CPhyGenCtl User's Manual

Table of Contents

Contacting **The *Moving Pixel* Company**

Phone	+1.503.626.9663	US Pacific Time Zone
Fax	+1.503.626.9653	US Pacific Time Zone
Address	**The *Moving Pixel* Company** 4905 SW Griffith Drive, Suite 106 Beaverton, Oregon 97005 USA	
Email	information@movingpixel.com	
Web site	http://www.movingpixel.com	

Documentation

1 Overview

The CPhyGenCtl application is the controlling software for the CPhy Generator (P339) made by **The *Moving Pixel* Company** (TMPC). Using this instrument, the user can generate CSI protocol and pattern stimulus on a MIPI CPhy bus for receiver testing. This document describes the use and operation of CPhyGenCtl and the corresponding behavior of the CPhy Generator.

For those familiar with earlier TMPC MIPI products, the CPhyGen instrument is similar in functionality to the P331/P332/P338 DPhy probe family. Unlike prior DPhy solutions that required use of a general-purpose pattern generator (PG3A), the CPhy Generator is a stand-alone instrument containing an internal pattern generator. Thus, it is the only custom hardware required for CPhy generation.

The CPhy Generator is connected via USB to a host computer that runs the CPhyGenCtl software. It has the following capabilities:

- Connects via either USB2 or USB3, allowing for fast program download times.
- Supports one to four CPhy lanes, supporting frequencies up to 2.6 Gsym/s
- Has SMA outputs for each wire of each lane
- Contains internal pattern generator with 2 GB of program memory
- Provides up to 15 ns of integer symbol lane skew, with fractional symbol lane skew below 1.5 Gsym/s.
- Provides real-time, per-lane, high and low voltage, fine-adjustments for both LP and HS signals.
- Supports fine adjustment of CPhy bus timing
- Provides 4 KB receive buffer for DUT LP response capture
- Supports lane 0 LP contention detection
- Implements arbitrary logical-to-physical lane output mapping
- Provides configurable Trig Out signal, which can be asserted via MIPI stream and/or software control.

The CPhyGenCtl application is a Windows application that embodies knowledge of MIPI DSI, CSI and C-PHY protocols to build programs for the CPhy Generator. Based on its predecessor software, PGRemoteForP338, it provides the following functions:

- Comprehensive video support:
  - Support for 3D stereoscopic frame construction (DSI 1.2).
  - Purchase option for encoding and sending of DSC video frames (DSI 1.2).
  - Common source input image file formats (jpg, png, tiff, gif), used for both video-mode and Write Memory commands.
  - Automatic resizing of input images (in software) to fit display or camera dimensions.
  - Basic test pattern generation via naming syntax (syntax dialog provided).
  - Convenient image preview function in GUI.
  - Automated CSI/DSI video-mode frame generation based on user frame timing.

  - Automatic partitioning of single Write Memory command into multiple Write Memory command sequence.
  - Video-mode frames can be added to macros.
  - Video-mode frames can be constructed with a single-bit error at a given line and bit position.
- Generic File command support:
  - Uses text file description to describe mixed low-level LP/HS transitions and packet definition.
  - Allows nearly-arbitrary data lane signal generation for conformance testing.
  - Provides higher-level embedded commands for easy command definition, including HS burst entry and HS burst exit sequences. Also, automatic ECC and CRC generation.
  - Supports nested files for reuse of common definitions.
- Support for low-level CPhy testing:
  - Low-level test HS burst sequences using user-defined or PRBS data.
  - Comprehensive CPhy protocol configuration for setting preamble, postamble, and sync sequences for each lane. Also, provides configuration for user bus timings, e.g. HSPreare, HSExit, etc.
  - New flexible bus timing specification in component units of ns, UI, and TLPX, allowing for frequency agile configurations.
- Powerful and easy-to-use GUI controls for command manipulation:
  - Simple definition, naming, and sending of commands, including video-mode commands.
  - Push-button interface for assigning and organizing commands so they are available for single-click sending.
  - Macro definition for building and complex command sequences.
  - Provides named frame timing configurations and CPhy timing configurations.
  - Provides dialogs to reorder/delete/sort named command, frame timing, and CPhy timing drop-down lists.
- Automation Support:
  - Script command allows text file usage for: configuration, command/macro/program definition and output.
  - Remote-control capability via .NET DLL.

2 Setup & Installation

2.1 Installing CPhyGenCtl

To install CPhyGenCtl, simply execute the setup.exe file on the installation CD and step through the setup windows. As part of the installation, a USB driver is installed called CyUSB3.sys. This driver is required for the software to recognize the CPhy Generator instrument when connected to USB.

Periodically, new versions of CPhyGenCtl are available on the Moving Pixel Company web-site. These upgrade versions do NOT install the CyUSB3.sys driver and assume the install machine already has the driver installed. Thus, new machines always must first have the initial setup installation run before any upgrades. Note that previous versions do not need to be uninstalled before running installation of an upgrade version of CPhyGenCtl.

The CPhyGenCtl application is installed by default in the **c:\Program Files\TMPC\CPhyGenCtl** directory. The following shortcuts are provided in the Start Menu, under CPhyGenCtl

- **CPhyGenCtl.exe**: a link to the application executable
- **CPhyGenCtlUsersManual_x_x.pdf**: a link to this manual
- **Uninstall CPhyGenCtl**: a link to an uninstall script

The CPhyGenCtl application runs under the WinXP, Win7, and Win8 operating systems and uses the Microsoft .NET Framework 3.5 platform.

Once the software has been installed for the first time, you will need to copy the license file "TMPCLicense.txt" obtained from The Moving Pixel Company to the application directory. The license file enables CPhy operation (and perhaps other options in the future) based on the serial number of your CPhy Generator. Note that the software may be installed on and the license file copied to multiple host machines.

If a license file (or the correct license string in the license file) is not present, the software can still run in "offline" mode. This mode allows the user to interact with the GUI, define and save commands, and develop RPC applications without connecting to real hardware.

2.2 Connecting the CPhy Generator

The CPhy Generator is easily set up.

- Connect the 24V power supply adaptor to the instrument.
- Connect a USB cable from the host machine to the instrument. Note: a USB3 cable is required to connect to the CPhy Generator. However, either a USB3 or USB2 port on the host computer can be used for connection (although USB2

operates at reduced speed). A USB3 port is distinguished from USB2 by its blue color.

- Connect your DUT to CPhy lane outputs via SMA cables
- Power on the instrument

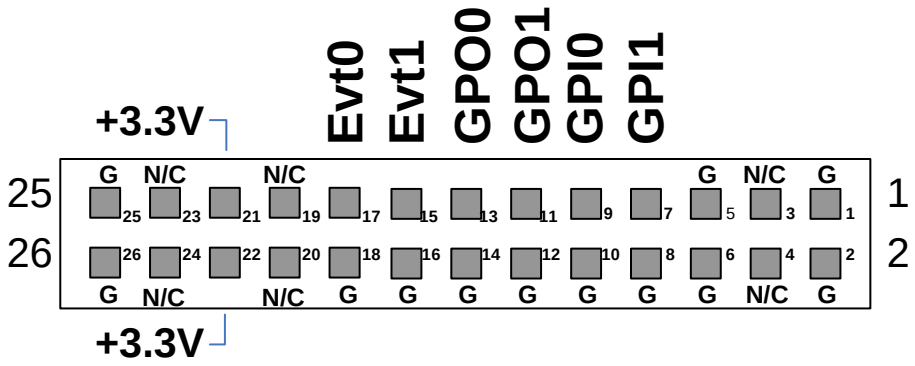

Figure 1— P339 CPhy Generator Back-Panel Pin-out

For a description of the P339 CPhy Generator hardware, please see the P339 datasheet. For reference in this document, the back-panel pin-out is shown in Figure 1. Currently, the pins used on the back-panel connector and their functions are:

- **Evt1** – represents an OR of all the six lane 0 contention flags
- **GPI0** – used as external event in

Remaining signals on the connector are unused.

After power-on, assuming the CyUSB3 driver is installed, the instrument will appear in the device manager of the host machine as "P339 CPhy Generator". At this point launching CPhyGenCtl should show the hardware serial number in its Connect dialog, ready for use.

2.3 Quick-Start Summary

The following summarizes the overall procedure for installing and configuring the TMPC software and hardware for CPhy testing:

1) Install CPhyGenCtl (section 2.1)
2) Set up, connect, and power on the CPhy Generator (section 2.2)
3) Launch the CPhyGenCtl application and connect to the CPhy Generator instrument (section 5.3)
4) Configure instrument operational parameters (section 5.4).
5) Configure video timing parameters for your device (section 5.7.3)
6) Configure CPhy bus timing and protocol definitions.(section 5.5)
7) Define commands and assign them to buttons, if desired (section 5.6.1).

Once these steps are completed, the system is ready to start testing.

3 Common Command Data and Element Types

Various data and element types are used in the CPhyGenCtl application, whether as parameter arguments to GUI commands, script commands, or component commands of a text file used in the File Command. This section describes common data and element types and how to specify them.

3.1 Bytes

Bytes are 8-bit integer values ranging from 0 to 255 and can be specified in decimal or hexadecimal. Unless otherwise specified for a particular command, the default radix for command byte arguments is decimal. If a command indicates it uses "hex byte" arguments, then the default radix is hexadecimal.

To explicitly define the radix of a data byte, append either a 'd' for decimal or 'h' for hexadecimal. For example, the following list describes a 4 byte sequence from 16-19 inclusive:

16 11h 18d 13h

3.2 Integers

Integers are 32-bit signed integer values and can be specified in decimal or hexadecimal (in which case the value is always non-negative). Unless otherwise specified, the default radix for integer arguments is decimal.

To explicitly define the radix of an integer, append either a 'd' for decimal or 'h' for hexadecimal. For example, the following list describes a 4 integer sequence from 256-259 inclusive:

256 101h 257d 103h

3.3 Byte or Integer Lists

Some commands allow parameters to be byte or integer lists. These lists are specified by one or more byte or integer values respectively, separated by spaces (see previous two sections for examples).

3.4 CPhy HS Symbols

CPhy HS symbols ("symbols" in this document) are defined in the CPhy specification and are represented by integers ranging from 0-4 inclusive. In addition, in CPhyGenCtl, a non-standard symbol value of 7 is also supported, representing an error symbol transition. An error symbol transition (really, non-transition)

CPhy symbols consist of values from 0-4, as according to the CPhy specification, and also a non-standard value of 7, which can be used to force an illegal duplicate state on the bus. A symbol value of 7 represents a non-transition ("stay in the same state") for the next symbol period.

As specified in the CPhy specification, the symbol-to-state transition mapping in the CPhy Generator is as follows:

Table 1 – CPhy HS Symbols

Symbol Value	State Transition
0	Rotate CCW, same polarity
1	Rotate CCW, opposite polarity
2	Rotate CW, same polarity
3	Rotate CW, opposite polarity
4	Same phase, opposite polarity
7	Same phase, same polarity (invalid state transition)

3.5 CPhy HS States

CPhy HS states ("states" in this document) represent high-speed voltage levels for the ABC wires of a trio and can be specified in CPhyGenCtl as integer values ranging from 0-7. In CPhy, HS states correspond to legal voltage combinations on the three trio wires, where "legal" is defined as a proper permutation of high, medium, and low voltages. As such, there are 6 legal states: LMH, LHM, MHL, MLH, HLM, and HML, using the following definitions:

- H represents a "high-level" voltage
- M represents a "mid-level" voltage
- L represents a "low-level" voltage.
- The first letter corresponds to the voltage on wire A
- The second letter corresponds to the voltage on wire B
- The third letter corresponds to the voltage on wire C

CPhy state values and their corresponding wire voltages are defined in the CPhy generator as follows:[1]

Table 2 – CPhy HS States

State Value	ABC Wire Voltages
0	MMM (illegal HS state)
1	LMH (+Z)
2	MHL (+Y)
3	LHM (-X)
4	HLM (+X)
5	MLH (-Y)
6	HML (-Z)
7	MMM (illegal HS state)

[1] Note that the CPhy transmit states used here correspond to the state values that would be measured by a CPhy receiver. As receiver measurements are made differentially between wires, the differential voltages seen by the receiver are not the same as the single-ended voltages driven by the transmitter and (and described in table Table 2 – CPhy HS States.

State values of 0 and 7 are not legal values as defined by CPhy. However, they can be used if desired to cause medium-level voltages to be output on all three wires in a trio.

3.6 CPhy LP States

CPhy LP states represent low-power voltage levels for the ABC wires of a trio and can be specified in CPhyGenCtl as integer values ranging from 0-7, though only four of the eight LP states are used in CPhy: 0 (LP000), 1 (LP001), 4 (LP100), and 7 (LP111). The wire-order for LP symbols is ABC, e.g. LP100 represents wire-A at LP high-voltage, and wire-B and wire-C at LP low voltage.

Table 3 – LP States

LP State Value	**LP State**
0	LP000
1	LP001
2	LP010 (unused LP state)
3	LP011 (unused LP state)
4	LP100
5	LP101 (unused LP state)
6	LP110 (unused LP state
7	LP111

Some commands encode the LP state for all four lanes as a 16-bit hex value. The argument name for this value is "LPValue[15:0]". For this argument, each nibble of LPValue contains the LP state for a corresponding lane as follows:

- LPValue[2:0] – LP State for lane 0
- LPValue[6:4] – LP State for lane 1
- LPValue[10:8] – LP State for lane 2
- LPValue[14:12] – LP State for lane 3

4 Operational Concepts

4.1 Pattern Generator (PG)

As previously mentioned, prior MIPI solutions offered by The Moving Pixel Company have used a general-purpose pattern generator (PG3A) and a MIPI-specific probe (P331, P332, P338) for DPhy stream generation. The P339 CPhy Generator streamlines this architecture by incorporating the pattern generation function into the instrument. Thus, the PG3A is no longer required.

However, this document and the CPhyGenCtl software still refer to the PG as a distinct (internal) component of the CPhy Generator. For example, the status pane in the main window shows whether the PG is running or idle, and buttons labeled "Stop PG" and "Restart PG" are used to control the CPhy Generator's internal pattern generator engine for transmission.

4.2 CPhy Bus Behavior

Once the CPhy Generator has been powered on and CPhyGenCtl is connected, CPhy lanes are driven to LP111. This is the default state of the instrument whenever the PG is idle.

When commands are sent using CPhyGenCtl, one or more lanes become active, either transitioning to HS mode or signaling in LP mode, depending on the command and the current DT Mode setting. Generally, the Lane Cnt setting determines which lanes become active, though it is possible for the low-level File command to send LP data on all 4 lanes, independent of Lane Cnt.

The Options->Loop Commands menu option determines whether commands are sent once or looped on the bus indefinitely. If this option is not checked, the command will be sent and then the bus will return to the LP111 state, with the PG status indicating that it is idle. Otherwise, PG status will indicate it is "Running" while the command loops.

Some lower-level phy commands may loop the PG, regardless of the "Loop Commands" setting. These include the TGR and PRBS test sequence commands.

Looping commands can be stopped by clicking the "Stop PG" button, in general returning the PG to idle state and the CPhy bus to LP111.[2] Also, unless command insertion is enabled (Options->Enable Command Insertion), sending another command while a command is looping will cause the application to ask whether you want to stop the PG to allow the sending of the command.[3]

[2] An exception to this rule is when the "Hold Last Symbol Test Mode" option is enabled. In this case, looping HS packets including TGR and PRBS sequences stop looping, output the Postamble sequence, and hold the last Postamble symbol indefinitely until "Stop PG" is pressed a second time.

[3] Unless the GUI option "Suprress confirmation message when stopping PG" is set, in which case the looping command is automatically stopped.

5 Using CPhyGenCtl

The CPhyGenCtl software is the configuration and control application for the CPhy Generator. Using its GUI interface, users can interact with the instrument, configuring commands to send on the bus, monitor status and review DUT response data. In addition, the application supports a remote control interface (called RPC) that supports controlling nearly all capabilities accessible through the GUI using a remote, user program.

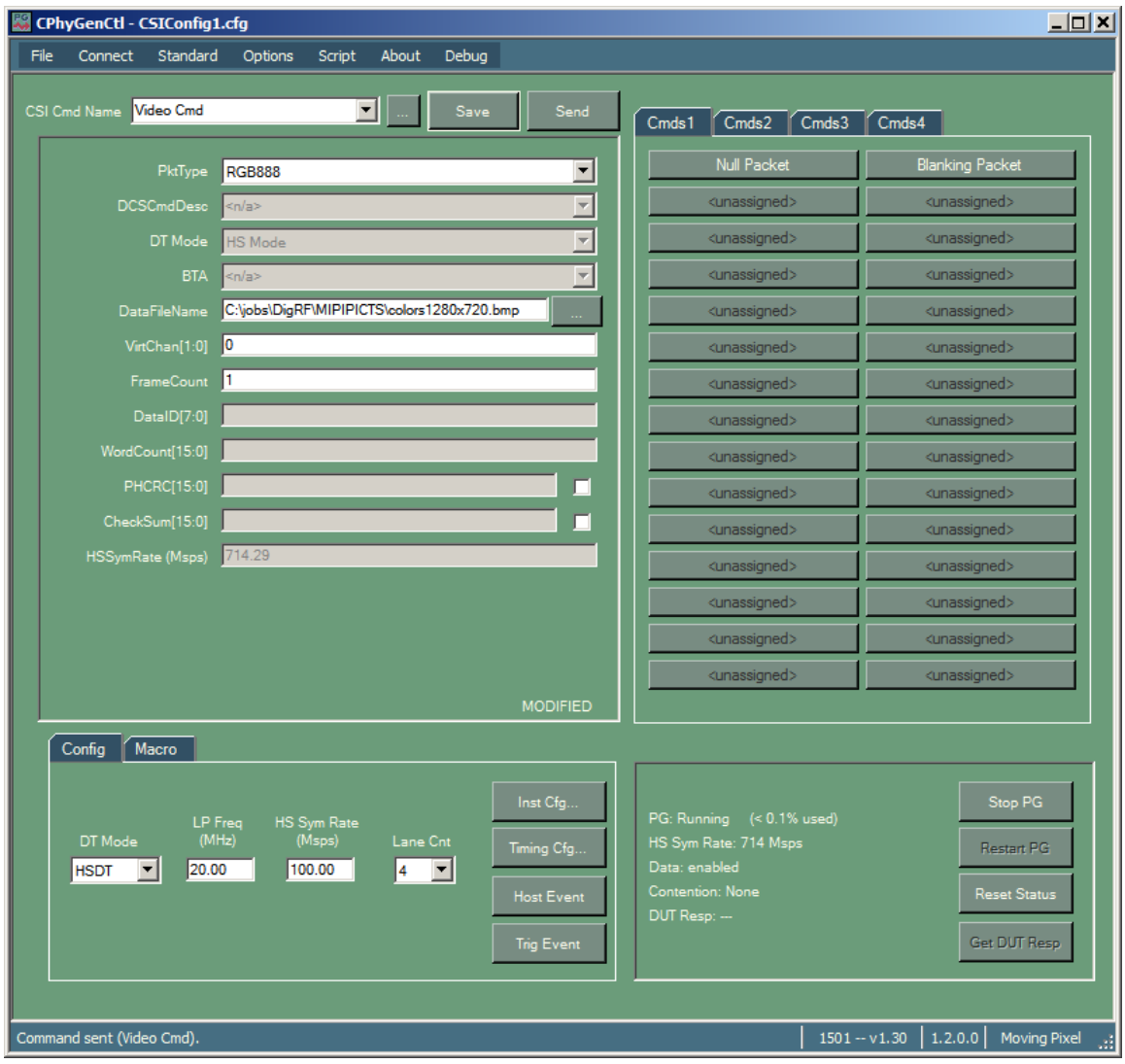

Figure 2 – CPhyGenCtl Main Window

5.1 GUI Overview

Most of the interaction between the user and DUT occurs in the CPhyGenCtl main window (see Figure 2). It has several distinct sections.

- The left pane of the main window allows the user to define commands and their arguments, naming them to allow for later recall and sending to the CPhy Generator, as well as assigning them to command buttons.
- The right pane of the main window consists of 30 command buttons that can each be individually associated with a defined command, allowing single-click sending of command commands to the CPhy Generator.
- The bottom left pane of the main window provides controls for configuring parameters of instrument transmission, video timing definitions, physical characteristics of the CPhy bus, and trigger events.
- The bottom right pane of the main window comprises instrument status and operational controls.
- The status bar at the bottom of the main window displays informational and error messages and indicates the current instrument connection and software/firmware versions.

The following sections describe more details of CPhyGenCtl operation.

5.2 Generic List Dialog

A generic List dialog is used to organize named item lists associated with a drop-down control. Named items are listed in a central list-box, with controls on the side to manipulate items in the list. The List dialog supports the follow functions:

Table 4 – List Dialog Functions

Function	**Method**
Clear the list	Click on the Clear button
Sort the list alphabetically	Check the Sort checkbox
Reset the list to its default	Click the Defaults button
Delete items in the list	Select one or more items in the list and click the Delete button
Move item(s) up in the list	Select one or more items in the list and click the Up button
Move item(s) down in the list	Select one or more items in the list and click to Down button

After viewing and/or modifying the list, clicking OK closes the dialog and records any changes. Clicking Cancel closes the dialog and discards any changes to the list.

A List dialog is associated with three controls in CPhyGenCtl:

- "Cmd Name" control in the main window
- Frame Timing Configuration control in the Frame Timing dialog
- CPhy Timing Configuration control in the CPhy Timing Configuration dialog

Immediately to the right of each of these controls is a button labeled with an ellipsis ("…"). Clicking this button brings up the List dialog, initialized with element names from the associated control. Figure 3 shows an example Frame Timing List dialog.

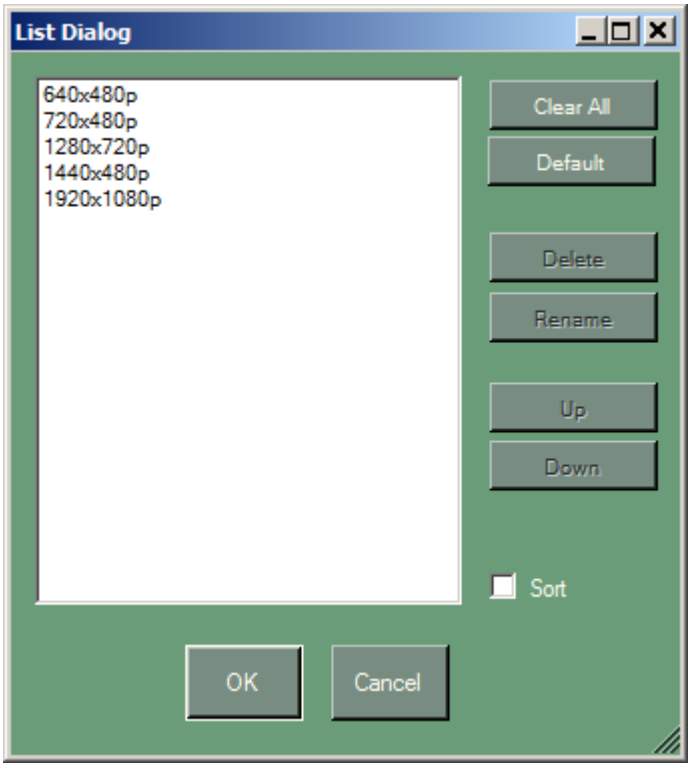

Figure 3 – Frame Timing List Dialog

5.3 Connecting to the CPhy Generator

Before any commands can be sent, the CPhyGenCtl application needs to connect to the CPhy Generator connected via USB to the host machine. The appropriate license file for the instrument must be located in the application directory for successful connection (see section 2.1).

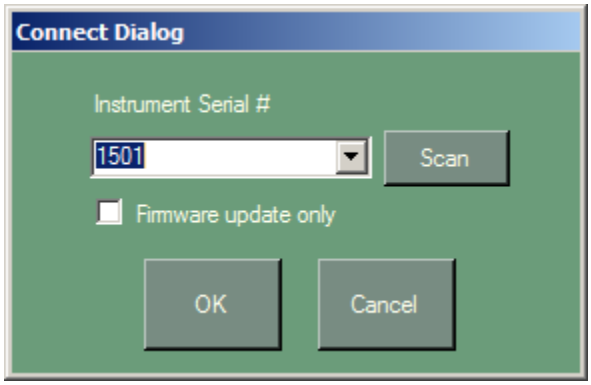

Figure 4 – Connection Dialog

When CPhyGenCtl is launched, the application automatically brings up a connection dialog (see Figure 4) with a drop-down control showing visible CPhy Generator instruments available for connection. An alternative way to bring up this dialog once the application is running is via the "Connect…" menu option in the Connect menu.

To connect to an instrument or change connection to a different instrument, select the serial number of the instrument you want to connect to and click OK. Select "Offline" to run in offline mode without an instrument connection. Click the Cancel button will maintain the current connection state. Once the dialog is closed, the instrument serial number and firmware version will be displayed in a bottom-right corner of the status pane to indicate the current connection. Otherwise, "Offline" will be displayed.

If an instrument is not powered on and connected via USB before this dialog is launched, the Scan button can be used to show newly available instruments.

Also, if an instrument's firmware has been corrupted and does not boot properly, it is important to check "Firmware update only" when connecting to the instrument for the purpose of updating its firmware. This will disable communication with the central FPGA, which may not respond properly if it is in a bad state. After connecting to the instrument in this way, please proceed directly to the Update Firmware dialog (Connect menu) without other main window control interactions.

When testing is complete, the "Disconnect" menu can be used to close the instrument and return to offline. The application does this automatically on closing so it is not required.

5.4 Instrument Configuration

There are two main sets of controls for instrument configuration. The main window contains the most important and commonly adjusted controls such as LP and HS symbol frequencies, lane count, and DT Mode. Less used controls can be found in the Instrument Configuration Dialog.

5.4.1 Main Window Configuration Controls

Four controls in the "Config" tab at the bottom of the main window are used to describe principal CPhy configuration parameters. These controls are described below:

DT Mode: this combo box sets the default data transfer mode for commands whose DT Mode field is set to "Default". Packets may be sent as either HS packets (HSDT mode) or LP packets (LPDT mode). Individual commands can override the default DTMode setting by setting the DTMode field in their command arguments. Note: this mode only applies to non-video commands and does not affect video-mode operation or low-level DPhy/CPhy commands such as escape commands as well as the PRBS/TGR test sequences.

HS Sym Rate: this control sets the lane symbol rate of the CPhy bus for all commands *except* video-mode commands.[4] The supported range for the HS Sym Rate setting of the P339 CPhy Generator is 23.4 Msps to 2600 Msps.

Note that changing this control setting does not immediately affect the symbol rate, in particular if a program is currently running. Instead it schedules a frequency reconfiguration for the next sent command. To see the current symbol rate on the CPhy bus for a running program, look at the HS Sym Rate display in the status pane *not* the HS Sym Rate setting in the instrument configuration pane.

LP Freq: this control sets the frequency of LP bits on the CPhy bus (i.e. determines T(LPX) in the DPhy/CPhy standard).

Lane Cnt: this control selects the number of CPhy lanes to use (1-4). HS packet data is demultiplexed onto CPhy lanes according to the CSI protocol specification. Unused lanes are held in the LP111 output state.

5.4.2 Instrument Configuration Dialog

Once connected, the CPhy Generator can be configured with parameters specific to your testing needs via the Instrument Configuration dialog brought up from the main window via the "Inst Cfg..." button in the lower-left pane (Config tab) of the main window. Note that adjustments can be made even while a program is running

After entering desired settings on the dialog, click the OK button to accept them or Cancel if you want to discard them. Since settings are updated as they are adjusted, Cancel in this case means setting control values back to their values before the dialog was opened. Clicking on the "Defaults" button sets all controls to their default values.

Figure 5 shows the Instrument Configuration dialog. Below is a description of its controls:

Force Test Pattern: checking this box causes a test HS symbol pattern to be output on all CPhy lanes at the current HS Sym Rate setting.[5] The pattern alternates between state value 6 and state value 1. This pattern is generally used while the instrument is connected to an oscilloscope to ensure the instrument is working properly and to visually adjust voltages and delays for calibration.

[4] Video mode commands follow the currently selected frame timing configuration, which sets the HS Sym Rate for the video command. main window HS Sym Rate setting if it has been changed during program output.

[5] If a program is currently running, the HS Sym Rate used for the test pattern will be the current program symbol rate, which may be different than the main window HS Sym Rate setting if it has been changed during program output.

Common HS Voltage: this check box, when checked, disables the HS voltage controls for all lanes except for lane 0. Adjustments to data lane 0 HS voltages are automatically applied to all lanes. Otherwise, HS voltage settings can be independently adjusted.

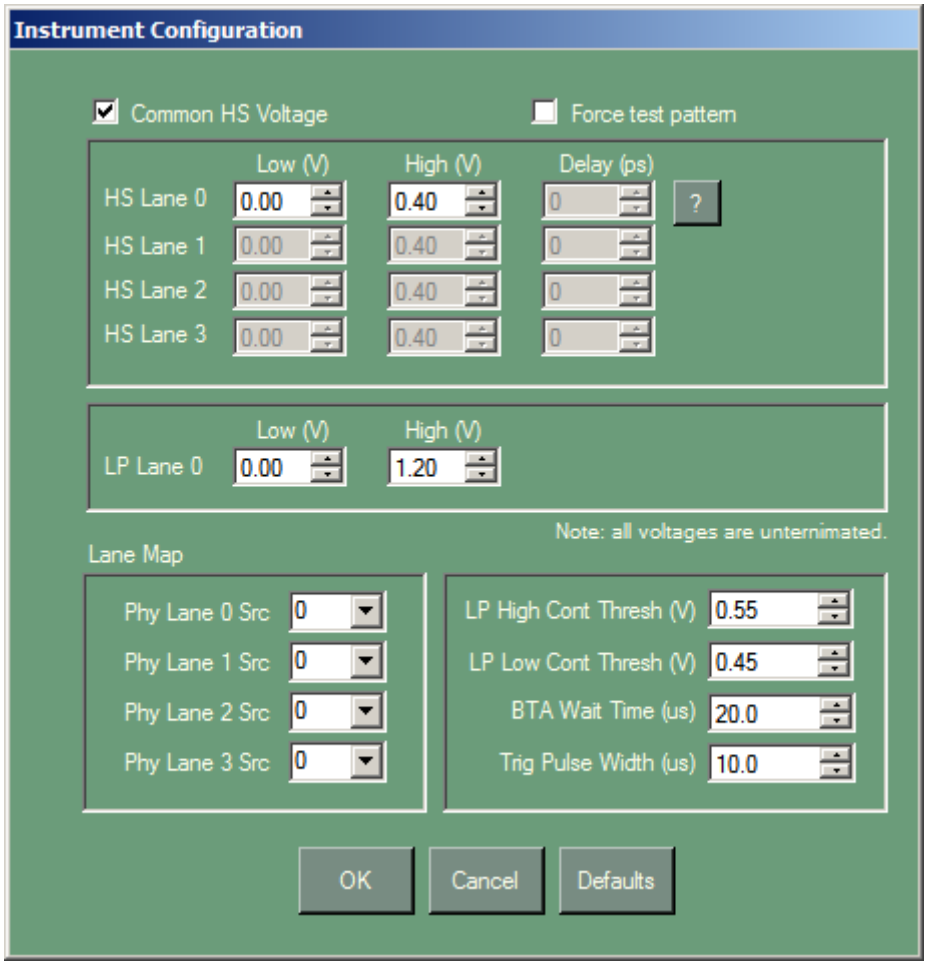

Figure 5 – Instrument Configuration Dialog

HS Low/High Voltage Controls: these controls set the *unterminated* HS voltage levels for CPhy lanes. When connecting to real hardware, HS voltages generally should be set to 0.0V (low) and 0.4V (high). Slave termination resistance will reduce HS levels to the expected swing of 0.1 - 0.3V. HS voltage limits are -0.6V and 1.2V.

HS Delay Controls: these controls set the relative delays of each lane. Delays controls are only enabled when the user has checked "Enable Lane Delays" in the CPhy Timing Configuration dialog (Options->Configure CPhy Timing). This is because programs must be built appropriately to support delay adjustment and so must be statically enabled.

When enabled, delay adjustments are in ps and have a range of 0 to 15000 ps. Values are quantized according to equations related to the HS symbol rate. For example, at rates above 1500 Msps, only integer symbol delays are supported. For rates between 750 and

1500 Msps, ½ symbol delays are supported. For rates between 375 and 750 Msps, ¼ symbol delays are supported. And so on.

LP Low/High Voltage Controls: these controls set the LP voltage levels for all CPhy lanes. LP voltages generally should be set to 0.0V (low) and between 1.1V and 1.3V (high). LP voltage limits are -0.28V and 1.8V.

LP Contention Thresholds: these two controls determine the threshold voltage used for contention detection.

The LP Low contention threshold determines the voltage at which a LP Low fault is flagged (i.e. when the transmitter attempts to drive an LP0 and measures a voltage on the line greater than the threshold). This value is nominally 0.45V according to the CPhy specification.

The LP High contention threshold determines the voltage at which a LP High fault is flagged (i.e. when the transmitter attempts to drive an LP1 and measures a voltage on the line less than the threshold). This value is nominally 0.55V according to the CPhy specification.

BTA Wait Time: (not applicable for CSI) this control sets the time-out period after a BTA is sent to wait for a return BTA sequence from the DUT. Normally, if a return BTA is seen, the probe acknowledges the BTA and continues its program. Similarly, if the time-out period expires, the probe reasserts LP101 and continues its program.

Trig Pulse Width: set the trigger pulse width when a trigger pulse is sent, either through the "Assert Trigger" command in a program or the "Trig Event" button in the main window.

Phy Lane 0 Src: sets the logical source lane for physical lane 0. While normally, this is set to zero, this control allows logical lane 1, 2, or 3 to be output on physical lane 0 as well. (Note, however, that contention detection always is associated with physical lane 0 regardless of logical mapping).

Phy Lane 1 Src: sets the logical source lane for physical lane 1. While normally, this is set to one, this control allows logical lane 0, 2, or 3 to be output on physical lane 1 as well. One use of this control is to duplicate lane 0 traffic on lane 1 for separate scope monitoring during single-lane mode communication.

Phy Lane 2 Src: sets the logical source lane for physical lane 2. While normally, this is set to two, this control allows logical lane 0, 1, or 3 to be output on physical lane 2 as well.

Phy Lane 3 Src: sets the logical source lane for physical lane 3. While normally, this is set to three, this control allows logical lane 0, 1, or 2 to be output on physical lane 3 as well.

Defaults: button to reset all controls to their defaults. The user is prompted via a message box to confirm the reset operation.

5.5 CPhy Timing Configuration

The CPhy Timing Configuration Dialog provides controls for the user to specify various aspects of CPhy bus behavior. General categories of controls are:

- Per-lane symbol sequence settings for preamble, postamble, and sync sequences
- CPhy bus timing registers
- TGR length registers
- Lane delay enable

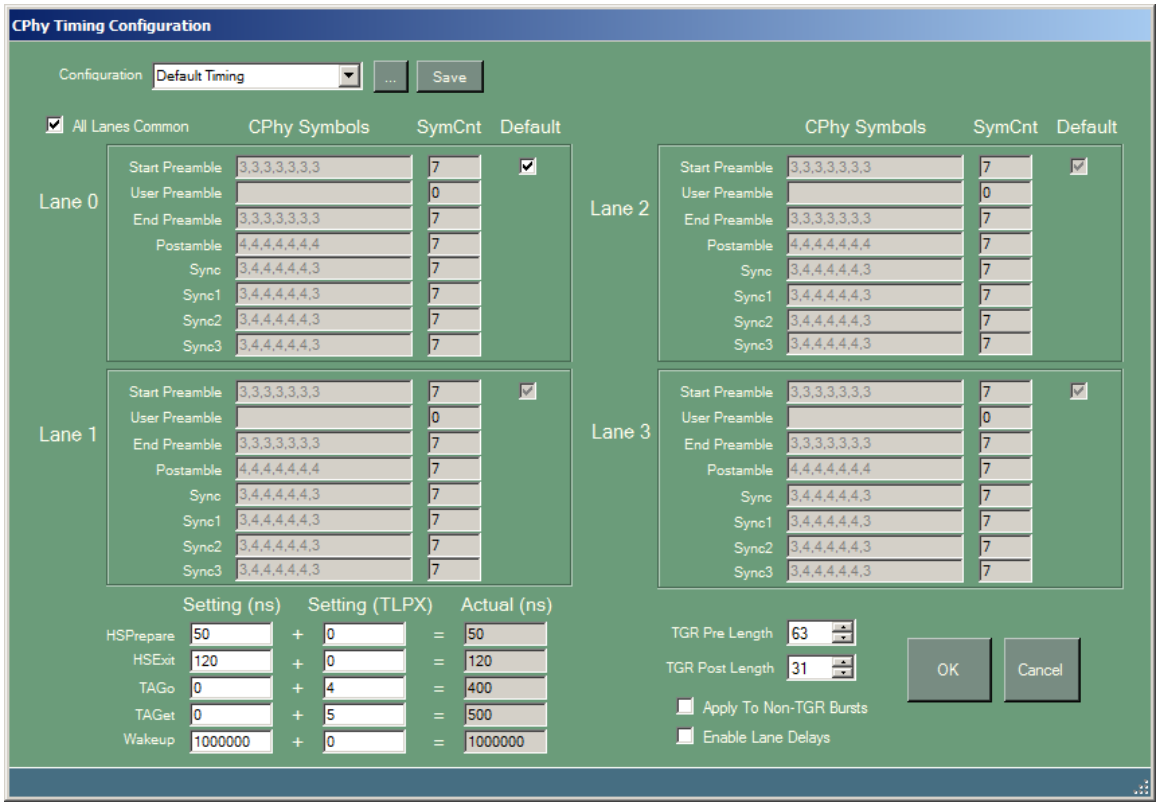

Figure 6 – CPhy Timing Configuration Dialog

5.5.1 Symbol Sequences

Each lane has entries for defining the symbol sequences for various protocol segments of the HS burst. CPhy typically allows for the definition of only one of the sequences -- User Preamble – and if defined, requires that it be 14 symbols in length. For low-level testing, however, CPhyGenCtl allows other sequences to be set to non-standard values or sequence lengths.

The text boxes defining these sequences consist of one or more symbols separated by commas. Associated with each sequence is a read-only text box that displays the number

of symbols in the sequence (SymCnt). The following protocol segments can be defined for each lane:

1. Start Preamble – the first symbols sent for the HS burst preamble. This sequence is also replicated by (TGR Pre Length + 1) for TGR and PRBS packets, as according to the CPhy specification. CPhy declares that this sequence should consist of seven 3s.
2. User Preamble – the next symbols sent for the HS burst preamble. This sequence is optional in the CPhy specification, but, if present, is declared to be 14 symbols in length.
3. End Preamble – the final symbols sent for the HS burst preamble. CPhy declares that this sequence should consist of seven 3s.
4. Postamble – the symbols sent at the end of an HS burst. CPhy declares that this sequence should consist of seven 4s.
5. Sync – the symbols sent after the preamble just before an HS packet header. CPhy declares that this sequence should be the seven symbol sequence: 3, 4, 4, 4, 4, 4, 3.
6. Sync1 (DSI only) – the symbols sent before all subsequent packets in an HS burst (i.e. all but the first packet). DSI declares this sequence should be the seven symbol sequence: 3, 4, 4, 4, 4, 4, 3.
7. Sync2 – the symbols sent between duplicate packet headers (PH). CPhy does not call this sequence out separately from Sync as they are declared to be the same.
8. Sync3 (DSI only) – the symbols sent between the last PH and the start of packet payload of a long packet. DSI declares this sequence should be the seven symbol sequence: 3, 4, 4, 4, 4, 4, 3.

A checkbox is provided for each lane labeled "Default". If checked, all sequences for that lane are set to their CPhy defaults and controls are set read-only. Uncheck these boxes to change symbol values.

A checkbox is also provided labeled "All Lanes Common". When checked, the sequence definition controls for lanes 1, 2, and 3 are duplicated from lane 0 settings and made read-only.

Finally, there are a couple application requirements when setting symbol sequences:

1. The protocol segments for each lane must have the same number of symbols.
2. The Sync1, Sync2, and Sync3 sequences are required to be 0 or 7 symbols.

An error message is displayed if these requirements are not met.

5.5.2 CPhy Bus Timings

In the lower-left corner of the CPhy Timing Configuration dialog are controls for the user to set CPhy bus timings:

- HSPrepare

- HSExit
- TAGo
- TAGet
- Wakeup

Please see the CPhy specification for timing parameter definitions.

For each timing parameter, the user can set two integer values that are summed to form the actual timing parameter used. One value is in nanoseconds and the other value is in units of TLPX (the LPFreq clock period). Note that if the "Enable Lane Delays" checkbox is checked, the minimum HSPrepare and HSExit times is 30 ns, enforced regardless of user settings.

5.5.3 TGR Length Registers

Two TGR registers – TGR Pre-Length and TGR Post-Length – define additional replication counts for the Start Preamble and Postamble respectively during TGR and PRBS commands. For example, a TGR Pre-Length of 3 means that the Start Preamble will be repeated an additional three times for a total replication count of four during TGR and PRBS commands.

The checkbox labeled "Apply To Non TGR Bursts" is provided to use the TGR Pre-Length and TGR Post-Length multipliers for all HS packet types (not just TGR). If this option is checked, HS preamble and postamble sequences are lengthened accordingly. Note that this usage is non-conforming to the CPhy standard, but is useful for low-level testing.

5.5.4 Lane Delay Enable

A checkbox labeled "Enable Lane Delays" indicates whether or not the output stream is constructed to support lane delay adjustment of up to 15 ns. Special stream construction is required to support lane delays due to hardware limitations at LP/HS boundaries.

The reason this is required is that hardware must always switch all lanes from LP to HS or vice-versa at the same time. As lane skew is implemented by delaying data relative to other lanes, the stream must be constructed such that the delay still results in a reasonable, decodable CPhy waveform. To achieve this, the application takes advantage of the fact that hardware outputs a mid-HS level when outputting LP111 or LP000 symbols in HS mode.

As a result of enabling lane delays, the HS Prepare time will nominally end in a 15 ns period at mid-level HS voltage. Similarly, the HS Exit time will nominally start with a 15 ns period at mid-level HS voltage. This waveform modification should still result in legal decoding of HS bursts by a receiver, with the benefit of being able to skew individual lanes relative to each other by up to 15 ns.[6]

[6] Note that actual lane delay settings are located in the Instrument Configuration dialog (via the Inst Cfg… button in the main window).

5.6 Commands

5.6.1 Defining a New Command

Commands are defined using the left pane controls of the main window and are grouped based on the current standard according to category. Thus, before defining a command, the user should select the desired protocol via the Standard menu, currently either CSI or DSI. Then, to define a new command, perform the following steps:

1) Select the command type via the PktType drop-down control. Note that once a command field is edited, the label in the "Cmd Name" control changes to "<New Cmd>" and "MODIFIED" appears below the command fields to prevent confusion with any existing command definition.
2) If the command type is a DCS command from the DSI command set (i.e. DCS Short Write, DCS Long Write, or DCS Read Request), the DCSCmdDesc drop-down control will be enabled. Select the DCS command type from this control.
3) Depending on the command selected, argument fields will be labeled and enabled for the user to enter values. Certain arguments are pre-filled such as the virtual channel, DT Mode, and BTA. The user may change these pre-filled values if desired.
4) Commands that support sending either as a LP or HS command initially have their DTMode argument set to "Default", which means the command will be sent using the current default DTMode setting (DT Mode control in the lower-left of the main window). Alternatively, the user can specify the DTMode for certain commands by changing the DTMode argument to either LPDT or HSDT.
5) Setting BTA to "Yes" requests that a bus-turn-around sequence follow the sending of the command. This is usually used for read commands to allow the device under test to respond with data.
6) Numeric fields are assumed to be decimal unless appended with 'h' (hexadecimal) or 'b' (binary). For example "67h" = 103 and "1101b" = 13.
7) Certain derived fields of the command will also be displayed in read-only text boxes: DataID, WordCount, PHCRC, CheckSum). The PHCRC and CheckSum fields can be modified to cause packet errors by ticking the checkbox associated with the field. The other read-only fields cannot be modified directly by the user but are computed according to the MIPI specification based on the command arguments.[7] Note that video commands do not fill in these fields because they do not represent a single packet.

5.6.2 Naming a Command

Once a command is defined, the command may be named so that it can be assigned to a command button and saved in a configuration file. To name a command, click on the Save button to bring up a dialog to enter the command name. In the Name Cmd dialog, the command name field is initialized with the original command name present before

[7] To modify read-only fields, the user can use the "Send Cmd To File" menu option (File menu). This outputs the command bytes to a text file that can subsequently be modified and resent using the File command.

editing. This allows the user the option to simply click OK to replace the original command. Otherwise, type in a new name to create a new command.

Named commands show up in the "Cmd Name" drop-down box of the main window. Selecting a named command in the list fills in the argument field controls with values associated with the command.

5.6.3 Editing an Existing (Non-Macro) Command

An existing non-macro command definition can be edited as follows:

- Load the command field values into the edit controls either by right-clicking on a button having a command assignment and selecting "Edit Current Command" or selecting the command name from the "Cmd Name" drop-down control.
- Edit the command fields.
- Click the Save button.
- Click OK in the naming dialog.
- Click OK to the confirmation dialog to replace the existing command definition.[8]

Macro editing is described in Section 5.8.2.

5.6.4 Managing Commands (renaming, ordering, deleting)

The button labeled with an ellipsis ("…") immediately to the right of the "Cmd Name" drop-down control allows the user to manage the named command list, including renaming commands, reordering and sorting commands in the drop-down list, deleting commands, etc. Click this button to bring up a List dialog to perform these functions. See section 5.2 for more information about this dialog.

5.6.5 Saving/Restoring Command Configurations

A "default.cfg" file is provided with CPhyGenCtl installation that has a few example commands defined. Once you add new command definitions, you can save your command configuration, command set, and button assignments in a configuration file.

Note that command definitions and button assignments are *not automatically saved when the application is closed.* Use the "File->Save XXX.cfg As..." menu option and specify the name of the file to save to. Similarly, you can load a configuration file with the "File->Load..." menu option. The last configuration file saved before exit is restored on relaunch.

5.6.6 Special Command Types

5.6.6.1 Video Mode Commands

Video-mode commands are one of many possible video-mode command types that cause the application to build and send one or more video frames, which are sourced by user-provided image files (or special test pattern names). A video-mode frame consists of

[8] The confirmation dialog can be disabled in the GUI Options dialog (Options->GUI Options).

multiple packets and has specific structure defined by the current standard (CSI or DSI). Timing between frame structure elements is controlled by the current frame timing settings. Video-mode commands are described further in section 5.7.

5.6.6.2 Display Stream Compression (DSC) Video

DSI version 1.2 defines a new packet type: Compressed Pixel Stream. This packet type allows a compressed stream constructed according to the VESA Display Stream Compression (DSC) standard to be sent. CPhyGentCtl provides a separate purchase option for extensive support for DSC-related functions, which includes the following[9]:

- Automated building of DSC streams from image files and test patterns for video-mode and Write Memory commands that use the compressed video format.
- Automated extraction of PPS data for DSC files, or automated construction of PPS data for image files when sending the Picture Parameter Set command.
- Support for the DPX and DSC file types in compression commands
- User decoding/encoding of DSC files to/from image files
- Image viewing of DSC files
- Write Memory partitioning and video-mode packetization option to send either one chunk per packet or possibly multiple chunks per packet.

Compressed video is described further in section 5.7.5.

5.6.6.3 Macros

A macro is a user-defined sequence of MIPI commands, sent as a single unit. Macros can be defined in the GUI, in a script file, or via a user program using RPC, then named and sent like any other command. Macro commands are described in section 5.8

5.6.6.4 Phy Commands

Phy commands are special low-level or LP signaling commands. These are defined by the CPhy specification such as BTA, ULPS, etc and include TGR and PRBS sequences. Phy Commands are described further in section □.

5.6.6.5 File Command

The File Command is general-purpose, low-level command that uses a text file as input to define arbitrary LP and HS signaling sequences. Using special file command syntax, low-level LP and HS lane states and higher-level packet data bytes can be intermingled as desired to create user test sequences and error packets. Command syntax also supports useful features such as automatic ECC and CRC generation, HS burst signaling generation (e.g. HS entry/exit and sync), and packet data demultiplexing. The file command is described further in section 5.10).

5.6.6.6 Script Commands

Script commands are RPC (remote procedure call) commands contained in a user-defined text file. RPC commands provide the function of almost every action provided in the

[9] Without the DSC option, CPhyGenCtl does not provide these functions but still supports sending compressed video using binary files.

GUI, such as command and macro definition, command button management, frame timing, CPhy timing, instrument configuration, sending of commands, etc.

Script files are sent using the "RPC Script" command, specifying or browsing for a file name in the "File Name" argument field. Please see the document "CPhyGenCtlRPC.pdf" for more information.

One new feature to CPhyGenCtl is the ability to embed script commands directly in the filename text. If the filename begins with a comment string "//" or script command symbol "#", then the filename string is treated directly as a script. For multi-line scripts, editing can be achieved using the Script Edit Dialog, which is brought up instead of a file browsing dialog when the ellipsis button "…" is clicked next to the command filename (assuming the filename starts with a "//" or "#").

The Script Edit Dialog is simply a dialog with a text box containing the script that the user can type in. A Clear button is provided to clear the text box. Otherwise, the user can edit the script and then click OK to accept the changes or Cancel to discard any changes.

5.6.6.7 Variable Argument Commands

Certain commands allow the user to specify a variable number of parameters through an argument text box named "Params[]", specifically the following commands:

1. Generic Short Write Command (DSI)
2. Generic Read Request (DSI)
3. DCS Short Write (DSI)
4. DCS Read Request (DSI)
5. Custom DCS Command (DSI)
6. Custom Command
7. Custom Long Command

The differences between these commands are minimal. The Custom Command and Custom Long Command allow the user to specify an arbitrary DataID byte. The other commands implicitly fill in the Data Type and allow the user to specify a Virtual Channel argument. The DCS commands also allow the user to specify the DCS command byte. All allow a variable number of arbitrary arguments, computing the PHCRC, packet length (for long packets), PHCRC, and Checksum (for long packets).

The actual packet constructed may have either long or short packet formatting, depending on the number of values parsed from the Params[] argument string. For non-DCS packet types, if Params[] contains 0, 1, or 2 bytes, then short packets will be constructed. For DCS packet types, if Params[] contains 0 or 1 byte, then short packets will be constructed (prepended with the DCSCmd byte). Otherwise, long packets will be constructed and you will see additional read-only fields for the packet length and checksum appear.

The Params[] argument string consists of byte values separated by white-space or commas. As with other argument fields, values are assumed to be decimal unless

appended with 'h' (hexadecimal) or 'b' (binary). For example, "30 41h 10101b" is a legal parameter string.

5.6.6.8 Long Data File Format Commands

Certain non-video packet types (e.g. Long Pkt (Non-Image Data), Generic Long Write, DCS Long Write) have a "File Name" argument. For these commands, the file is treated simply as a binary file. Note that the File Command has a special ASCII text file format described further in section 5.10.

With the exception of DCS Write Memory commands, the entire file is sent in a single long packet. The WriteMemory commands have special support in CPhyGenCtl because these commands are often used to send video data to a frame buffer on the DUT. The next section describes this support.

5.6.6.9 DCS WriteMemory Commands

For the Write Memory Start and Write Memory Continue commands, the WriteLen argument specifies the number of bytes to send and the FileOffset argument specifies the starting byte offset in the file or frame buffer.

IMPORTANT: the parameter name FileOffset is misleading when using BMP files and the video format is not RGB888. The parameter would better be called FrameOffset. The reason is that this parameter is used *after* the file has been imported and converted to the output video format.

For example, for a 100 x 100 image in RGB666 (18-bits per pixel), an offset of 1800 references pixel 800 (= 1800 / (18/8)), which is the first pixel of line 8. In RGB888, an offset of 1800 references pixel 600, the first pixel of line 6. The line length in RGB666 is 100 * (18/8) = 225 and, in RGB888, is 100 * 3 = 300

CPhyGenCtl has additional features that enhance the behavior of sending Write Memory commands:

- **BMP File Decoding –** If a file has extension ".bmp", the image data is first imported before it is sent in the long packet. In this case, FileOffset refers to the byte offset in the decoded image data and not the file. In addition, the decode format (e.g. RGB 8-8-8, 5-6-5, etc) can be selected via the ConfigWriteMemory dialog.
- **Block Partitioning –** A single WriteMemory command can be partitioned into a sequence of WriteMemory commands. The user configures this behavior via "Configure WriteMemory Commands…" in the Options menu.

Figure 7 shows the ConfigWriteMemory dialog which has the following controls:

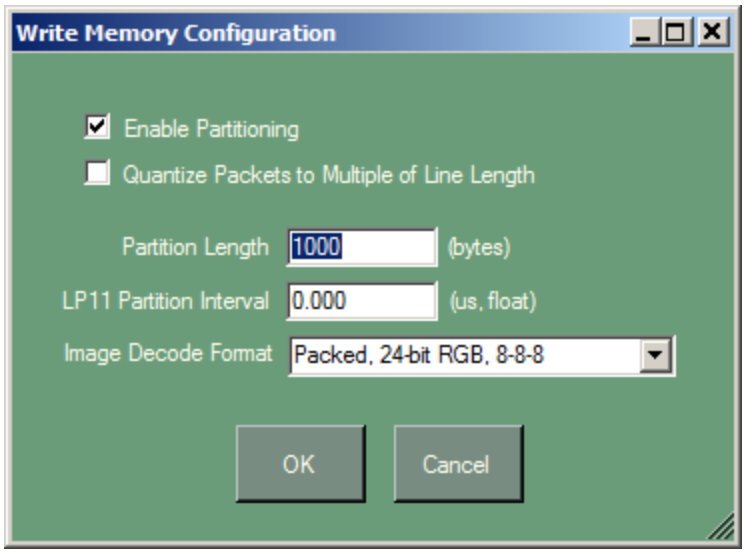

Figure 7 – Config WriteMemory Dialog

Enable Partitioning CheckBox – when checked, partitioning is enabled and WriteMemory commands will be deconstructed into a sequence of WriteMemory commands according to the parameters in the dialog. Otherwise, WriteLen values greater than 64 KB will return an error.

Quantize Packets to Multiple of Line Length – when checked, partition lengths are set to the greatest multiple of the image line length less than the "Partition Length" setting. The image line length is computed in one of two ways, depending on whether "Allow Image Rescaling" is enabled. If rescaling is enabled, then the image line length is computed using the HActive setting in the Frame Timing configuration and the pixel format for the selected "Image Decode Format". Otherwise, the image line length is computed using the input line width times the bytes per pixel for the format specified in "Image Decode Format".

Partition Length – indicates the maximum length to use for component WriteMemory commands in the sequence.

LP11 Partition Interval – indicates the amount of time to spend in LP11 between each component Write Memory command in the sequence. Specify zero for no delay between component commands. In this case, if the "Send single packet per HS burst" option is not set and the command is sent in HSDT mode, the entire command sequence will be sent in a single HS burst (otherwise, each component Write Memory command will be sent in its own burst).

Image Decode Format – specifies the destination image format to decode files that have an extension of ".bmp" or ".jpg" (or also ".dpx" for compressed video).

5.6.6.9.1 Sending Compressed Video

If the "Image Decode Format" is set to "DSC Compressed" then data sent with the Write Memory command is assumed to be compressed video. Behavior is different depending on whether the DSC license option is installed:

With DSC License:

- Files with ".dsc" and ".dpx" extensions may be used
- Image files and test patterns can be used. If necessary (and enabled), the image is rescaled to the current output frame dimensions (as specified in the current Frame Timing configuration). Then, the image is compressed using the current DSC configuration settings.
- For image and DSC files, partitioning occurs (if enabled), ignoring the partition length. The partition length is set to the chunk size in the compressed stream or a multiple of the chunk size, based on the "Send One Chunk Per Packet" setting in the DSC configuration.
- For binary files, partitioning occurs (if enabled), according to the Partition Length setting. In this case, the user should set the Partition Length to the length of one or more chunks per the DSI standard.

Without DSC License:

- Only binary files may be used. Behavior is same as for binary files with license.

5.6.6.9.2 Partition Example 1

Consider a Write Memory Start command issued with the following arguments:

File Name = test.bmp
File Offset = 0
Write Len = 100000

and Write Memory configuration settings:

Enable Partitioning = true
Quantize Packets to Multiple of Line Length = false
Partition Length = 30000
LP11 Partition Interval = 1.5 us
Image Decode Format = Packed 24-bit RGB 8-8-8

Then, CPhyGenCtl sends the following packet sequence:

1) Write Memory Start with RGB888 image data bytes 0-29999
2) LP11 for 1.5 us
3) Write Memory Continue with RGB888 image data bytes 30000-59999
4) LP11 for 1.5 us
5) Write Memory Continue with RGB888 image data bytes 60000-89999
6) LP11 for 1.5 us

7) Write Memory Continue with RGB888 image data bytes 90000-99999
8) LP11 for 1.5 us

5.6.6.9.3 Partition Example 2

Consider a Write Memory Continue command issued with the following arguments:

File Name = test.bmp
File Offset = 0
Write Len = 60000

and Write Memory configuration settings:

Quantize Packets to Multiple of Line Length = true
Partition Length = 30000
LP11 Partition Interval = 0.0 us
Image Decode Format = Packed 24-bit RGB 8-8-8
HActive = 640 (frame timing dialog)

Then, CPhyGenCtl sends the following packet sequence:

1) Write Memory Continue with RGB888 image data bytes 0-28799
2) Write Memory Continue with RGB888 image data bytes 28800-57599
3) Write Memory Continue with RGB888 image data bytes 57600-59999

Note that the output image line length is 1920 bytes and that the first two partitions contain exactly 15 lines of image data because "Quantize packets to Multiple of Line Length" is set to true.

5.6.6.10 LP Delay

The "LP Delay" command is a low-level command that allows the user to assert a particular LP state on all lanes of the CPhy bus for a specified period of time. This command is generally used in a macro sequence.

The "LP Delay" command has two arguments:

- **Delay (us)** – the length of time to hold the bus in the specified LP state. This value is floating point.
- **LPValue[15:0] –** value encoding the LP states for all four lanes as described in section 3.6.

5.6.6.11 Delay To Pos

The "Delay To Pos" command is equivalent to "LP Delay" except that its Delay parameter is relative to a position in a macro called "Zero Pos", defined by the user with the "Mark Zero Pos" command. This command is useful when defining video frames in macros, allowing the user to define relative times between packets (e.g. active video packets, line start, line end, etc). See "LP Delay" for a definition of command arguments.

5.6.6.12 Mark Zero Pos

The "Mark Zero Pos" command (no arguments) defines a "Zero Pos" location in a macro. Subsequent "Delay To Pos" commands use "Zero Pos" as a reference position/time in the macro to define the relative duration of a delay. Note that multiple "Mark Zero Pos" commands are allowed in the same macro.

5.6.6.13 Wait Ext Event

The "Wait Ext Event" command (no arguments) inserts an indefinitely looping "LP Delay" period in a macro stream, causing output to suspend in the LP state until an external event edge occurs.

5.6.6.14 Assert Trigger

The "Assert Trigger" command is equivalent to "LP Delay" except it has a third parameter ("Grig Cmd") that specifies the specific trigger command to send, controlling the Trig Out signal of the CPhy Generator. The trigger command is implemented near the beginning of the "LP Delay" period, but exact timing depends on the output frequency and may vary relative to the period start with each trigger command.

Table 5 shows the trigger commands. Note that the trigger pulse width is defined in the Instrument Configuration dialog.

Table 5 – Assert Trigger Commands

Trigger Command Value	**Description**
1	Set Trig Out low
2	Set Trig Out high
3	Toggle Trig Out state
4	Pulse Trig Out

5.6.6.15 Cmd Insertion Point

The "Cmd Insertion Point" is equivalent to "LP Delay" except it additionally defines a location in the macro where another command can be inserted while the macro is looping. This feature is similar to command insertion during video mode.

The first two parameters (Delay, LPValue[15:0]) are the same as for "LP Delay". A third parameter, called Discard, is a flag that affects how the "LP Delay" period is affected by the length of the inserted command.

Table 6 – Cmd Insertion Point Discard Setting

Discard Value	Description
0	The inserted command is truly inserted into the program stream. Thus, if the Delay setting is 10 us and the inserted command requires 20 us, the ending time of the LP delay period would be 30 us.
1	The inserted command consumes the LP delay period as needed. If the inserted command does not complete before the end of the delay period, then subsequent stream

	data will be held off until the inserted command is complete. Thus, in the prior example, the ending time of the LP delay period would be 20 us.

Please refer to section 5.12.3 for further details of command insertion.

5.7 Video Mode Commands

5.7.1 Overview

When a user sends a video-mode command (e.g. DSI Pixel Stream packet types, CSI RGB, YUV, Raw, etc) CPhyGenCtl automatically builds an entire video-mode packet stream representing one or more video frames, having the appropriate timing and structure according to the MIPI standard. The video sequence can be played once or looped continuously to provide ongoing real-time video appropriate for emulation of CSI camera streams.

All video mode commands accept a "File Name" argument to specify an image file (BMP, JPG, GIF, TIF,) that contains the video frame data to send. To send a video sequence of more than one frame, "File Name" uses a special naming convention to allow CPhyGenCtl to derive the file names for subsequent frames in the sequence (see section 5.7.2).

Before sending a video mode command, additional parameters of video mode playback must be appropriately configured using the Frame Timing dialog (see section 5.7.3). These timing parameters, like other application settings, are saved and restored automatically in an application configuration file when CPhyGenCtl is closed and restarted.

Once a video command has been defined and the timing configuration parameters have been set, sending the command initiates the sending of video data. The internals of how video frames are constructed is described in section 5.7.4. Once video frame packets have been built and downloaded to the CPhy Generator instrument, video begins playing on the CPhy bus.

If the "Loop Commands" option (Options menu) is checked, video frames are played and looped indefinitely, stopping only when the 'Stop PG' button at the bottom of the main window is pressed. When the CPhyGenCtl stops a looping video sequence, the current frame is completed before exiting and returning to the LP111 quiescent state.

5.7.2 Video Command Definition

All video-mode commands have the four arguments:

1) File Name – source image file or test pattern name
2) VC – virtual channel
3) Frame Count – number of frames to send

4) Video Error – control word encoding a data bit error position in the frame

The VC and "Frame Count" arguments are self-explanatory. However, the "File Name" and "Video Error" fields are described further in the next sections.

5.7.2.1 File Name Syntax

The "File Name" argument to video-mode (and Write Memory) commands may reference many source types and structures:

- **Test Pattern**: special naming conventions are provided to allow simple test patterns to be constructed and used as the input image source for video-mode commands.
- **Image File**: an image file with extension BMP or JPG
- **Video File**: common video files (AVI, MPEG2, MPEG4, WMV, FLV).[10]
- **DPX File**: an image file with extension DPX.[11]
- **DSC File**: a compressed image file with extension DSC. [11]
- **Binary File**: any file that doesn't fall in the above categories is treated as a binary file. In this case, rescaling and format conversion are not supported and the file data is expected to be in the proper format for the video-mode command being sent. In particular, the file should have at least the number of bytes required by the current timing configuration and video format. That is, HActive * VActive * BytesPerPixel.[12]

When "Frame Count" is greater than one, thus requiring an image sequence as input, the file name may:

1) explicitly refer to an image sequence (i.e. video file)
2) implicitly refer to an image sequence (using special multi-file sequence naming)
3) reference only one image (in which case it replicated for the entire sequence)

5.7.2.1.1 Multi-File Sequence Naming

To play a sequence of multiple frames, the video-mode "File Name" argument may reference the first frame of the sequence and have the following format:

"<name><index>.<ext>"

where <name> is an arbitrary file name, <index> is an integer and <ext> is the file extension. CPhyGenCtl derives the file names of subsequent frames automatically by incrementing <index>.

[10] If a video file is referenced in a Write Memory command, currently only the first frame is used.
[11] DSC and DPX files are only supported when using the DSC compressed video format.
[12] The number of bytes per pixel for binary files when using the DSC compressed video format is derived from the configuration file in the current DSC configuration, which must have a BITS_PER_PIXEL assignment.

For example: if "c:\frame1.bmp" is specified as the file name and the frame count is set to four, CPhyGenCtl builds a video sequence from the files:

1) c:\frame1.bmp
2) c:\frame2.bmp
3) c:\frame3.bmp
4) c:\frame4.bmp

Multi-file Sequence Notes:

- All single image file types can be used in multi-file sequence naming.
- If "File Name" is a video file, the first "Frame Count" frames are used from the file.
- If "File Name" does not have the multi-file naming syntax (or a test pattern name is used), the sequence is still built repeatedly using the single given "File Name" reference.

5.7.2.1.2 Stereoscopic File and Sequence Naming

Stereoscopic file and sequence naming can be accommodated with further conventions. If these conventions are not used, then standard naming conventions can be used to define a single-image stream from which left/right images are automatically created by extracting alternating pixels from each image in the sequence.

To provide stereoscopic images, the further convention is to append an 'L' and 'R" to the file name or file sequence and specify the left image file name in the command. Note this convention applies only to image files and binary files and not currently to video files.

For example, a stereoscopic video sequence can be specified with "File Name" set to "c:\frame1L.bmp" and uses the files:

1) c:\frame1L.bmp, c:\frame1R.bmp
2) c:\frame2L.bmp, c:\frame2R.bmp
3) c:\frame3L.bmp, c:\frame3R.bmp
4) c:\frame4L.bmp, c:\frame4R.bmp

5.7.2.1.3 Test Pattern Naming

Special file names can be used to generate test patterns rather than specifying an image file name. Simple solid color frames, basic ramps, and grids can be generated using special file name conventions.

In order to aid usage, press Ctl-t to bring up the Text Pattern Syntax dialog. Figure 8 shows this dialog which describes the naming convention used to generate different test patterns.

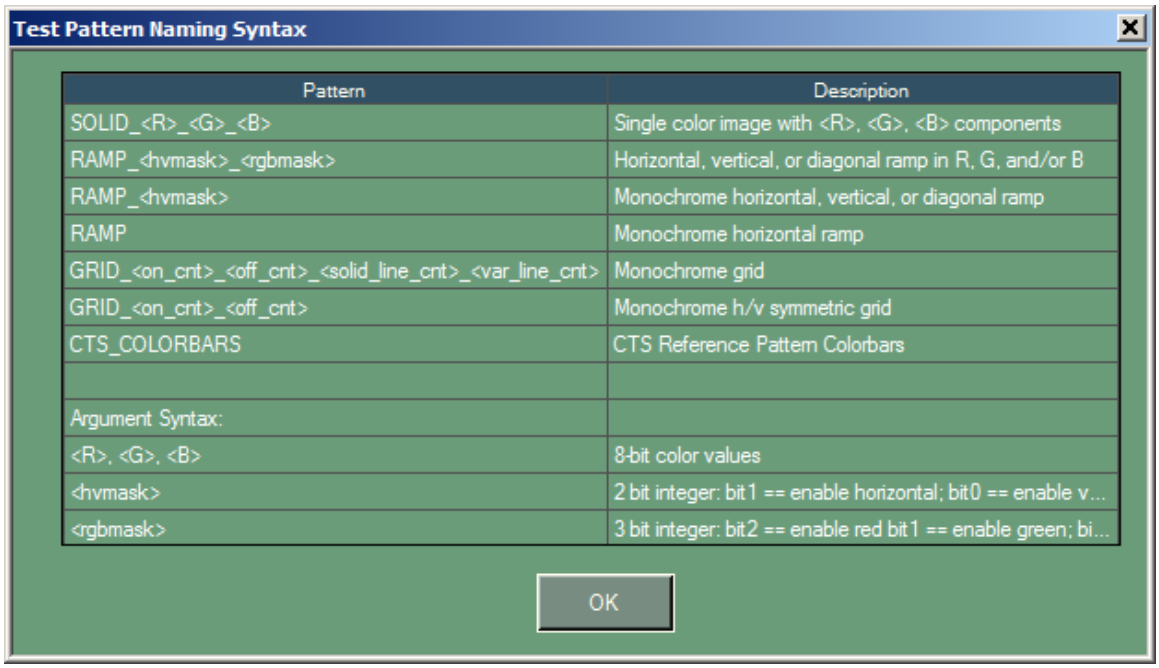

Figure 8 – Test Pattern Syntax Dialog

A single, solid color frame is generated with the syntax:

SOLID_<R>_<G>_<B>.

For example, to generate a full-frame consisting of the single color with red = 100, green = 150, and blue = 200 (8-bit values), enter "solid_100_150_200" for the "File Name" argument of the video-mode or Write Memory command.

A horizontal, vertical, or diagonal ramp can be generated with the syntax:

RAMP_<hvmask>_<rgbmask>.

The <hvmask> value, if present, can be 1 (vertical), 2 (horizontal) or 3 (diagonal). The <rgbmask> value, if present, specifies which color planes are active. For example, RAMP_3_4 specifies a diagonal ramp in red-only and RAMP_2_3 specifies a horizontal ramp in both green and blue.

A monochrome grid pattern can be generated with the syntax:

GRID_<on_cnt>_<off_cnt>_<solid_line_cnt> _<var_line_cnt>

The grid pattern consists of two types of lines: solid lines and variable lines. Variable lines are defined by the user with <on_cnt> and <off_cnt>, basically defining the repeating pattern of "on" pixels followed by "off" pixels. Then the test pattern is formed

by a repeating pattern of solid lines for <solid_line_cnt> followed by variable lines for <var_line_cnt>.

For example, <on_cnt> = 1 and <off cnt> = 9 defines a variable line that has a single white pixel every 10 pixels. If <solid_line_cnt> = 1 and <var_line_cnt> = 9, then the naming syntax is GRID_1_9_1_9 and the pattern formed is a grid with single line thickness with 10 x 10 squares. GRID_5_15_5_15 defines a grid with 5-pixel line thickness with 20 x 20 squares.

Finally, the CTS Reference Pattern Colorbars can be generated using the test pattern name:

CTS_COLORBARS

This test pattern is described in the MIPI CTS for DPhy document in Appendix D.

5.7.2.2 Video Error Syntax

The "Video Error" argument allows the user to encode a bit position in the video frame(s) to force a bit error in the frame data during output. The argument is encoded as follows:

- Video Error bits [31:16] indicates the error line in the image (ones-based)
- Video Error bits [15:0] indicates the error bit position in the line (zero-based)
- It is set to 0 for no error insertion.

For example, to cause a bit error in the most-significant bit of pixel 10 of line 10 in a CSI Raw 10 image, this corresponds to bit position 10 * 10 + 9 = 109 = 0x6D. Thus, set:

Video Error = A006Dh.

5.7.2.3 Image Viewing Dialog

An image viewer is built into the the application to view image files used in video and DSI WriteMemory commands. The viewer can be invoked by pressing Alt-i, causing the image file (if present) of the current command to be displayed in a separate image window. Note that this also applies if the filename is the name of a test-pattern, which is a convenient way to see that the pattern name syntax is correct. If the image size is large, it is re-scaled to a reasonable viewing size. Press Return or click OK to close the image window.

5.7.3 Frame Timing Dialog

The "Timing Cfg…" button in the main window brings up a dialog to set video-mode frame timing settings. These settings select video-mode protocol options and define active and blanking parameters of video-mode playback. Please note that this dialog does not configure DSI video frames sent via command-mode (i.e. using the Write Memory commands) *except* if an image file is specified and the "Allow Image Rescaling" option is enabled. In this case, the input image is rescaled according to HActive by VActive before sending.

Figure 9 shows the CSI layout of the Frame Timing dialog:

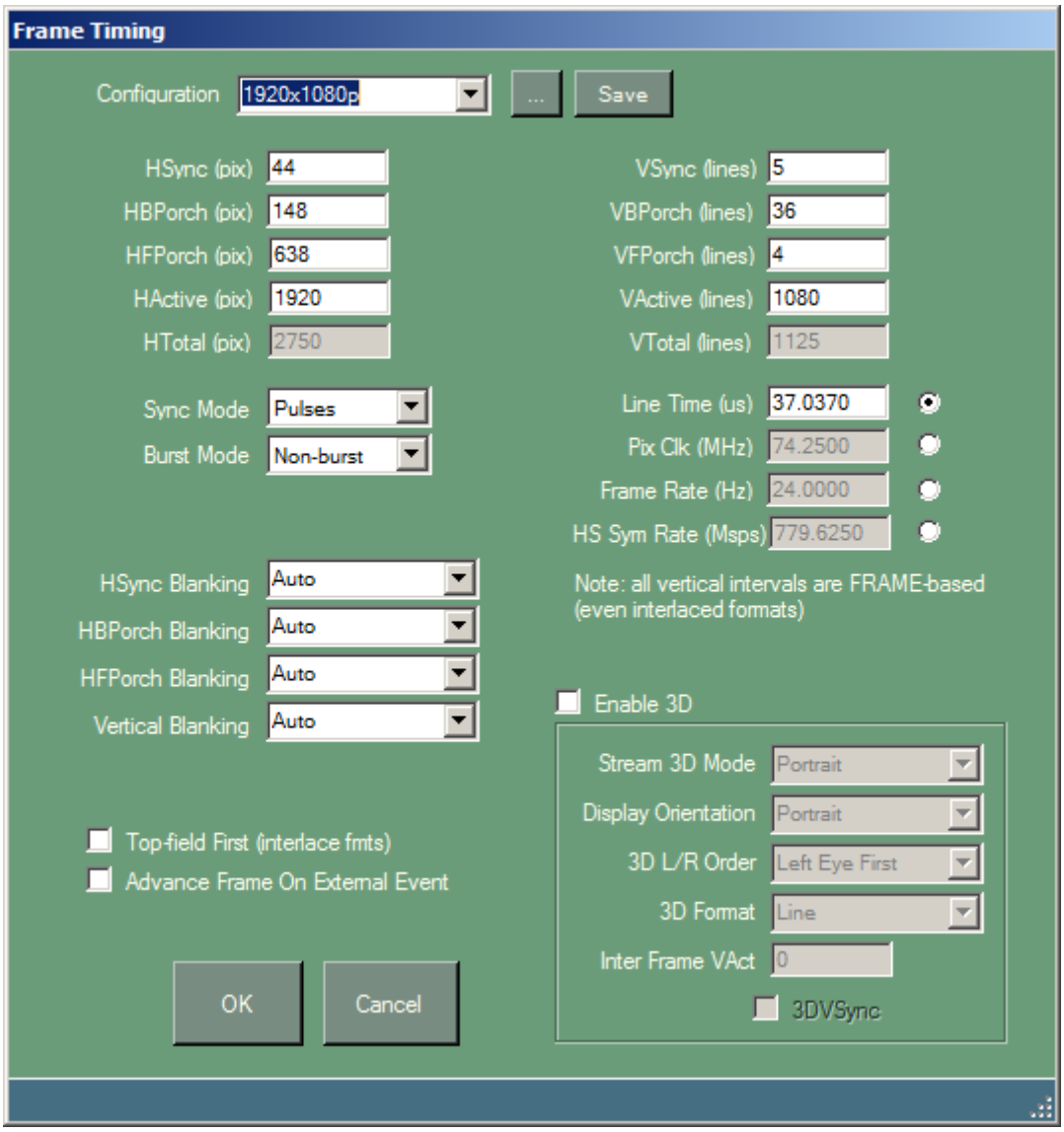

Figure 9 – Frame Timing Dialog (CSI)

Depending on the current MIPI protocol (DSI or CSI), certain fields have different names or are disabled. In particular, CSI:

- Has options "Use LS/LE", "Frame Numbering", and "Line Numbering"
- Disables front-porch and back-porch timings
- Disables 3D stream construction fields and "Top-field First" interlace option.

And, alternatively, DSI:

- Has options "Sync Mode", and "Burst Mode"

- Enables front-porch and back-porch timings
- Enables 3D stream construction fields and "Top-field First" interlace option

5.7.3.1 Frame Timing Configurations

At the top-most part of the Frame Timing dialog is a configuration drop-down control along with two buttons, labeled '…' and 'Save'. These controls are used for named configuration management as follows:

Configuration: drop-down control showing pre-defined and user-defined configuration names. Selecting a configuration name fills in its definition in the remaining frame timing dialog controls.

Pre-defined configurations contain example settings for common frame dimensions. However, please be aware that these configurations may not work for all combinations video formats and lane counts, because of legal timings depend significantly on these parameters.

…: button to launch a List dialog to manage named frame timing configurations (i.e. rename, reset, reorder, delete, etc).

Save: button to save the current control settings as a named frame timing configuration.

5.7.3.2 Frame Parameter Controls

Controls near the top part of the window specify horizontal and vertical blanking component dimensions as well as the active dimensions of the frame. Read-only H and V totals are automatically computed as values are entered. For CSI, the back porch and front porch controls are disabled.

For CSI, note that the HSync blanking period must be long enough to account for all packet overhead (i.e. SOT/EOT and packet headers) in the line. For DSI, each blanking period must be long enough to account for each individual blanking segment overhead, with front porch additionally accounting for tactive packet overhead.

If the blanking settings are insuffiicient, an error message will be displayed when a video command is sent. The message will also estimate the additional number of pixel needed for HSync to make the video frame implementable, e.g.

"Error: HSync is too short to implement LP11 blanking (need approx. 17 more pixels)".

5.7.3.3 Frame Timing Controls

Controls in the center-right of the window let you specify the pixel clock timing in four different ways, via: line time duration, pixel clock frequency, frame rate, or HS symbol rate (if enabled). Entering a value in one of these four controls automatically updates the other three.

The HS Sym Rate text box sets/indicates the lane symbol rate for the current timing configuration. This value will be dependent on the current timing settings, the current

packet type setting in the main window (indicating the video format) and the lane count. If the current packet type is not a video command, HS Sym Rate displays "N/A" and the text box is disabled. Otherwise, the text box is enabled, allowing the user to enter a symbol rate. Note that if the Lane Count is subsequently changed, the HS symbol rate changes along with it (keeping the line time, pixel clock, and frame rate constant).

5.7.3.4 CSI Frame Construction Controls

There are three frame construction options for CSI:

Use LS/LE: drop-down control selecting Line Start/Line End packet usage:

- **Yes**: Line Start and Line End packets are inserted into the video stream and bracket each active video packet.
- **No**: Line Start and Line End packets are not inserted into the video stream.

Frame Numbering: drop-down control selecting Frame Numbering usage:

- **0**: the frame number parameter in the Frame Start and Frame End packets is set to zero.
- **1,2,3**…: the frame number parameter in the Frame Start and Frame End packets is set, starting with 1 and incrementing each frame.

  Note that the frame number only increments for each unique frame in the sequence and repeats when looping. For example, if only a single-frame sequence is sent, the repeating frame always has frame number equal to '1'.

Line Numbering: drop-down control selecting Line Numbering usage:

- **0**: the line number parameter in the Line Start and Line End packets is set to zero.
- **1,2,3…**: the line number parameter in the Line Start and Line End packets is set, starting with 1 and incrementing each line.

5.7.3.5 DSI Frame Construction Controls

There are three frame construction options for DSI:

HSync Mode: drop-down control selecting the HSync Mode:

- **Events**: horizontal syncs in active lines of video frames are conveyed with only an HSync Start packet
- **Pulses**: horizontal syncs in active lines of video frames are conveyed with an HSync Start packet, followed by blanking time, and then an HSync End packet. Note this option is only available in Non-Burst mode.

Burst Mode: drop-down control selecting the burst mode:

- **Non-Burst**: non-burst mode assumes the DSI clock is just sufficient to transmit active lines with no extra bandwidth. When this option is selected, the HS Bit Rate control is made read-only in the frame dialog and automatically set according to frame timing settings.
- **Burst**: burst mode assumes the clock is set faster than necessary to transmit the video frame. Currently, the excess bandwidth associated with each line is treated as part of horizontal front-porch blanking.

Top-Field-First (interlace fmts): selecting this option causes the causes the top-field (i.e. lines 1, 3, 5…) of the image to be sent before the bottom-field (i.e. lines 2,4,6…) when sending interlaced video. Otherwise, the bottom-field is sent before the top-field.

5.7.3.6 Common Frame Construction Controls

Common frame construction options for both CSI and DSI indicate the method to be used for blanking segment implementation:

HSync Blanking: this control allows the user to specify whether to implement the HSync blanking period via LP111 or an HS blanking packet. A third option is "Auto" which implements the blanking segments via LP111 unless it is too short in which case the segment is implemented with an HS blanking packet.

HBPorch Blanking: this control (disabled for CSI) allows the user to specify whether to implement the HBackPorch blanking period via LP111 or an HS blanking packet. A third option is "Auto" which implements the blanking segments via LP111 unless it is too short in which case the segment is implemented with an HS blanking packet.

HFPorch Blanking this control (disabled for CSI) allows the user to specify whether to implement the HFrontPorch blanking period via LP11 or an HS blanking packet. A third option is "Auto" which implements the blanking segments via LP111 unless it is too short in which case the segment is implemented with an HS blanking packet.

Vertical Blanking this control allows the user to specify whether to implement the blanking periods during Vertical blanking as LP111 or HS blanking packets. A third option is "Auto" which implements the blanking segments via LP111 unless it is too short in which case the segment is implemented with na HS blanking packet. Note that vertical blanking lines in DSI have one blanking period in "Event" mode and two blanking periods in "Pulse" mode (with the HSync End packet in between them).

Advance Frame On External Event: this checkbox specifies if each frame in a video sequence should loop individually and only advance to the next frame in the sequence when a rising edge is detected on the external event line of the instrument (GPI0 of the back-panel connector).

5.7.3.7 Stereoscopic Frame Controls

The DSI 1.2 specification introduces a new option for sending stereoscopic frames based on the MIPI SDF specification. Stereoscopic frame construction is supported via controls in lower-right corner of the Frame Timing dialog and, for the most part, correspond directly to parameters fields of the SDF specification.

Enable 3D: when checked, this checkbox enables 3D stereoscopic video mode, and enables the remaining stereoscopic configuration controls. As according to the SDF specification, when 3D mode is enabled, the VSync Start packet is sent with a parameter byte that indicates the format of the 3D frame.

Stream 3D Mode: this drop-down control selects whether video-mode frames should be considered to be in portrait or landscape mode.

Display Orientation: this drop-down control selects whether the display is operating in portrait or landscape mode.

3D L/R Order: this drop-down control selects whether data is sent left-eye first or right-eye first.

3D Format: this drop-down control determines how left and right images are to be interleaved (pixel, line, or frame).

3D VSync: this checkbox determines whether a second VSync is sent between left and right images when 3D Format is set to Frame mode.

Inter Frame VAct: the text box accepts an integer value that determines how many lines should be sent between left and right images when 3D Format is set to Frame mode and 3D VSync is not checked.

5.7.4 Video Frame Construction Details

In contrast to relatively simple non-video-mode commands, video-mode commands define a complicated program stream of LP signaling and HS packets.

In CSI, a frame consists of the following:

- (VSync – 1) blanking lines
- First active line with Frame Start packet
- (VActive – 1) remaining active lines
- Blanking line with Frame End packet

All lines are of equal length in time; thus the lines with Frame Start and Frame End have shorter real horizontal blanking to account for the timing difference.

In DSI, a frame consists of the following:

- Blanking line with VSync Start packet
- (VSync – 1) blanking lines
- **Pulse Mode**: Blanking line with VSync End packet
- **Event Mode**: Blanking line with normal HSync Start packet
- (VBPorch – 1) blanking lines
- VActive active lines
- VFPorch blanking lines

A blanking segment can be implemented by HS blanking packet or a round-trip transition to LP11 for the required duration. Blanking segments set to "Auto" that are long enough to incur the overhead of the round-trip transition to LP11 use this method. Otherwise, the interval is implemented with a blanking packet. Of course, blanking packets also have overhead, and so segments must be at least long enough to accommodate the minimum length blanking packet.

While the desired timing characteristics of a video line are defined in the frame timing dialog, actual timings may slightly differ. This is because of the complexity of constructing arbitrary packet timing given internal restrictions for storing and playing LP states and HS bits in the instrument. As a result, minor quantization of blanking segments may result.

5.7.4.1 Stereoscopic Frame Construction

The 3D stereoscopic standard impacts only the payload content of video-mode frames. Thus, the timing, dimensions, and structure of 3D frames are identical to non-3D frames. Regardless of 3D parameter settings, output frames always have dimension HActive by VActive. Input images are always rescaled if necessary during output frame construction.

Video-mode commands sent when stereoscopic video is enabled (i.e. Enable 3D checked) cause 3D frames to be constructed and sent. If the "File Name" argument references a single image or image sequence (see section 5.7.2.1), then each single image is used to build a pseudo-stereoscopic image pair by extracting alternating left/right pixels, rescaling if necessary, and then building the composite output image based on the SDF specification.

If the "File Name" argument references a true stereoscopic image or image sequence, then each image pair is imported, rescaled if necessary, and then used to build the composite output image.

Note that in the case where 3D Format is set to Frame and 3DVSync is checked, 2 * "Frame Count" images are sent in the output.

5.7.4.2 Command Insertion

Note that video frames are automatically built with a command insertion point in the first vertical blanking line. Command insertion is a special mode that allows most commands

to be inserted into the program stream *while the program is looping.* This function is also available for the user during macro looping using the "Cmd Insertion Point" command. Please see section 5.12.3 for more information.

5.7.4.3 Video Memory Requirements

Memory requirements for the storage of video sequences in the CPhy Generator cannot be exactly described, due to the complexity of encoding, packing, and outputting the program stream as well as numerous dependencies such as number of lanes, video format, frame timing, and CPhy bus timing. However, in general, a reasonable upper bound can be given, based on HS symbol packing in the instrument's memory.

Basically, the packing efficiency of HS symbols in memory is 75%, regardless of the number of lanes. Since the amount of memory in the CPhy Generator is 2 GB, this means a maximum of 1.5 GB of memory could be available for video symbol data.

CPhy maps 2 bytes to 7 symbols, or equivalently 21 bits of memory, implying a (16/21) conversion ratio. Thus, 1.5 GB of memory could store about 1.14 GB of video payload bytes.

Max Frames = 1.14 GB / ((HActive * VActive * BytesPerPixel)

A more accurate way to determine how many frames will fit into memory with your frame timing settings is to send a small sequence consisting of, say, 10 or 20 frames. Once running, the status readout will indicate how much memory of the PG is used, as a percent of the total available.

Note that, currently, the CPhyGenCtl application does not manage host memory with great efficiency and cannot generally achieve the maximum frame count supported by the CPhy Generator (an out-of-memory error occurs when building the sequence). If this is a concern, please contact The Moving Pixel Company for more information. To improve the maximum frame sequence count, please ensure your host computer has a 64-bit processor, with at least 8 GB of memory, and a large page-file configuration.

5.7.5 DSC Compression

Extensive Display Stream Compression (DSC) support is provided with the DSC License option (please contact the Moving Pixel Company for licensing information). Without the licensing option, "Compressed Pixel Stream" and "Picture Parameter Set" commands can still be sent, but only accept binary file input for their "File Name" parameter. This section documents the additional features supported with the DSC License.

5.7.5.1 DSC Configuration Dialog

The DSC Configuration dialog sets all parameters related to DSC encoding and is brought up using the "Configure DSC" menu item (Options menu). As with other configurations (frame timing and CPhy timing), DSC configurations can be specified and saved, allow for easy selection in the future.

A summary of dialog controls follows (see Figure **10**). Please note that dialog settings always override any equivalent settings in the configuration file.

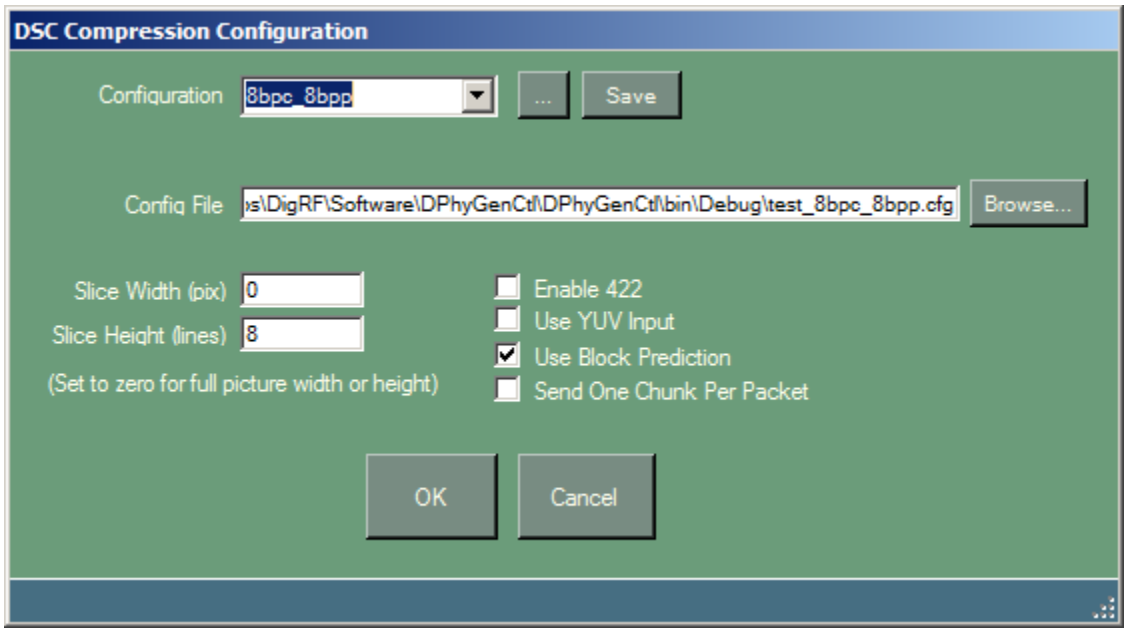

Figure 10 – DSC Configuration Dialog

- **Configuration**: drop-down control to select a named DSC configuration. When selected, the parameters of the configuration fill in the other dialog controls.
- **"…"**: button next to the Configuration drop-down control to bring up a List dialog to order, rename, and delete named configurations.
- **Save**: button to save the current control settings to a named configuration. A name dialog is provided for the user to enter in a configuration name. If the name exists, the user is asked whether it is okay to overwrite the existing configuration.
- **Config File**: file name of a DSC configuration text file. The syntax of this file is beyond the scope of this document (please see the DSC reference encoder documents). However, it should be noted the LINE_BUFFER_BPC is set to BITS_PER_PIXEL + 1 by default, but can be overridden if set in the configuration file. Standard configuration files are located in the application directory and are already pre-defined in named configurations for use in CPhyGenCtl.
- **Browse**: button to bring up a file dialog to select a configuration file.
- **Slice Width**: text box to set the slice width. Set to 0 for the entire picture width.
- **Slice Height**: text box to set the slice height. Set to 0 to use the entire picture height.
- **Enable 422**: checkbox to enable 422 color component sampling.
- **Use YUV Input**: checkbox to use the YUV color space.
- **Use Block Prediction**: checkbox to enable block prediction.

- **Send One Chunk Per Packet**: checkbox selecting how chunk data is to be sent, both for Write Memory commands and video-mode commands. If checked, one chunk will be sent per packet. Otherwise, one or more chunks representing an image line will be sent in a single packet.

5.7.5.2 Sending Compressed Frames In Video Mode

The "Compressed Pixel Stream" packet type is used to send one or more frames of compressed video. This command is very similar to other video-mode packet types, accepting the same arguments and implementing the frame timing as configured by user settings. The biggest difference is that image frames are first compressed (if not already compressed) using the current DSC configuration settings before the output video-mode frame is built.

Notes:

- In addition to the standard image, video, test pattern, and binary file types, the "File Name" argument has been extended to support DSC and DPX file types for this command. If a DSC file is provided, chunk data is sent directly as "Compressed Pixel Stream" packet payload. If a DPX file is provided, it is first compressed (like other non-binary file types) before used as packet payload.[13]
- Like other video-mode packet types, setting Frame Count greater than one causes a multi-frame sequence to be sent. Multi-file naming syntax can be used to specify the sequence of source image files to use.
- The "Video Error" parameter also is supported, with its value still encoding the line and error bit location. Note that the bit location indicates the error bit in the *compressed source data* and so can only affect payload data (not the packet header or CRC). This applies regardless of how many packets are used to send a line (i.e. independent of the "Send One Chunk Per Packet" setting).

5.7.5.3 Sending Compressed Frames In Command Mode

As with uncompressed images, DSC compressed video frames can be sent using the DCS Write Memory commands. This is achieved by selecting the "DSC Compressed" option for "Image Decode Format" in the Write Memory Configuration dialog and sending a Write Memory command.

Command arguments and behavior are the same, with the following extensions and clarifications:

- In addition to the standard image, test pattern, and binary file types, the "File Name" argument has been extended to support DSC and DPX file types for this command. If a DSC file is provided, chunk data is sent directly as Write Memory packet payload. If a DPX file is provided, it is first compressed (like other non-binary file types) before used as packet payload.

[13] Note that the DPX file type does not support automatic rescaling even if "Allow Image Rescaling" is enabled. Thus, the file must already have the correct output frame dimensions.

- The "Partition Length" and "Quantize Packets to Multiple of Line Length" settings in the Write Memory configuration are ignored. The payload length is always either one chunk or one line's chunk data, based on the setting of the DSC configuration option "Send One Chunk Per Packet".

Please see the more detailed description of operation in section 5.6.6.9.1.

5.7.5.4 Sending PPS Information

The Picture Parameter Set command conveys DSC compression information to the DUT in advance of sending compressed video packets. Without the DSC license option, only binary files can be sent with this command.

With the DSC license option, the user can also specify one of the following as the "File Name" argument:

- DSC compressed file name. In this case, the PPS structure in the file is sent as payload for the command.
- Any valid image or test pattern file name. In this case, a PPS structure is built using the current DSC configuration settings and active frame dimensions in the current frame timing configuration.

5.7.5.5 Compressing/Uncompressing Image Files

With the DSC license option, two keyboard functions are enabled in the main GUI window:

- **Control - C**: Compress the file or test pattern associated with the current "File Name" and output to a DSC file. The user is prompted for an output file name. If "Allow Image Rescaling" is enabled, the input image is first rescaled to the active frame dimensions in the current frame timing configuration.
- **Control - U**: Uncompress the DSC file in the current "File Name" field and output to an image file. The user is prompted for an output file name, which can have any common image extension (e.g. BMP, JPG, DPX).

5.7.5.6 Viewing DSC Images

With the DSC license option, the image viewing keyboard function is extended to view DSC and DPX files. Simply press **Control – I** to view the file when entered into the "File Name" field of the current command.

5.8 Macros

A macro is a sequence of non-video commands, grouped and named as a single, new command. Once defined, they can be assigned to command buttons and sent to the CPhy Generator like any other command. When viewing and creating macros, controls in the Macro tab of the main window is used. These controls displays the component commands of the macro and allows basic macro editing (see **Figure 11** below).

5.8.1 Defining a new macro

To define a new macro, click on the "Start Macro" button at the bottom right of the Macro tab pane. This button will change its caption to "End Macro" and will activate the Macro Cmds list-box with no component commands initially present.[14] In this editing mode (called macro mode), commands that are sent either via a command button, via the Send button at the top of the main window, or via an RPC programmatic command are entered into the macro instead of immediately sent to the CPhy Generator.

Figure 11 shows an example of a macro in edit mode.

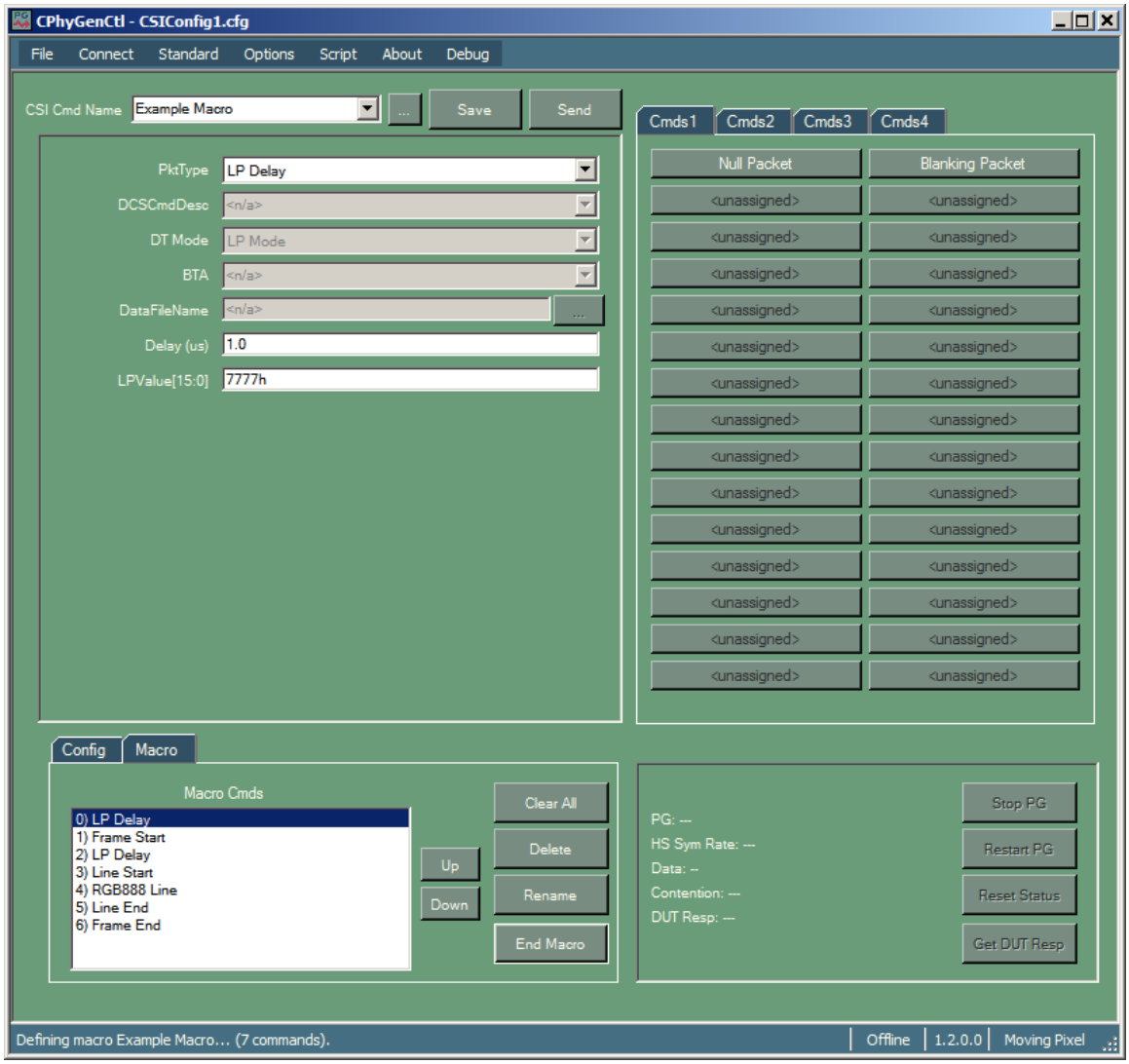

Figure 11 – Macro Editing

Commands are inserted into the macro at the location highlighted in the macro pane. When inserted, a dialog is shown for the user to name the component command of the

[14] Note: if the current Cmd Name is a macro then the list-box will be initialized with the macro's component commands. If unwanted, the Clear All button can be used to delete them.

macro. The user can scroll and select a command (or the blank line at the end of the command list) to mark the next command insertion location. Selecting a command also displays its arguments for viewing and modification.

Five buttons are available in the macro pane to manipulate the component commands of a macro.

- **Clear All** removes all the component commands in the macro.
- **Delete** removes the selected component command from the macro.
- **Rename** brings up a dialog to rename the selected component command
- **Up** moves the selected component command up within the command sequence
- **Down** moves the selected command down within the command sequence.

When macro definition is complete, the user clicks on the "End Macro" button, bringing up a dialog to name the macro, save it, and exit macro mode (the user may also **Cancel** to return to macro editing or **Discard Edits** to exit macro made, discarding all changes to the macro entirely.

As with command definition, if the user specifies the same name when saving a macro, the user will be asked for permission to replace the existing macro definition.

5.8.2 Editing an Existing Macro

To edit and existing macro, the user has two options. If the macro is assigned to a command button, the user may right-click on it and select "Edit Current Command…". Alternatively, the user can select it in the command combo-box at the top of the main window then click on the "Start Macro" button. The macro can then be edited as normal and saved by clicking on the "End Macro" button. The name field will be filled in with the original macro name and the user can click OK and confirm overwriting the existing macro.

5.8.3 Copying a Macro

To copy a macro, simply load a macro for editing, click on the "End Macro" button, and replace the original name with a new name.

5.8.4 Editing a Component Command

To edit a component command of a macro, perform the following:

1) Open a macro for editing.
2) Click on a component command in the macro to retrieve its argument settings.
3) Change the argument settings as desired.
4) Click the 'Send' button to insert the modified command into the macro (both the new command and old command will now be in the macro list)
5) Click the 'Delete' button to delete the old component command.
6) Click the 'End Macro' button and Click 'OK' to save the modified macro.

5.8.5 Allowed Component Commands

The intent of a macro is to output a MIPI command sequence in real-time on the bus. Allowed macro component commands include all standard or custom MIPI packet types (LPDT or HS) accessible via the PktType drop-down control, as well as all Phy commands and Miscellaneous commands except for the "RPC Script" command.

In addition, the following notes apply to macro behavior:

- Component commands with BTA enabled are allowed in macros and behave as expected. The PG waits for a return BTA before continuing with the remaining component commands in the macro.
- Write Memory Start and Write Memory Continue commands are fully supported in macros, including image file decoding and command partitioning to support large "Write Len" counts.
- Macros cannot contain other macros; that is, macro nesting is not allowed. However, macros can nonetheless be added to other macros, in which case the component commands of the macro being added are themselves duplicated and added to the macro being edited.
- Video-mode commands can now be added to macros. Please see the next section for more information.

When a macro is built using an RPC program or script, allowed macro component commands are any command that can be sent via the following RPC commands:

- SEND_MIPI_CMD
- ADD_MIPI_CMD
- SEND_IMPAIRED_MIPI_CMD

In addition, a new feature supported by macros is the ability to contain certain non-MIPI configuration commands, allowing on-the-fly application reconfiguration (not hardware reconfiguration!) for subsequent commands in the macro. For example, certain frame timing or CPhy timing parameters can be changed between two frames defined in the macro.

Generally, the following classes of non-MIPI RPC commands are allowed in macros:

- Frame buffer load and deallocation
- WriteMemory configuration
- CPhy Configuration
- Frame Timing configuration (except for frequency-related settings)
- DSC configuration
- Option setting that applies to program construction
- Bayer encoding type

For more details, please refer to the document CPhyGenCtlRPCCommands.pdf, which indicates which configuration RPC commands are allowed in macros (Note 9).

5.8.6 Video-Mode Commands In Macros

Video-mode command behavior in macros is determined by the setting of the option "Enable Video Mode In Macros" (Options menu). This section describes the two behaviors.

5.8.6.1 "Enable Video Mode In Macros" Enabled

If the option "Enable Video Mode In Macros" is enabled, a video-mode command in a macro causes a full video frame to be sent using the current frame timing configuration. However, note that when video-mode frames are played in macros, the clock rate used is the HS Sym Rate setting for the macro, *not the configured HS Sym Rate in the frame timing configuration*. If accurate frame timing is required, the user must set the HS Sym Rate in the main window to the same as the HS Sym Rate in the frame timing dialog when sending the macro

5.8.6.2 "Enable Video Mode In Macros" Not Enabled

If the option "Enable Video Mode In Macros" is not enabled, only a single video data packet is played in the macro. In this case:

- The entire contents of the given file are put in the video packet (which, to be MIPI legal, must be less than 64 Kbytes)
- The "Frame Count" parameter is ignored
- The file argument is assumed to reference a binary file. That is, image file decoding is not supported.

Note that one additional variation is supported for video commands, though in general, it is accessible only through RPC (either via script or user program). This variation allows the video command to reference an internal frame buffer, previously loaded with a LOAD_FRAME RPC command. In this case, the filename used to reference the frame buffer is USERFRAME<n> where n is 0 through 3 and the two video command arguments refer to a byte offset and byte length in the frame buffer data.

5.8.7 Looping In Macros

The "Start Loop" and "End Loop" commands are used in macros to provide an easy way to duplicate a group of commands. Simply bracket the command group with the loop commands, for example

- …
- Start Loop
- Cmd0
- Cmd1
- Cmd2
- …
- End Loop
- …

With the exception of infinite looping, this construct *does not* implement true looping and thus does not save any memory or construction time. But it is a useful convenience, especially for video frame construction, where a command sequence that implements one video line can be looped multiple times. Be aware that large loop counts can easily cause out-of-memory errors and so should be used with care (infinite loops do not consume additional memory).

The "Start Loop" command has one argument, "Loop Count", which indicates the number of times to loop the command group between start and end. A value of 0 indicates infinite looping, which causes the macro program to be constructed and sent as described in the next paragraph.

Commands that precede the infinite Start Loop are output normally and can be considered like an initialization sequence. Commands contained in the infinite loop are treated like a normal looping macro, outputting indefinitely until the "Stop PG" button is pressed. Note that commands that follow the infinite loop are discarded and never sent.

Generally, nesting of loops is *not* allowed, though multiple loops can occur serially within a macro. One exception is that non-infinite loops can occur within an infinite loop.

5.8.8 HS Component Command Behavior

When DSI macros are parsed by CPhyGenCtl, the application can group consecutive HS component commands (that don't have BTA enabled) into a single HS burst. Whether consecutive HS commands are sent in a single burst or sent individually in their own burst is determined by the setting of the "Send Single Pkt Per HS Burst" option (Options menu).

For mixed burst behavior, the user can disable this option, which will by default group consecutive HS commands into a single burst. Then, for those HS commands to be sent in their own burst, the user can the user can put "LP Delay" commands between them.

5.8.9 Additional Notes on Macro Behavior

Below lists some final miscellaneous notes on CPhyGenCtl behavior when working with macros:

- As macros are commands, macro names must be distinct from command names.
- On the other hand, component command names in macros are simply descriptive names for the component command and have no relation to command names in the application. Component command names do not need to be unique to each other or to command names in the application.
- Once component commands are added to a macro, they are duplicated and no longer are associated with the original command. Specifically, if the original

command is then modified and saved, the component command in the macro having the same name does NOT change.

- When building a macro via script or RPC, component commands are added to the macro when using the RPC commands SEND_MIPI_CMD or ADD_MIPI_CMD.
- Also, when building a macro via script or RPC, sending a configuration command that is explicitly allowed in a macro (see section 5.8.5) adds the command to the macro. On the other hand, sending a configuration command that is not explicitly allowed in a macro is instead immediately executed and NOT added to the macro.

5.9 Phy Commands

Several commands under the "Phy Commands" category invoke lower-level test modes of the CPhy Generator instrument. These consist of the following commands, many of which implement a superset of the CPhy TGR and PRBS test specifications:

- Escape Command Sequences (e.g. BTA, ULPS, etc)
- TGR Data Sequence
- TGR Symbol Sequence
- TGR State Sequence
- PRBS9 Sequence
- PRBS11 Sequence
- PRBS18 Sequence

These commands are described in the following sections.

5.9.1 ULPS

The ULPS command causes the designated lanes to enter the ULPS state. The LaneMask[3:0] argument indicates which lanes are to transition to ULPS (with bit 0 representing lane 0, bit 1 representing lane 1, etc.).
This command implicitly embeds a Wait Host Event command following the ULPS entry sequence, which waits for an event signal that can be generated by the user using the "Host Event" button of the main window. Behavior is as follows:

When the command is sent, the PG outputs the ULPS entry sequence on selected lanes and continues to run as the instrument holds lanes indefinitely in ULPS. During this time, status indicates that the PG is "Waiting Host Event". Until this event is seen (or the PG stopped), lanes remain in ULPS. When the host event occurs, the PG outputs the ULPS exit sequence on selected lanes and returns to LP111.

Note that this command can be looped like most commands. If the "Loop Commands" option is checked, then the ULPS command will repeat after the Host Event occurs. From the user's perspective, in this case, nothing seems to change as the PG status remains static: the PG continues to run and the "Waiting Host Event" message is still displayed.

5.9.2 BTA

BTA (bus turn-around) is a handshake protocol to transfer bus ownership to the DUT. This protocol is only applicable in DSI. Read commands by default have their BTA argument set to "Yes" and this argument can be set to "Yes" by the user for other commands as well. When BTA is set, the BTA signaling sequence is appended to the command. In addition, a BTA sequence can also be sent on its own, without accompanying data by selecting "Send BTA" in the PktType field of the command.

After a BTA sequence is sent, the CPhy Generator suspends program output and places its drivers in high-impedance until it receives a return BTA sequence from the slave DUT or times out. When a return BTA sequence is seen on the bus, the instrument drives a BTA acknowledgement sequence and then resumes program output. In the event of a BTA timeout, the instrument resumes program output without sending a BTA acknowledge.

This wait-for-BTA process can be initiated without first sending an initial BTA sequence by using the "Wait For BTA" command. This command does not transmit any data on the bus but simply drives high-impedance and waits for a BTA sequence from the slave DUT or eventually timing out.

Notes:

- These commands are most useful when embedded in a macro with other MIPI commands. Component commands in the macro following a "Send BTA" or "Wait For BTA" are not output until the return BTA is recognized or the BTA timeout has expired.
- The BTA timeout is set in the Instrument Configuration dialog with the control "BTA Wait Time". To avoid any timeout, the user can set the "Disable Command Timeout" option (Options menu). If necessary, the "Stop PG" button can be used to terminate a command waiting for a return BTA.
- While the CPhy Generator is waiting for a return BTA from the DUT, the main window status bar will display "Waiting on BTA…".

5.9.3 Escape Command

The Escape Command option lets the user send a phy escape command byte. The only argument is the command byte to send. Note that this command is different than a packet sent in LPDT mode. LPDT mode uses an escape command byte of 0x87 to put the receiver in LPDT mode, and then sends the packet bytes.

5.9.4 TGR Data Sequence

The TGR function has been adopted from CPhy to provide low-level support for arbitrary HS data transmission, which can be useful for general testing.

The TGR Data Sequence command starts by sending an HS burst entry sequence on all active lanes using the current CPhy timing settings. The only difference from a normal HS burst entry sequence is that the Start Preamble sequence is replicated by the "TGR

Pre Length" setting plus one, as defined by the CPhy specification for TGR and PRBS test modes.

Following the HS burst entry sequence, all active lanes loop on a specific data pattern defined by the command. The pattern loops for the number of iterations specified by the LoopCnt parameter, eventually exiting back to LP11 with an HS burst exit sequence. The only difference from a normal HS burst exit sequence is that the Postamble sequence is replicated by the "TGR Post Length" setting plus one, as defined by the CPhy specification for TGR and PRBS test modes.

If LoopCnt is set to zero, the burst data loops indefinitely in HS mode, never completing a single burst until the "Stop PG" button is pressed (at which point the HS burst exit sequence is output and the PG becomes idle). Otherwise, after one complete burst has been sent, the burst is repeated again and again, only stopping when the "Stop PG" button is pressed. Note that this outer looping occurs even if the "Loop Commands" option is unchecked.

Note that one constraint on the LoopCnt parameter is that the total number of looped bytes in the sequence (i.e. LoopCnt * sequence bytes) must be greater than or equal to 64. Otherwise, an error message is displayed.

The TGR Data Sequence has one command argument (Params[]) that accepts a variable-length list of hex bytes. The CPhy specification describes the TGR pattern as consisting of exactly 2 bytes, but CPhyGenCtl extends this implementation, supporting an arbitrary list length as long as it contains an even number of bytes.

On the CPhy bus, looping TGR data bytes are pair-wise converted to CPhy symbols, then mapped to driver states as they are output. Thus, the looping list of N bytes is converted to 7 * (N/2) looping CPhy states on the bus.

For example, if Params[] is set to "aa bb cc dd", the following symbol sequence will be looped on all active lanes:

4, 2, 2, 4, 2, 2, 3, 0, 3, 4, 4, 0, 3, 1

5.9.5 TGR Symbol Sequence

The TGR Symbol Sequence is not described by the CPhy specification but is a variation of the TGR Data Sequence concept with nearly identical bus behavior. Like the TGR Data Sequence command, there is one Params[] command argument, only instead of data bytes, the user specifies CPhy symbols. There are no length restrictions on the number of symbols in the Params[] command argument.

For example, if Params[] is set to "4 4 4 4 4 7", the following state sequence might be output (depending on encoder state):

6, 1, 6, 1, 6, 6, 1, 6, 1, 6, 1, 1

5.9.6 TGR State Sequence

The TGR State Sequence is not described by the CPhy specification but is a variation of the TGR Data Sequence concept with nearly identical bus behavior. Like the TGR Data Sequence command, there is one Params[] command argument, only instead of data bytes, the user specifies CPhy states. There are no length restrictions on the number of states in the Params[] command argument.

For example, if Params[] is set to "1 6 2 5 4", this five state sequence will be output repeatedly on the CPhy bus.

5.9.7 PRBS9 Sequence

The PRBS9 Sequence is defined in the CPhy specification and is a variation of the TGR Data Sequence concept with nearly identical bus behavior. However, unlike the TGR sequence where data is specified by the user, PRBS logic automatically generates 16-bit data words to be encoded into symbols, then states, and sent on the CPhy bus. Implementing a superset of the CPhy requirement, each lane runs its own PRBS generator independently and generates a sequence whose length is specified by the user using the WordCnt parameter.

If WordCnt is set to zero, the burst data loops indefinitely in HS mode, never completing a single burst until the "Stop PG" button is pressed (at which point the HS burst exit sequence is output and the PG becomes idle). Otherwise, after one full burst has been sent, the command has completed, but like other commands may optionally be looped either in a macro (using Loop Start and Loop End commands) or as an individual command using the Option->Loop commands menu option).

In addition to the WordCnt parameter, the PRBS9 command accepts four more arguments, initializing each lane's PRBS generator.[15] Optionally, the user can specify identical seed values causing all active lanes to have identically synchronized data patterns:

- L0_Seed[8:0] – lane 0 starting PRBS seed (should be nonzero)
- L1_Seed[8:0] – lane 1 starting PRBS seed (should be nonzero)
- L2_Seed[8:0] – lane 2 starting PRBS seed (should be nonzero)
- L3_Seed[8:0] – lane 3 starting PRBS seed (should be nonzero)

5.9.8 PRBS11 Sequence

The PRBS11 Sequence is nearly identical to the PRBS9 Sequence (see previous section) except that the PRBS algorithm is generated from an 11-bit LFSR. Thus, the initial seed arguments are 11-bit values:

- L0_Seed[10:0] – lane 0 starting PRBS seed (should be nonzero)
- L1_Seed[10:0] – lane 1 starting PRBS seed (should be nonzero)
- L2_Seed[10:0] – lane 2 starting PRBS seed (should be nonzero)

[15] Note that inactive lanes remain in LP111 and their seed values have no effect on output.

- L3_Seed[10:0] – lane 3 starting PRBS seed (should be nonzero)

5.9.9 PRBS18 Sequence

The PRBS18 Sequence is nearly identical to the PRBS9 and PRBS11 Sequence (see previous sections) except that the PRBS algorithm is generated from an 18-bit LFSR. Thus, the initial seed arguments are 18-bit values:

- L0_Seed[17:0] – lane 0 starting PRBS seed (should be nonzero)
- L1_Seed[17:0] – lane 1 starting PRBS seed (should be nonzero)
- L2_Seed[17:0] – lane 2 starting PRBS seed (should be nonzero)
- L3_Seed[17:0] – lane 3 starting PRBS seed (should be nonzero)

5.10 File Command

The File Command is general-purpose, low-level command that uses a text file as input to define arbitrary LP and HS CPhy signals. A special syntax is used to allow signal definition for CPhy lanes, with support for higher-level protocol segments, including automatic CRC generation, automated HS burst segment generation (e.g. HS entry/exit, preamble/postamble and sync), and packet data demultiplexing. Raw CPhy state sequences, CPhy symbol sequences, and packet data bytes can be intermingled as desired to create user test sequences and error packets.

The File Command can be found under the "Miscellaneous Commands" category in the PktType drop-down of the GUI and has one argument ("File Name") that specifies the source text file name to parse and use to construct the CPhy signal. The command is sent like any other normal command and can be part of a macro. The DT Mode is ignored when sending the File Command, but other GUI settings such as the HS Sym Rate, LP Freq, Lane Cnt, and CPhy bus timings may affect the output, depending on the component commands in the file.

The file used for the file command is a form of script file. Each line can be one of the following:

1. **Command line** – A command line always begins with the '#' symbol, followed by a space, followed by the command and any arguments, separated by spaces. The command and arguments always use only one line.
2. **Data line** – A data line consists of one or more hex data values, separated by spaces.
    - Multiple data lines can be used to form a longer sequence of data values.
    - Concatination of data lines occurs to form one data sequence until the next command line (or end of file).
    - A data sequence is always associated with the preceding command.
    - Depending on the associated command, data values may represent data bytes, HS symbols, HS states, single-lane LP states, or 4-lane LP states.
3. **Comment line (or blank line)**: Lines that start with "//" or that are blank (with only spaces or tabs) are considered comment lines and are ignored by the parser.

In constructing the bus output stream, the file associated with the File Command is parsed in sequence: one line at a time, one component command at a time. After parsing the entire file, the result is a single output stream that is then sent on to the CPhy bus, looping if enabled.

If the File Command is sent stand-alone (i.e. not in a macro), it should assume the initial bus state is LP111 in all lanes. Similarly, after the command is sent, the bus is forced back to LP111. In a macro, the File Command begins at the last state occurring in the previous component command in the macro, which generally will be LP111. It is possible, however, for a previous component command in a macro to finish in a bus state other than LP111, for example, from a second File Command.

Note that certain degenerate patterns may not be able to be implemented by the CPhy Generator as there are structural requirements on the program stream it can output. Thus, it is possible for the stream described in the File command to generate an error during construction if it fails these requirements. The error message will say

"Program cannot be sent by hardware at the current HS Sym Rate".

While exact requirements are too complicated to document, failure is always due to too few symbols/states on average in constructed output records relative to the HS Sym Rate throughput requirements. Generally, the work-around is to add more states, symbols, or bytes to the stream or decrease the HS Sym Rate.

The next sections describe component commands and data.

5.10.1 File Command File Syntax

Table 7 – File Command File Syntax

Command	Description
HS_BYTES <lgwd>	Sends one or more HS packet bytes, allocated to lanes according to the argument.
HS_BYTES_PLUS_CRC <lgwd>	Sends one or more HS packet bytes, automatically appending a 2-byte CRC to the stream. Bytes are allocated to lanes according to the argument.
HS_SYMBOLS <lg>	Sends one or more HS CPhy symbols. Symbols are assigned to one lane or replicated across active lanes according to the argument.
HS_STATES <lg>	Sends one or more HS CPhy states. States are assigned to one specific lane or replicated across active lanes according to the argument.
LP_STATES [ACT] [<dur>]	Sends one or more LP states. State values

	either specify all lane states at once or states are replicated across active lanes according to whether "ACT" is present.
LP_ESC_BYTES	Sends one or more data bytes in LP escape mode using spaced-one-hot encoding.
HS_BURST_ENTRY	Sends the HS burst entry LP sequence on active lanes using the current CPhy timing settings. No element stream is associated with this command.
PREAMBLE	Sends the currently defined preamble sequence on active lanes. No element stream is associated with this command.
SYNC	Sends the currently defined sync sequence on active lanes. No element stream is associated with this command.
SYNC1	Sends the currently defined sync1 sequence on active lanes. No element stream is associated with this command.
SYNC2	Sends the currently defined sync2 sequence on active lanes. No element stream is associated with this command.
SYNC3	Sends the currently defined sync3 sequence on active lanes. No element stream is associated with this command.
POSTAMBLE	Sends the currently defined postamble sequence on active lanes. No element stream is associated with this command.
HS_BURST_EXIT	Sends the HS burst exit LP sequence on active lanes using the current CPhy timing settings. No element stream is associated with this command.
PH	For CSI, sends duplicated PH sequences on each active lane, with sync2 sequence between PH. PHCRC is automatically generated for the PH so exactly 4 bytes should follow this command (i.e. Reserved, DataID, Data0/WC0, Data1/WC1). For DSI, sends duplicated PH sequences demultiplexed across active lanes, with <lane count> sync2 sequences between PH. The PHCRC is automatically generated for the PH so exactly 3 bytes should follow this command (i.e. DataID, Data0/WC0, Data1/WC1).
PAYLOAD	Sends one or more packet data bytes, demultiplexed across active lanes, followed by

	an automatically generated CRC, then EOT.
IF <flag>	In conjunction with ENDIF, allows blocks to be enabled or disabled for parsing. Nesting of the command is allowed.
ENDIF	In conjunction with IF, allows blocks to be enabled or disabled for parsing. Nesting is allowed.
FILE <filename>	Parses the commands in the given text filename, inserting them into the current stream. Nesting of this command is allowed. Individual component commands and their associated data *may not* span multiple files.
RADIX <radix>	Sets the default radix to use for command data that doesn't have an explicit radix ("d", "h", or "b" following value, e.g. 10h, 10111b, etc). For backward compatibility, the default radix is initialized to 16 at start of the File command.
LOOP_START <loop count>	In conjunction with LOOP_END, defines a looping block. Lines in between LOOP_START and LOOP_END are repeated <loop count> times. Can be used to repeat both data and/or command lines. Nesting is allowed, but LOOP_START and corresponding LOOP_END must occur in same file.
LOOP_END	Defines the end of a looping block.
<lg>	Name is shorthand for "lane group". Consists of one of the following symbols: ACT | 0 | 1 | 2 | 3
<lgwd>	Name is shorthand for "lane group with demux". Consists of one of the following symbols: ACT | 0 | 1 | 2 | 3 | DEMUX
<dur>	Integer duration in ns. Value will be quantized to a multiple of the HS Sym Rate and must be greater than or equal to 40 ns.
<flag>	Integer value used for boolean function: 0 == false, 1== true
<filename>	String value containing a file path. If the filename is relative, its path is considered to start from the same directory as the text file containing the command using the argument.
[]	Arguments in brackets are optional.

5.10.2 Lane Group Arguments

This section describes the meaning of the lane group arguments.

ACT

This argument causes the element stream associated with the command to be duplicated and assigned to each active lane.

For example, if lane count is two, the following command assigns the same three-byte stream (AA BB CC) to lanes 0 and 1:

```
# HS_BYTES ACT
AA BB CC
```

DEMUX

This argument causes the byte stream associated with the command to be demultiplexed two bytes at a time across active lanes. Padding bytes of '0' are appended to the stream to align all lanes to the next even byte boundary.

For example, if lane count is three, the following command:

```
# HS_BYTES DEMUX
1 2 3 4 5 6 7
```

assigns bytes to lanes as follows:

```
lane 0: 1 2 7 0
lane 1: 3 4 0 0
lane 2: 5 6 0 0
```

0, 1, 2, 3

These arguments designate a lane assignment for the element stream associated with the command. If the lane is not an active lane, the command is ignored. If this argument form is used, identical commands with the same sequence length must be present in active lane order.

For example, if lane count is four, a legal command sequence would be:

```
# HS_BYTES 0
1 2 3 4
# HS_BYTES 1
a b c d
# HS_BYTES 2
11 12 11 12
# HS_BYTES 3
a3 b2 c1 d0
```

Note this command sequence would also be legal for lane count < 4, where inactive lane commands would be ignored.

5.10.3 LP_STATES Component Command

The LP_STATES command has two forms.

The first form uses the "ACT" argument, which indicates that data values are singular LP states and will be duplicated and applied to all active lanes (inactive lanes remain in LP111).

The second form does not have the "ACT" argument, which indicates that data values are treated as 16-bit hex values, with each 4-bit nibble representing the LP state for its associated lane (bits 3..0 => lane 0, bits 7..4 => lane 1, etc).

Both forms of the command have an optional argument to indicate an integer duration (in ns) for each LP state in the sequence. If this argument is not present, 1/LPFreq is used for the duration.

For example, the following represents the HS burst entry sequence on all active lanes at 10 MHz

```
# LP_STATES ACT 100
7 1 0
```

To output the HS burst entry on lane 0-2, regardless of the number of active lanes and using the current LPFreq setting, discard the "ACT" parameter and use 16-bit values that describe the LP state for each lane:

```
# LP_STATES
7777 7111 7000
```

5.10.4 PH Component Command

The PH component command defines 4 bytes (CSI) or 3 bytes (DSI) to use in constructing the proper HS packet header sequence.

In CSI, the sequence consists of two identical 6-byte PH segments separated by the sync2 sequence, duplicated for each active lane. The four bytes provided by the user that the application uses to construct the PH segments are: Reserved, DataID, Data0/WC0, Data1/WC1. The PHCRC is automatically calculated for the PH.

In DSI, the sequence consists of two identical 6-byte PH segments (with different construction than CSI) separated by N sync2 sequences (where N is the number of active lanes), demultiplexed across active lanes. The three bytes provided by the user that the application uses to construct the PH segments are: DataID, Data0/WC0, Data1/WC1. The PHCRC is automatically calculated for the PH.

For example, in CSI:

```
# PH
0 12h 4 0
```

is short-hand for:

```
# HS_BYTES_PLUS_CRC ACT
0 12h 4 0
# SYNC2
# HS_BYTES_PLUS_CRC ACT
0 12h 4 0
```

In DSI:

```
# PH
12h 4 0
```

is short-hand for:

```
# HS_BYTES DEMUX
12h 84h 00h 80h eeh 81h
# SYNC2
# HS_BYTES DEMUX
12h 84h 00h 80h eeh 81h
```

5.10.5 PAYLOAD Component Command

The PAYLOAD component command defines payload bytes for an HS packet, demultiplexing across active lanes, and including postamble and HS burst exit signaling. For example:

```
# PAYLOAD
1 2 3 4 5
```

is short-hand for:

```
# HS_BYTES_PLUS_CRC DEMUX
1 2 3 4 5
# POSTAMBLE
# HS_BURST_EXIT
```

Note also that the "Send Cmd To File" (not yet implemented) option allows any single command (non-macro, non DCS WriteMemory, non-video command) to be sent to a file rather than to hardware. The format of this file is ASCII text, suitable for editing and re-sending to hardware via the File command.

5.10.6 LOOP_START and LOOP_END Component Commands

The LOOP_START and LOOP_END commands can be used to repeat data and/or commands in a file. The <loop count> argument for the LOOP_START command defines the number of repetitions. Nesting is allowed as long as both LOOP_START and corresponding LOOP_END occur in the same file. Note that looping does not save program memory but is merely a convenient shorthand when using the File Command.

The following example sends ten valid DSI Generic Long Write packets:

```
// send 10 packets
# LOOP_START 10

// SOT
# HS_BURST_ENTRY
# PREAMBLE
# SYNC

// pkt header
# PH
29h b0h 36h

// pkt payload
// (note DSI2 requires SYNC3 sequence between packet header and payload)
# SYNC3
# HS_BYTES_PLUS_CRC DEMUX
# LOOP_START 1000
1 2 3 4 5 6 7 8 9 10 11 12 13 14
# LOOP_END

// EOT
# POSTAMBLE
# HS_BURST_EXIT

// end packet loop
# LOOP_END
```

5.11 Command Buttons

The top-right pane of the main application window consists of four tab pages of 30 command buttons (labeled Cmds1, Cmds2, Cmds3, Cmds4). Clicking on the tab page header makes the 30 button command set available for use.

Command buttons can be associated with a named command by the user, allowing single-click sending of the named command on the CPhy bus. When assigned with a command, buttons are labeled with the command name and made active. Otherwise, they are inactive and show the label "<unassigned>". Note that buttons maintain separate DSI and CSI command sets, selected when the protocol is changed using the Standard menu.

Command buttons have an associated context menu, brought up by right-clicking on the button. The operations available from the button context menu are as follows:

- **Assign Current Cmd**: assigns the currently selected command in the "Cmd Name" drop-down control to the button
- **Assign Cmd**: assigns a named command selected from a sub-menu to the button
- **Remove Assignment**: removes the current command assignment from the button
- **Remove All Assignments on Current Cmd Page**: removes all command assignments from all 30 buttons on the button page.
- **Remove All Assignments on All Cmd Pages**: removes all command assignments from all button pages
- **Set Button Tooltip**: sets a tooltip string to display when the mouse hovers over the command button. To clear the tooltip, set the tooltip to the empty string. Note that removing a button assignment also clears the tooltip string.
- **View Current Cmd**: fills in the argument field controls with values associated with the command. For a macro, argument fields for the first component command in the macro are shown. In addition, the Macro tab is selected and the macro command definition is shown as read-only.
- **Edit Current Cmd**: for non-macro commands, this command is identical to View Current Cmd. For macros, in addition, the macro command definition is brought up in edit mode.

Note that, unlike application settings such as option settings, CPhy timing definitions, frame timing, and instrument configuration settings, command definitions and button assignments are *not* automatically saved when the application closes. Instead, command definitions and button assignments must be saved explicitly to a configuration file using the Save Cfg… menu option (File menu).

5.12 Operation

5.12.1 Sending a Command

Once the CPhy Generator instrument, frame timing, and CPhy timing configurations are set and desired options are checked, commands may be sent to the CPhy Generator for output. Unnamed commands are sent by simply selecting the command PktType, filling in the command arguments, and clicking the Send button. Named commands, once defined, can be sent by selecting the command name from the "Cmd Name" drop-down control and clicking the Send button. Finally, if a named command has been assigned to a button, left-clicking the button outputs the command on to the CPhy bus.

After the user requests a command to be sent, messages in the status bar will update progress. During this time, CPhyGenCtl performs the following tasks:

- The command is parsed and any necessary external file data imported (and decoded in the case of BMP files)

- MIPI protocol packets are built along with any necessary signaling structures (i.e. SOT, EOT, BTA, etc).
- Packet data is packed into internal data records and a play sequence is constructed for the CPhy Generator to use during output.
- Data records and play sequence are downloaded to the instrument.
- The instrument is "Run", initiating output, and control returns to the user at this point. Sending of the command is complete.
- However, if the command is a looping command, or contains event wait component commands, the CPhy Generator continues to run. Information in the status pane of the main window updates on-going operation.

5.12.1.1 Stop PG

As described above, the sending of commands goes through several stages of processing, some of which may take some time. Program download, in particular, can take tens of seconds to complete for video commands, when large blocks of video data are sent. During this time, the user may click the "Stop PG" button (located in the bottom-right pane of the main window) to abort command sending. In addition, once a command is running and the PG is actively looping (or waiting on an event), the Stop PG button may be clicked to terminate operation.

5.12.1.2 Restart PG

The "Restart PG" button in the bottom-right pane of the main window is used to resend the previous command, bypassing the potentially lengthy download process. This is useful, in particular, for video commands. When the PG is restarted, all timing configuration and parameters such as frequency, lane count, and clock start/stop mode from the original command are used (ignoring any changes since the command was sent).

5.12.2 Status

During operation, additional status is displayed in the lower-right pane of the CPhyGenCtl main window. This status is updated every second and includes the following:

- The PG run state (running, idle)
- The memory use of a running program (as a percent available in the instrument)
- The HS symbol rate of a running program.
- The data lane status (enabled, waiting on event, holding last HS symbol)
- Whether any LP contention has been detected since the last status reset
- Read response data received from the DUT after a BTA.[16]

[16] Note: read response data is only displayed after a running program has stopped or the user explicitly requests that the read response data is read from the instrument.

5.12.2.1 Contention Detection

The CPhy Generator is equipped with contention and data receivers on lane 0, whose thresholds can be set by the user in the Instrument Configuration dialog (see section 5.4.2 for more details). When the probe is outputting an LP high voltage and its receiver detects a voltage below the LP High contention threshold, an LP high fault is detected. When the probe is outputting an LP low voltage and its contention receiver detects a voltage above the LP Low contention threshold, an LP low fault is detected.

As these faults can occur on any of wire LP signal wire this results in six possible contention states. The mnemonics used in CPhyGenCtl for contention detection is as follows:

- **Tx0A** – contention detected on lane 0, wire A when transmitting LP0
- **Tx0B** – contention detected on lane 0, wire B when transmitting LP0
- **Tx0C** – contention detected on lane 0, wire C when transmitting LP0
- **Tx1A** – contention detected on lane 0, wire A when transmitting LP1
- **Tx1B** – contention detected on lane 0, wire B when transmitting LP1
- **Tx1C** – contention detected on lane 0, wire C when transmitting LP1

Once contention is detected, flags are held until a subsequent command is sent or the Status Reset button is pressed to clear state. Note also that the aggregate contention state is reflected on the Evt1 signal of the instrument back panel (see section 2.2), allowing for instrument trigger on the onset contention.

5.12.2.2 DUT Response

When a command or macro containing a BTA is sent, the CPhy Generator monitors the link for an LPDT response command from the DUT. The DUT response (if any) is displayed in the status pane of the main window as the string "DUT Resp: " followed by the first number of hex data bytes. If the response contains more than 12 bytes, a button labeled with an ellipsis ("…") appears so the user can click to view the entire response data in a dialog. Note that if the program is looping, DUT response data *does not appear* until the program is stopped or the user clicks the "Get DUT Resp" button.

To convey the LP signaling states transmitted by the DUT in generating its read response, an extra partial byte of data is included before the actual DUT response message. In particular, the first byte of each response conveys the initial two bits seen during escape entry (LP10, LP00, LP01, LP00), which is interpreted as binary '1' followed by binary '0' or equivalently a 0x1 data byte.

In addition, a special flag (0x100) is set in this value to indicate it represents a new escape-entry command. This allows for multiple responses to be queued and subsequently easily parsed by the user. Thus, to summarize, the first value displayed for each escape entry command should be 0x101.

Subsequent bytes displayed after escape-entry, represent the command data, starting with the escape command, which is expected to be 87h for an LPDT command. Thus, the first

header byte of each DUT response packet will be conveyed in the second byte of the displayed DUT response data following each 0x101 value.

The DUT response buffer is reset every time a CPhyGenCtl command is sent, but this could consist of a macro with multiple read commands. The response buffer is relatively large (4 Kbytes) and thus can capture a significant amount of data from multiple reads, which are simply concatenated in the buffer.

Example:

> During video-mode, say the user inserts four read commands over the course of a test and their responses are queued. When "Get DUT Resp" button is pressed and the ellipsis button is pressed,the DUT response bytes might look like those in Figure 12:

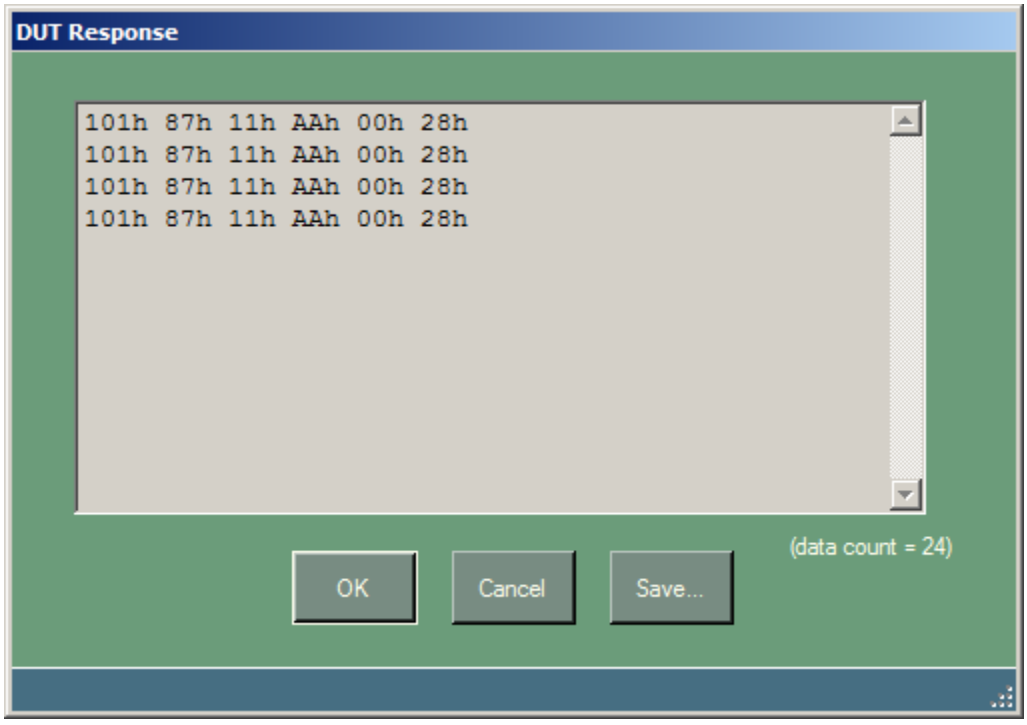

Figure 12 – DUT Response Dialog

Each response is delimited by the escape-entry value of 101h. This is followed by 87h for the LPDT command byte and then a DataID of 11h, indicating a generic short read response packet.

Note, there are two RPC commands that can be used to obtain the DUT response:

- GET_DUT_RESPONSE can be used programmatically to obtain the last response from the DUT.

- SAVE_DUT_RESPONSE can be used either in a script file or programmatically to save or append the last response from the DUT to a text file.

Please see the CPhyGenCtlRPC manual for more information.

5.12.3 Command Insertion

Command insertion is the ability to insert commands into an actively looping stream, whether in a video mode command or macro. While video mode commands automatically define an insertion point in the first vertical blanking line of a frame, the user must explicitly define an insertion point in a macro, denoting the location where the inserted command should output. The "Cmd Insertion Point" command is used for this purpose.

The following describes command insertion behavior:

- Command insertion is enabled using the "Enable Command Insertion" option. Checking this option means that a command sent while a program is running is intended for insertion.

- To send a subsequent command but not insert it, either stop the PG using the "Stop PG" button or uncheck the "Enable Command Insertion" option.

- An inserted command may be sent in HSDT or LPDT mode and will inherit the LPFreq, HS Sym Rate, and LaneCnt settings used in the looping program (regardless of GUI control settings).

- If the inserted command requires more time than that reserved by the Cmd Insertion Point (via its Delay parameter), the looping program is suspended until the inserted command completes.

- When a command is inserted, CPhyGenCtl checks to see if it can detect the insertion command code in the output stream. If it is not detected within ½ second, an error will be reported. To disable this check, select the "Disable command timeout" option. This option should be checked if the "Wait on external event to start" option is enabled or the macro contains a "Wait Ext Event" command.

- If a command is inserted when the PG is waiting on an external event to continue output, the inserted command is queued to be sent when output resumes and the insertion point is reached.

- If a macro contains multiple command insertion points, an inserted command will be output at the next encountered insertion point. Unless sequencing is controlled via external events, the user has no practical control over which command insertion point is used for the inserted command.

5.12.4 Controlling TrigOut

The CPhy Generator has a TrigOut front-panel signal that can be controlled both through the GUI and stream commands. It can be used as a scope or DUT-related triggering event.

The GUI main window has a "Trig Event" button that generates a pulse on the TrigOut signal when pressed. The event occurs whether or not the PG is running. However, note that the front-panel TrigOut *does not* occur if stream output is stalled waiting on an event. The duration of the pulse is configured in the Instrument Configuration dialog via the "Trig Pulse Width" control.

Also, the "Assert Trigger" command, which can be embedded in a macro, allows full control and timing of the TrigOut signal, allowing direct setting of the signal level (high or low), toggling state, or generating a pulse sequence (see section 5.6.6.14).

5.12.5 External Event Triggering

The GPI0 input on the back-connector of the CPhy Generator pin (see Figure 1) can be used as an external event signal for controlling output flow control. The triggering event on this signal is a rising edge (logic level '0' transitioning to a '1'). When detected, the latency of the event signal is bit-rate dependent, but generally is on the order of 100 ns.[17]

Currently, there are three ways this signal can be used:

1) To control when command output begins
2) To control frame advance during video sequence playback
3) To control when component commands in a macro are output

These ways are described in the next sections.

5.12.5.1 Controlling When Command Output Begins

The option "Wait on external event to start" (Options menu) allows users to configure the PG to wait for an external trigger event before command output begins (including video-mode commands). When this option is enabled, commands that are sent do not immediately output on the CPhy bus. Instead, while the stream is prepared and downloaded to the instrument as usual, output is held off until the external event is detected.

5.12.5.2 Controlling Frame Advance During Video Mode

The option "Advance Frame On External Event" in the Frame Timing dialog (see section 5.7.3) allows video-mode to be configured such that each frame loops indefinitely until the external event is detected. When the event is detected, output moves to the next frame in the sequence.

[17] However, the *effective* latency for the external input event when used for advancing video frames is much longer (one or two frames) due to the video program and pipeline architecture.

5.12.5.3 Controlling Macro Component Command Output

The command "Wait Ext Event" (section 5.6.6.13) can be used to pause output in a macro until the external event is detected. When the generator is paused, the CPhy bus is held in a static LP state as specified by the user.

Note that the "Disable Command Timeout" option (Options menu) applies in this case. If this option is checked, the instrument pauses and waits indefinitely for the external event. Otherwise, there is a static timeout period of one millisecond after which the program continues automatically.

5.12.5.4 Triggering the External Event Signal Via Software

Note that the HostEvent button in the main window can be used to cause the External Event signal (i.e. GPI0) to fire. This allows the user to test without external hardware connected or otherwise simply use a GUI button (or RPC command) for program flow control.

5.12.6 Wait Events

During program output, there are several events that cause output to suspend (in an LP state) until the event fires. They are:

4) **BTA Ack**: triggered by the DUT responding with the proper LP acknowledgement sequence immediately following a BTA. The default timeout associated with this event is 1 us and is not user-configurable.
5) **BTA Response**: triggered by the DUT eventually responding with a proper BTA response sequence after receiving the bus via a BTA. The default timeout associated with this event is 20 us but can be set by the user in the Instrument Configuration dialog.
6) **External Event**: triggered by a rising edge on the GPI0 input pin on the P339 back connector. The default timeout associated with this event is 1 ms and is not user-configurable.
7) **Host Event**: triggered in software via the HostEvent button of the main GUI window. There is no timeout associated with this event.

Actions that are associated with one or more of these events are shown Table 8:

Table 8 – Action/Event Association

Action	**Associated Event(s)**
Sending a MIPI packet with it BTA parameter set to "Yes"	BTA Ack, BTA Response
Sending a "Send BTA" command	BTA Ack, BTA Response
Sending a "Wait For BTA" command	BTA Response
Sending a "Wait Ext Event" command	External Event
Setting the "Wait on external event to start" option	External Event
Sending a ULPS command	Host Event

Most events as noted in their description have timeouts. Once program output has been suspended waiting on an event longer than its timeout, program output automatically resumes as if the event occurred.

If timeouts are not desired, for example during low-level testing, they can be disabled via the option: "Disable Event Timeout" (Options menu). When event timeouts are disabled, another feature of the software may prove useful. In particular, the Host Event button can be used to force any event causing program suspension, allowing simulation of the event.

5.12.7 Causing Packet Errors

The CPhyGenCtl GUI supports the ability to modify a packet's PHCRC and CheckSum fields to cause receive errors in packets. Normally, when a packet type is selected for a command in the main window and its parameters are filled in, read-only fields are displayed showing the DataID and PHCRC for short packets and, additionally, the WordCount and CheckSum for long packets. Checkboxes next to the PHCRC and CheckSum fields allow user modification of these fields to cause a receive error in the packet.

To change the PHCRC or CheckSum, simply check the associated checkbox, making the field value editable. Unchecking the control recomputes the correct PHCRC or CheckSum value and restores the field back to its original read-only state.

Note that the PHCRC value is duplicated in the packet header as well as across lanes. Use the File command if you need full control over all aspects of transmission, including changing individual PHCRC values in a packet header.

This capability applies to all CSI/DSI commands, including those within macros. With video mode, there is no ability for the user to set the PHCRC or CheckSum of any of the video frame packets.

When applied to the WriteMemory commands, if the command is partitioned into multiple WriteMemory commands, the user-set PHCRC and/or Checksum applies only to the first WriteMemory command. Remaining WriteMemory commands are always built with their correct PHCRC and Checksum.

An additional mechanism for introducing packet errors is provided under program control using RPC. In addition to allowing the PHCRC and Checksum to be set, the SEND_IMPAIRED_MIPI_CMD RPC call allows any byte in a packet to be XORed with an error mask before it is sent. Please see the description of this command for further information.

Finally, the File command can be used to define packets with errors, whether in the LP and HS signaling protocols, header, or payload. Please see section 5.10 for more information.

5.12.8 Script Recording

Script recording is a mode in which MIPI commands sent to the DUT are also written as RPC commands to a designated script text file. All commands, including macros and video mode commands are recorded. After recording, the script file can be sent as any other script file or used as a template for creating additional script files.

In addition, the current configuration state can be written to the script file. This allows the script file to initialize CPhyGenCtl to a particular configuration or be sent to the Moving Pixel Company to document your configuration when requesting assistance.

To enable script recording, select "Start Recording" from the Script menu. When this option is selected, a browse dialog is opened for you to designate the text script file name to use for recording.

Once script recording is enabled, the status bar displays the text "Recording" in a new pane. The script file is initialized with a header containing information about the current software and hardware configuration.

Any MIPI command sent (without error) during recording also causes an RPC command to be appended to the script file. In addition, selecting the menu option "Write Current State" from the Script menu causes the current state to be written to the script file. To end script recording, select "End Recording" from the Script menu.

5.12.9 Updating Firmware

Periodically, new firmware releases may be issued for the CPhy Generator. The latest firmware release can be downloaded from The Moving Pixel Company's website and (after unzipping) is labeled CPhyGenFirmware_VerX_Y.rbx, where the X and Y represent version and release digits respectively.

To update firmware in the CPhy Generator, perform the following steps:

- Connect to the instrument normally. If for some reason the instrument's firmware has been corrupted and is non-responsive, it is important in the Connect dialog to check the "Firmware update only" checkbox to prevent interaction with the FPGA.
- Select the Connect->Update Firmware…. menu option to bring up the firmware dialog.
- Browse for the .rbx firmware filename on your machine, obtained from the Moving Pixel Company.
- Click the Program button.
- Programming takes a couple minutes, with status updating progress.
- When programming is complete, close the application and power-cycle the instrument. On power-on, you can watch the instrument LEDs for the appropriate sequence. The red and yellow LEDs alternate for a couple seconds ending with the green LED remaining solidly on.

5.13 Customizing GUI Colors

In the CPhyGenCtl, the user can configure the colors of various elements of the GUI (e.g. the background and foreground colors of buttons, forms, menus, etc). This is achieved using the Color Options dialog, brought up via the Options menu by selecting the "Set Colors" menu item.

The Options dialog (see Figure 13) consists of a column of buttons on the left showing the current color settings of GUI controls. In the center of the dialog is a color square showing all the colors associated with a given luminance. Clicking and dragging the mouse inside of the color square will select the color of the currently selected control.

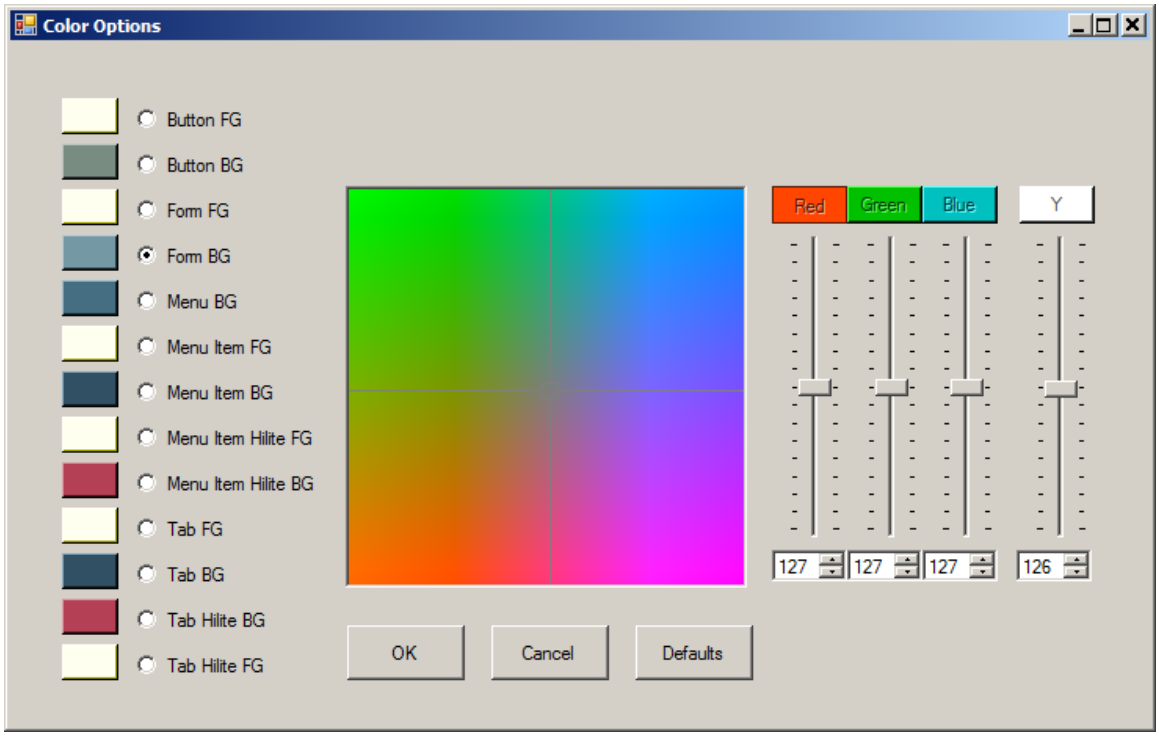

Figure 13 – Color Options Dialog

The luminance can be changed by adjusting the slider labeled 'Y'. Alternately, you can change the R, G, B color components using the sliders on the right of the color square. Yet another way to change the color is to click on a control button color (far right column). This loads the selected color as the current color. For example, to set the Button foreground the same as the Form foreground, click the Button FG radio button and click the color button next to the Form FG button.

As the control color changes, the main window colors will change in real time, so you can see how colors interact. To restore the default colors, click on the Defaults button. When you are done changing colors, to accept the new colors, click the OK button. To

discard your color changes, click the Cancel button. Color settings are saved on application exit and restored when the application is restarted.

5.14 GUI Options

The Options->Set GUI Options… menu item brings up a dialog to selectively enable/disable common warnings and configure behaviors in the application (see Figure 14). Options are self-explanatory.

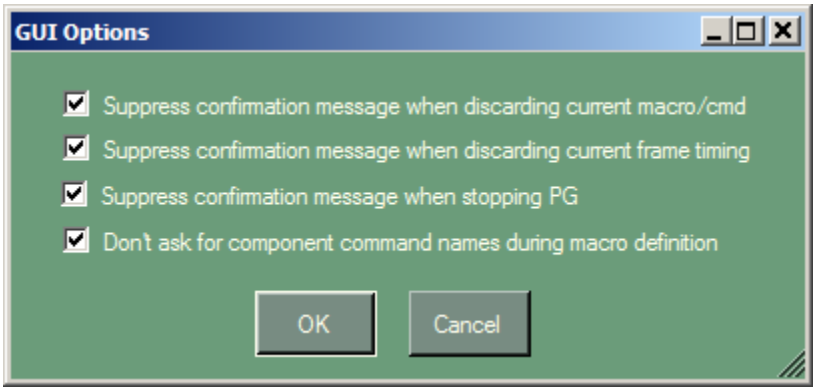

Figure 14 – GUI Options Dialog

5.15 Keyboard Shortcuts

A few keyboard shortcuts are available when the main window is active. A dialog listing them can be brought up using the Ctl-K key sequence (see figure):

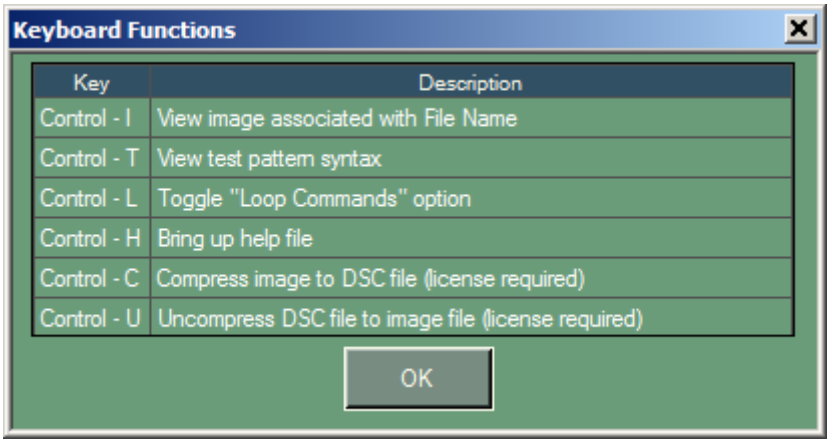

Figure 15 – Keyboard Function Dialog

Note that the view image and DSC compression functions apply to the current File Name for the current command.

5.16 Menus

This section outlines the menu commands available in CPhyGenCtl:

Menu	Description
FILE MENU	
Load…	Loads a previously saved command configuration file, overwriting any current commands and button assignments.
Save <fn>…	Saves the current command definitions and button assignments from both the CSI and DSI protocol sets to the current configuration file name. Note that the command configuration file does NOT store application settings, including options, CPhy configuration, PG configuration, and frame timing configuration settings.
Save <fn> As…	Same as the Save command above except a dialog appears to browse for and enter a new configuration file name.
<recent files>	Lists the four most recent configuration files. Selecting one of these file names loads the file.
Clear Recent File List	Clears the most-recent-file-list.
Exit	Exits the application.
CONNECT MENU	
Connect…	Brings up the PG connection dialog (see section 5.3).
Disconnect	Disconnects from the current instrument, putting the application in offline mode.
Enable RPC…	When checked, enables the application to accept and process incoming RPC requests (see the CPhyGenCtlRPC manual for more information).
Update Firmware…	Updates CPhy Generator firmware (see section 5.12.9).
STANDARD MENU	
DSI	Selects the DSI command set and button assignments.
CSI	Selects the CSI command set and button assignments.
OPTIONS MENU	
Loop Commands	When checked, causes commands (including video-mode commands and macros) to be looped indefinitely. Looping can be stopped with the Stop PG button.
Wait on external event to start	When checked, configures the PG to wait for an external event on its GPI0 input before outputting a sent command. When the event input is a logical '0', the PG waits until it goes to '1' before continuing (see section 5.12.5 for more information). The external event can also be used to control macro output using the "Wait Ext Event" packet type.
Disable event timeout	When checked, disables the normal timeout for wait events during program output, such as for a BTA response, BTA acknowledgement, and external events. This option is particularly useful for macros that include SEND_BTA, BTA_WAIT or WAIT_EXT_EVENT commands, which can

	cause arbitrary delays based on the testing environment and the behavior of the device under test. See section 5.12.6.
Enable "Hold Last Symbol" test mode	When checked, causes single HS packets including TGR and PRBS sequences to hold the last Postamble symbol on the bus when "Stop PG" is pressed for the first time. Status indicates the last symbol is being held and the PG is still running. Pressing "Stop PG" a second time terminates the test mode, returning the bus to the idle LP11 state.
Enable command insertion	When checked, sending a non-video command while a video-mode command loops causes the packet to be inserted (once) into the vertical blanking interval without stopping video playback. If unchecked, sending any command while a video-mode command loops causes CPhyGenCtl to confirm that the user wants to stop video playback to send the command. This function also works with a looping macro containing a command insertion point. See section 5.12.3.
Discard data during command insertion	When checked, an inserted command *replaces* existing data in the program stream for the duration of the inserted command. This mode thus does not affect the looping command timing (e.g. video). When unchecked, an inserted command is truly *inserted* into the program stream, extending looping command timing by the duration of the inserted command.
Enable Video Mode In Macros	When checked, sending a video-mode command when defining a macro inserts the video frame into the macro. If unchecked, only a single video-mode packet is added to the macro.
Send single packet per HS burst	When checked, macros that contain consecutive HS packets will be sent in individual HS bursts (i.e. bracketed by the HS burst entry/exit sequence). Otherwise, consecutive HS packets will be sent in a single HS burst.
Allow Image Rescaling	When checked, this option causes images specified in a Write Memory or video-mode command to be rescaled to the current output frame dimensions before conversion to the appropriate pixel format. Output frame dimensions are specified by the HActive and VActive fields of the Frame Timing configuration. If this option is not checked, an error is thrown when a command is sent whose image argument does not match the current output frame dimensions.
Enable EoT packets	When checked, an EoT packet is automatically appended to the end of an HS burst.
Send PPS with compressed video	When checked, sending a DSI compressed video-mode frame automatically causes a Picture Parameter Set command to be sent prior to the frame (as part of the same command).
Enable DSI scrambling	When checked, all normal DSI protocol packets sent in high-speed have their payload scrambled according to the DSI protocol.

Encode RAW format as Bayer…	When checked, a dialog is shown to select one of four Bayer patterns to encode CSI Raw video-mode data: GRBG, RGGB, BGGR, GBRG. The first two letters indicate the alternating colors for the first and every other line. The second two letters indicate the alternating colors for the second and every other line. If unchecked, Raw video-mode data is encoded as gray-scale.
Configure CPhy timing…	When selected, the CPhy Timing Configuration dialog, is shown, allowing the user to enter specific CPhy timing parameters such as HSPrepare, HSExit, preamble and postamble sequences, etc (see section 5.5).
Configure Write Memory Commands…	When selected, the Config Write Memory dialog is shown, allowing the user to configure how a DCS Write Memory command should be partitioned into multiple Write Memory commands (see section 5.6.6.9).
Configure DSC…	When selected, the DSC Configuration dialog is shown, allowing the user to specify the parameters of DSC compression (see section 5.7.5).
Set GUI Options…	Brings up a dialog for setting GUI options, allowing the user to selectively enable/disable command warning messages (see section 5.14).
Set Colors…	Brings up the Color Options dialog, allowing the user to change colors of various control elements in the GUI.
SCRIPT MENU	
Start Recording…	Brings up a file dialog to select an output text file to save recorded script commands. Then, the application is put in script recording mode, which logs and outputs MIPI commands to the script file as they are sent to the DUT. See section 5.12.8.
End Recording	Ends script recording.
Write Current State	Writes the application configuration state as RPC commands to the current script recording file (or otherwise prompts for a script file name to output current state).
ABOUT MENU	
Help…	Displays this manual as a help file.
Help RPC…	Displays the CPhyGenCtlRPC manual as a help file.
About…	Brings up a summary window displaying the current software version. This dialog also includes a summary of release notes of changes/fixes for each version

6 CPhyGenCtl RPC (Remote Procedure Calls)

To facilitate automated testing, CPhyGenCtl supports incoming RPC requests from other applications via a Microsoft .NET TCP server port. To support this function, a separate DLL called CPhyGenCtlRPC.DLL is provided, written in .NET. Any programming language that can interface to a .NET DLL can use this capability.

Alternatively, RPC commands have a script form that can be used in a text script file and run from the CPhyGenCtl GUI. In this case, no external programming is required. Many customers use script files extensively as they are a very powerful and convenient tool to configure and control CPhyGenCtl.

A separate document titled CPhyGenCtlRPC.pdf has been written to detail the RPC communication interface, supported commands, and example client code provided with installation of the CPhyGenCtl application. Please refer to this document for more information.