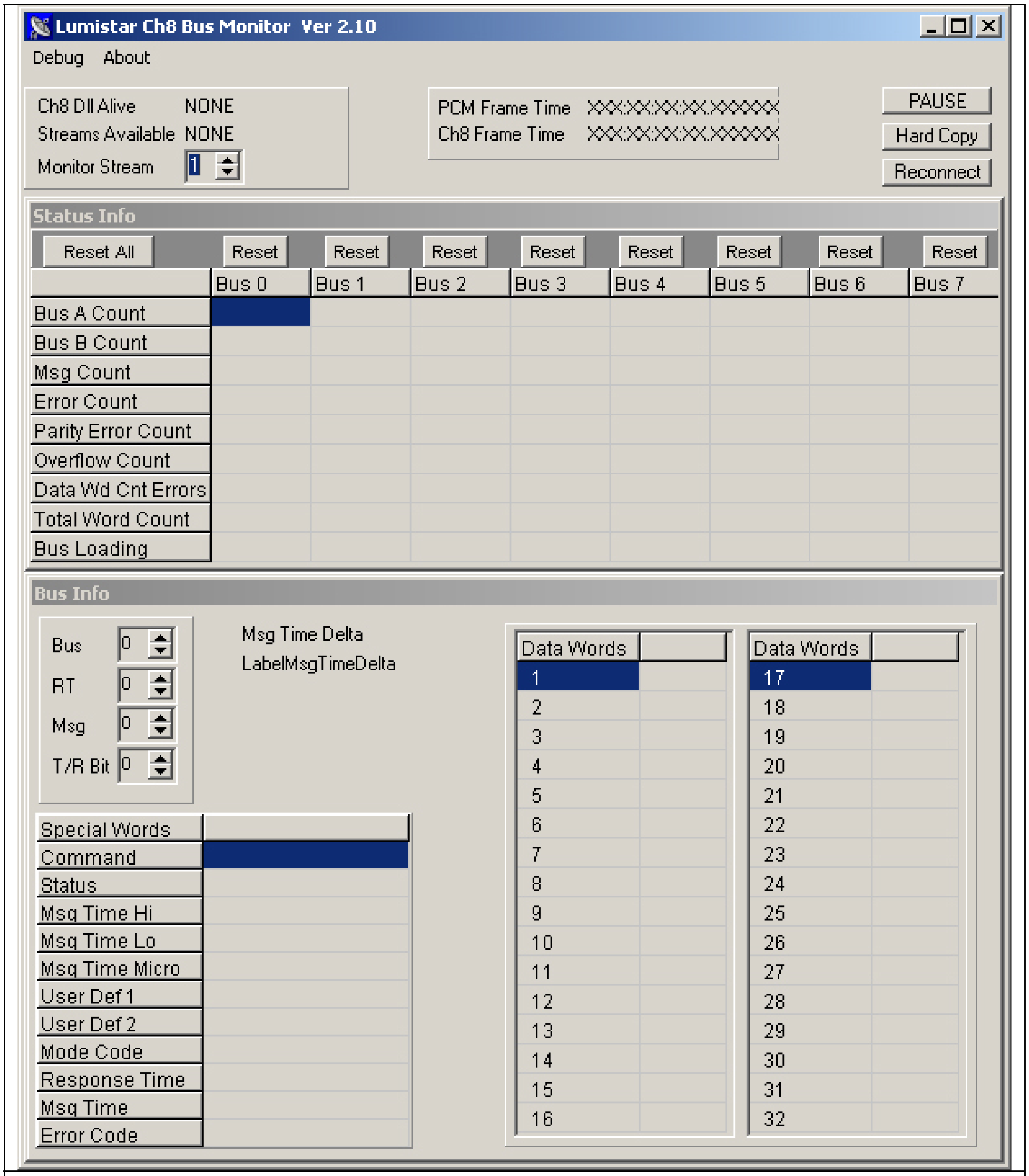
At the top center of the window is a box displaying the **PCM Frame Time** (the time stamp of the frame of data from the decommutator) and the embedded **Chapter 8 Frame Time** (i.e. Chapter 8 words 2, 3, and 4).

At the top right of the window are three controls: Pause, Hard Copy, and Reconnect. Clicking the **Pause** button halts the updating (at a 10 Hertz rate) of numerical data on the display. Clicking the **Hard Copy** button takes a snapshot of the application window and saves the image (Windows BMP format) in the root directory where the bus monitor application is located. If the Chapter 8 bus monitor is started before the server and primary decommutator (LS-50), there is a chance that the shared memory connection will not be active. In this scenario, click the **Reconnect** button to establish the link between the software decommutator and the Chapter 8 bus monitor.

The bus status information occupies the upper half of the display below the controls mentioned previously. This area contains statistical counters for each bus, as follows:

- Bus A Count – For type words 4 thru 15 from the command word, counts the number of words stored since the reset of the decoder on bus A.
- Bus B Count – – For type words 4 thru 15 from the command word, counts the number of words stored since the reset of the decoder on bus B.
- Msg Count – Counts the number of times a new command word was stored.
- Error Count – Counts the number of times a type word 8 and 12 from the command word were stored (error bus A and error bus B).
- Parity Error Count – If the software decommutator parity checking mode is not set to DON’T CARE, then parity errors are counted. The default parity checking mode is set to HOPE, meaning just count the errors, but don’t do anything about them.
- Overflow Count – Counts the number of times a type word 0 from the command word was received.
- Total Word Count – Counts all type words except type 1 from the command word (fill data). This is used for the calculation of bus loading.
- Bus Loading – The percentage of the maximum theoretical usage of the bus.

Individual statistics for each bus may be independently reset by clicking the **Reset** button above the respective column (0 through 7). Click the **Reset All** button to reset all statistics to zero.



Ch8 Dll Alive NONE
Streams Available NONE
Monitor Stream 1

PCM Frame Time XX:XX:XX:XX.XXXXXX
Ch8 Frame Time XX:XX:XX:XX.XXXXXX

	Bus 0	Bus 1	Bus 2	Bus 3	Bus 4	Bus 5	Bus 6	Bus 7
Bus A Count								
Bus B Count								
Msg Count								
Error Count								
Parity Error Count								
Overflow Count								
Data Wd Cnt Errors								
Total Word Count								
Bus Loading								

Bus 0
RT 0
Msg 0
T/R Bit 0

Msg Time Delta
LabelMsgTimeDelta

Special Words	
Command	
Status	
Msg Time Hi	
Msg Time Lo	
Msg Time Micro	
User Def 1	
User Def 2	
Mode Code	
Response Time	
Msg Time	
Error Code	

Data Words	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	

Data Words	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	

Figure 7-12 Chapter 8 Bus Monitor Window

The bus information section occupies the lower portion the Chapter 8 bus monitor window shown in the figure above. The bus info area displays the data values for the selected bus, R/T number, message number, and T/R bit position. There are four main areas of the bus info section that include:

- Word Selection – It is impossible to display all 600,000+ words on a single display. Using four click-wheels to select the desired Bus, R/T number, Message number, and Transmit bit position, the user may select the area of interest in the stream. The other sections will display data specific to these selections.

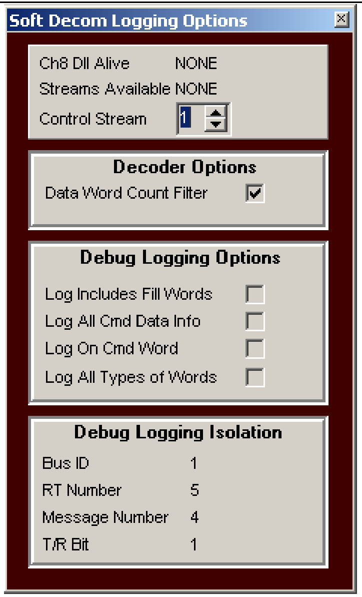
- Message Time Delta – To the right of the word selector is displayed the time difference between the embedded Chapter 8 frame time and the time the last message was updated. The time delta can indicate if the data is stale.
- Special Words – There are forty-one (41) words in each array of Chapter 8 data. The array consisting of ([bus][T/R bit][msg][RT num][word]), with lengths of [8][2][32][32][41]. Of those forty-one words, thirty-two (32) are data words. The other words are defined as special words as follows:
  1. Command Word
  2. Status Word
  3. Message Time Hi
  4. Message Time Lo
  5. Message Time Micro
  6. User Def 1
  7. User Def 2
  8. Mode Code
  9. Response Time

  These values are displayed in hex. For convenience, the three message time words (3,4, & 5) are put together to display the time of day format.
- Data Words – There are thirty-two (32) data words in the Chapter 8 array selected. The values are displayed in hex.

7.18.1 Chapter 8 Bus Monitor Debug Options

As an aid to analysis and troubleshooting, the Chapter 8 Bus Monitor includes a debugging feature. With it, the user may log information in an error log (located at System\ErLogs\Ch8Dll.log). The debug command features should only be used in playback mode due to certain peculiarities in logging ASCII data and the high stream data rates involved. Click the **Debug** command in the main menu to invoke the software decommutator logging options window shown in Figure 7-13 below right.

The soft decom logging options window has four main sections including: Stream Control, Decoder Options, Debug Logging Options, and Debug Logging Isolation. Each section is discussed in more detail in the numbered paragraphs that follow.



Soft Decom Logging Options

Ch8 Dll Alive NONE
Streams Available NONE
Control Stream 1

Decoder Options
Data Word Count Filter ☑

Debug Logging Options
Log Includes Fill Words □
Log All Cmd Data Info □
Log On Cmd Word □
Log All Types of Words □

Debug Logging Isolation
Bus ID 1
RT Number 5
Message Number 4
T/R Bit 1

Figure 7-13 Chapter 8 Software Decommutator Logging Options Window

7.18.1.1 Stream Control

This section shows the status of the Chapter 8 DLL (NONE, or ACTIVE), and indicates is Chapter 8 streams are available (NONE, or ACTIVE). If multiple streams are available, the user may select the desired stream via the click wheel.

7.18.1.2 Decoder Options

It has been noticed that some streams have a problem in that the number of data words following a command word are not correct. In the strictest sense, this is an error and the error would be counted in the decoder. However, if the **Data Word Count Filter** is not selected, then the data words are added to the array anyway and the error is not counted. If the filter is selected, then the data words are not added to the array and the error counter is incremented.

7.18.1.3 Debug Logging Options

In this section, the user may indicate with more specificity, the type of data that is to be recorded in the log. By checking **Log Includes Fill Words**, even fill words will be logged with the data, provided the **Log All Types of Words** check box is also selected. If the **Log All Cmd Data Info** check box is selected, then each 24-bit value will be logged as it arrives and is decoded. The information recorded includes: Cmd Word (upper 8 bits), Value (lower 16 bits), Bus ID (extracted from the Cmd Word), and Typw Word (extracted from the Cmd Word). If the **Log On Cmd Word** check box is selected, then the command word will be broken out and logged. The information recorded includes: Bus, RT, Msg, T/R bit, Cmd, Last5, Words To Follow, and Mode Code. By selecting **Log All Types of Words**, any type word that comes along will be logged. The information recorded includes: Bus, RT, Msg, T/R bit, Word Type (including Overflow), Value of the word, Data Words to Follow, and Data Word Count (number of words counted so far for this cmd word).

7.18.1.4 Debug Logging Isolation

This section acts as a filter for the logging options described in the previous paragraph. To limit the logging to the selected Bus, RT, Msg, and T/R bit, right click the mouse cursor and select/configure the desired element. To log all traffic for a given element, enter a value of “-1” for that element. A “-1” for the T/R Bit will record both bits.

7.18.2 Notes about Chapter 8 Bus Monitor Usage

The statistical counters can get quite large. To clear them and start again, on the Client (see part-2 of this manual), click Reset in the soft decom area. The user will be prompted to enter a value to fill the data with. The data will be filled with this value and the statistical counters will reset to zero. The default value to start the Chapter 8 arrays with is zero. To see which data words are being used, fill their value with some other number, and then look at the words for the desired address. If the value remains at the initial value, then it is not being updated in the stream. This is an easy way to determine the number of data words being used for a particular command word.

The User Def words are really not part of the 1553 data per say, but the IRIG specification dictates a bus ID be used when transmitting them. These words are counted as bus traffic by the software decommutator, and therefore the bus loading calculation will not be accurate for the bus defined in the user def words. For example, in the F18 program, the User Def 2 word is used for embedded audio, with a bus ID of 7. If this scenario becomes an issue, the software decommutator can be modified not to count the user def words as part of the bus loading calculation.

The bus loading percentage is calculated at a 1 hertz rate. The calculation is thus:

- totalbits = AryTotalWordCount[busnum] * 20.0;
- maxbits = dtime * 1,024,000.0; //max theoretical bits per second
- loading = totalbits/maxbits;

There seems to be no consensus on what to do when a parity error is detected. In LDPS, the default mode is to just count the errors and do nothing else about them (proceed as if nothing was wrong). Parity checking looks for odd parity.