



*WAVECREST Corporation*

# DTS-550 Clock/Jitter Generator

User's Manual

*WAVECREST* Corporation continually engages in research related to product improvement. New material, production methods, and design refinements are introduced into existing products without notice as a routine expression of that philosophy. For this reason, any current *WAVECREST* product may differ in some respect from its published description but will always equal or exceed the original design specifications unless otherwise stated.

---

Copyright 1999

***WAVECREST* Corporation**

*A Technologies Company*

7275 Bush Lake Road

Edina, Minnesota 55439

(612) 831-0030

(800) 733-7128

[www.wavecrestcorp.com](http://www.wavecrestcorp.com)

All Rights Reserved

First Printing: March 1998

Second Printing: October 1998

Third Printing: September 1999

---

This page intentionally left blank.

### **ORGANIZATION**

This user's manual has been organized to provide complete details regarding the use and capabilities of the DTS-550A Clock Generator with as little intimidation as possible. It is assumed that the user has some familiarity with test equipment such as signal generators and oscilloscopes and the terminology involved with their usage. The chapters have been organized as follows:

#### **Chapter 1 - Product Description**

This chapter provides basic information about the DTS-550 and its capabilities and a brief overview of how jittered signals are generated.

#### **Chapter 2 - Setup**

This chapter covers the setup procedure required to get the unit up and running. It also provides functional descriptions of connectors found on the front and back of the enclosure.

#### **Chapter 3 - The User Interface**

This chapter describes the buttons on the instrument's front panel and their intended function. It also gives detailed descriptions of controls and options available on the front panel and user interface display.

#### **Chapter 4 - GPIB interface**

This chapter discusses the GPIB interface and support of GPIB commands.

#### **Chapter 5 – Advanced Features**

This chapter discusses the advanced features present on the DTS-550 such as the ability to create a custom jitter memory file (JMF).

#### **Chapter 6 - An Application Example**

The user is taken step-by-step through the setup and use of the instrument during a typical application. Sample setup, jitter and pattern files are provided so the user can quickly and easily learn the most common controls on the DTS-550.

**Appendix A** contains complete electrical specifications for the DTS-550.

**Appendix B** provides troubleshooting hints and is intended to assist the user in solving some of the more common problems encountered when using the DTS-550.

**Appendix C** discusses options and accessories available for the DTS-550. It also includes instructions on using an external keyboard to interface with the system.

**Appendix D** contains the jitter distribution type graphs.

**Appendix E** contains information for maintenance and service of the DTS-550.

## TEXT CONVENTIONS

The following symbols and text conventions are used in this manual to describe the functions of the DTS-550:

**BOLD** text is used to indicate buttons on the front panel of the DTS-550 and menu commands.

A `COURIER FONT` is used to show information that is entered directly from the optional external keyboard.

Names of keys on the optional external keyboard are surrounded by the symbols `< >` and are in bold. If two or more keys are to be struck simultaneously, they will be joined with a hyphen. For example, if the user is expected to strike the `<CTRL>` and `<Q>` keys simultaneously, this would appear as follows: `<CTRL-Q>`.

# SYMBOLS

---

## CAUTION

---

The CAUTION symbol indicates a potential hazard to the instrument. The user should pay particular attention to the instructions next to this symbol to avoid damage to the equipment or the circuit it is connected to.

\*\*\*

Three asterisks are used to identify features or menu selections that are not fully functional at this time.

This page intentionally left blank.

## Table Of Contents

---

About This Manual .....	<i>iii</i>
Table of Contents .....	<i>vii</i>
Chapter 1 - Product Description .....	1
Chapter 2 - Setup .....	9
Hardware Description .....	9
Hardware Setup .....	11
Software Description .....	12
Chapter 3 - The User Interface .....	13
Selecting Controls and Entering Data .....	13
Front Panel Keypad Operation .....	13
Mouse Operation .....	14
Main Panel Operation .....	15
Main Clock Control Functions .....	16
Sync Control Functions .....	20
Jitter Control Functions .....	22
Main Panel Softkey Functions .....	26
Main Menu Functions .....	27
I/O Level Functions .....	29
Chapter 4 - GPIB Interface .....	33
GPIB Interface and Commands .....	33
BIT Definitions .....	35
Common Commands .....	37
Bus Commands .....	48
Device Commands .....	49
Chapter 5 - Advanced Features .....	91
Creating A Pattern Memory File (PMF) .....	91
Chapter - An Application Example .....	93
Appendix A - Specifications .....	97
Appendix B - Troubleshooting Hints .....	101
Appendix C - Accessories and Options .....	103
Using an External Keyboard .....	103
Using an External Monitor .....	103
Appendix D - Standard Jitter Distributions .....	105
Appendix E - Maintenance & Service .....	107



This page intentionally left blank.

# CHAPTER 1 - PRODUCT DESCRIPTION

The DTS-550 is a versatile clock generator allowing precise control of jitter amplitude, frequency and distribution on digital clock waveforms. This capability allows accurate, repeatable characterization of jitter tolerance in clock recovery circuits for performing worst-case analysis. Jitter amplitudes are programmable over a wide dynamic range at jitter frequencies.

The patented Direct Time Synthesis (DTS) technology allows the transition position in a clock sequence to be directly synthesized in the time domain. This unique edge placement flexibility allows the creation of clock waveforms exhibiting precise, digitally programmed variations in edge timing (jitter).

DTS technology allows previously unavailable jitter distributions to be programmed. In addition to standard Sine wave variations, the generator also provides Spread Spectrum Clock 1 (SSC1), Sawtooth, Triangle and Pseudo-Random distributions. Custom distributions may also be created to meet unique applications.\*

\*Custom distribution files are currently created by Wavecrest only.

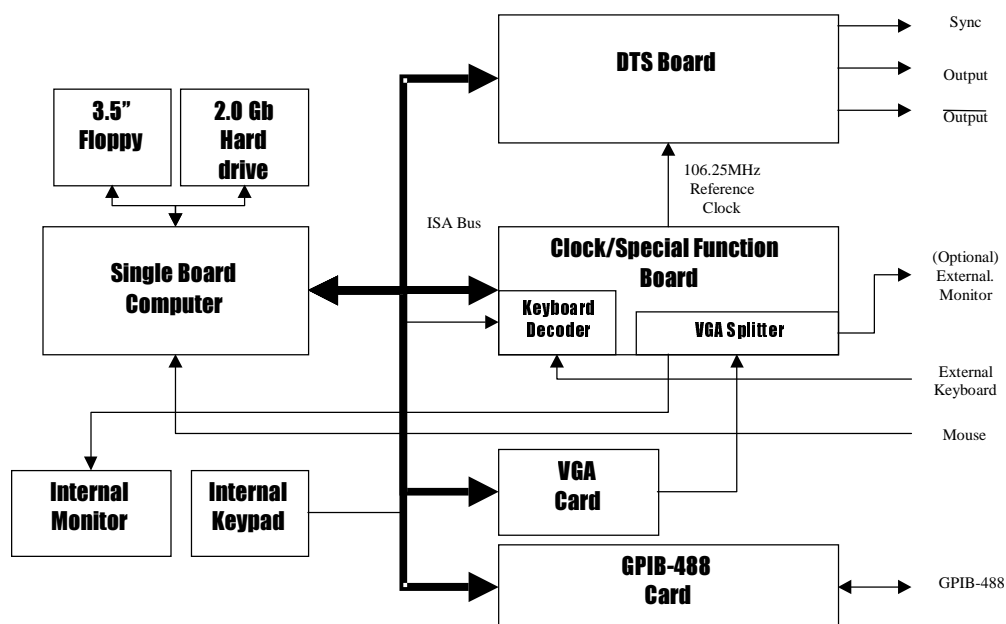
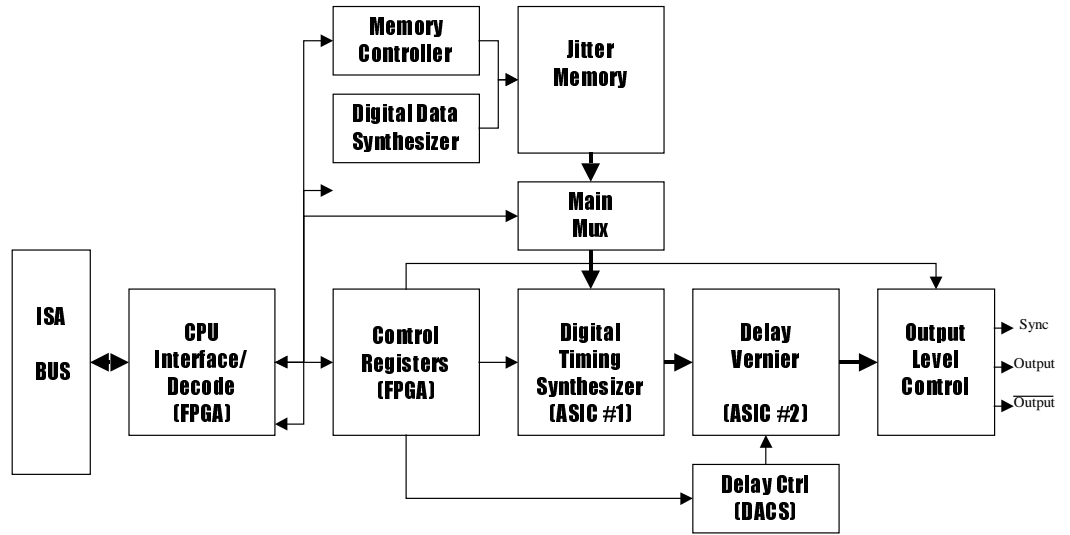


Figure 1.0 - DTS550 System Block Diagram

Instrument control is based on a Graphical User Interface with pull-down and pop-up menus. An embedded PC/AT compatible computer provides standard keyboard and mouse inputs, a VGA display (internal monochrome monitor or optional external color monitor) and a floppy drive for external program and data transfer. The front panel key switches, mouse and optional external keyboard all operate in parallel, allowing the DTS-550 to be used effectively in any environment.

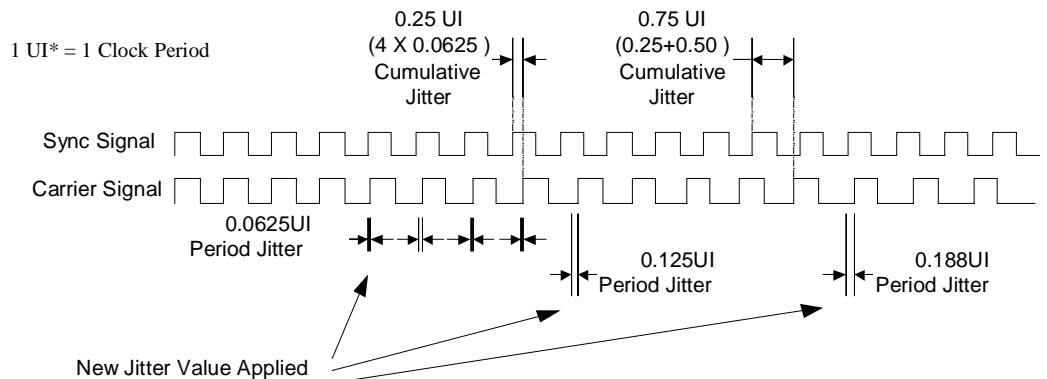
A GPIB interface is provided allowing remote control of the instrument using the GPIB command set. The versatility and flexibility of the DTS-550 make it an ideal source for high-frequency synthesized data, especially in environments such as Clock Jitter Tolerance application and test.

## Signal and Jitter Generation in the DTS-550



**Figure 1.1 - DTS-550 DTS Board Block Diagram**

Each edge of the DTS-550 OUTPUT/OUTPUT signal is ‘digitally synthesized’ by the combination of two custom ASICs located on the DTS-550 output board. The desired output frequency is ‘programmed’ into the first ASIC as the number of whole and fractional system clock counts (531.25MHz). ASIC#1 creates a series of repetitive digital pulse streams based on the number of whole system clock counts programmed into the ASIC. These pulses then are fed to the second ASIC (a programmable delay ASIC#2) which adds a specified amount of delay (based on the fractional system clock count) to perform the fine placement of the pulses. The delayed pulses are then combined with high speed logic gates to form the final output.



**Figure 1.2 - DTS-550 Jitter Timing Waveforms**

\* One Unit Interval (UI) is one cycle of the clock frequency, which is the normalized clock period. Jitter expressed in UIs describes the magnitude of the jitter as a decimal fraction of one UI.

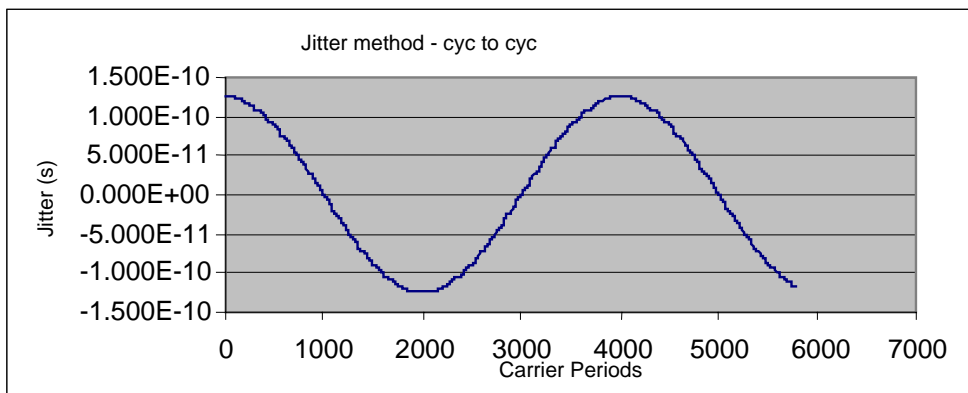
Jittered edges are created by reading the jitter data from memory. The jitter memory contains 4096 individual jitter data ‘points’ representing the jitter distribution selected. This jitter data is applied to the carrier signal each period through an accumulator register in ASIC#1. The accumulator will add the jitter data to each period, adjusting the jitter for each cycle as shown in Figure 1.2. The jitter value applied is in the shape of the selected jitter distribution. The DTS-550 will cycle through the jitter distribution once each jitter period, which is controlled on the DTS-550 interface with the Digital Data Synthesizer (DDS). The cumulative jitter shown in Figure 1.2 is the summation of the cycle to cycle jitter.

The DTS-550 Jitter method applies jitter every period for carrier frequencies below 350MHz and every other period for carrier frequencies above 350MHz\*. The jitter amplitude is displayed on the DTS-550 in terms of period and cumulative jitter. Jitter distributions may be selected from the standard patterns (sine, triangle, SSC1, Sawtooth or Pseudo-random) or created from custom patterns.

The DTS-550 jitter amplitude is displayed in peak to peak period jitter, giving the user a clear definition of the jitter applied each carrier period. Cumulative jitter is displayed and may be entered in the DTS-550 as well.

Figure 1.3 shows the DTS-550 Jitter on a cycle by cycle representation. This example is a simulation at 250 MHz carrier, 62.5KHz jitter frequency, sine distribution, and 0.25ns p-p period jitter (~160ns p-p cumulative jitter) amplitude.

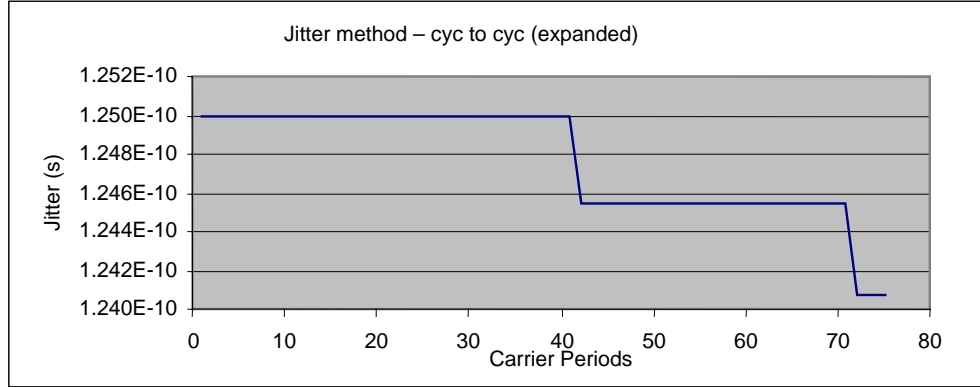
At the first carrier period the jitter is at a maximum deviation from the nominal carrier period. In this case, the peak is 0.125ns. Adding 0.125ns to the nominal carrier period ( $1/250\text{MHz} = 4.0\text{ns}$ ) will produce a period  $4.0\text{ns} + 0.125\text{ns} = 4.125\text{ns}$  long. The minimum carrier period ( $4.0\text{ns} - 0.125\text{ns} = 3.875\text{ns}$ ) occurs 2000 cycles later.



**Figure 1.3 - DTS-550 Jitter method period deviations**

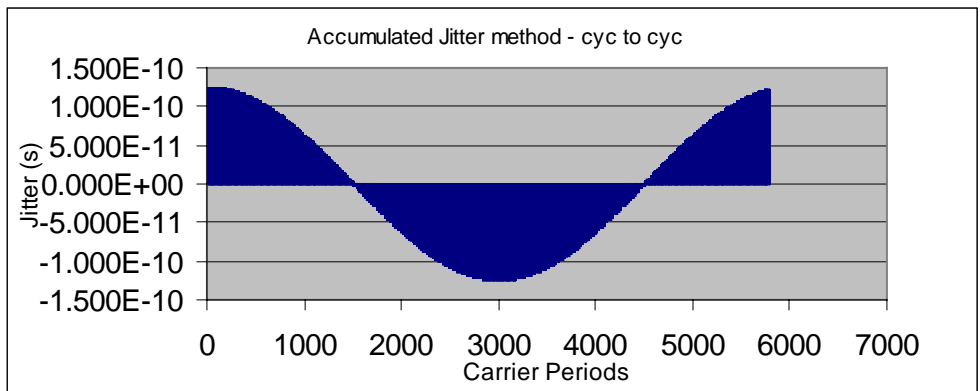
\* This is the case for larger amplitudes of jitter. The DTS-550 may only jitter up to every 16 cycles for low amplitudes of jitter. See Low Amplitude Jitter description.

Expanding cycle-to-cycle view of DTS-550 Jitter shown in , the synthesized sine wave can be seen. The expanded view shows the digital steps size of 0.2ps for this example.



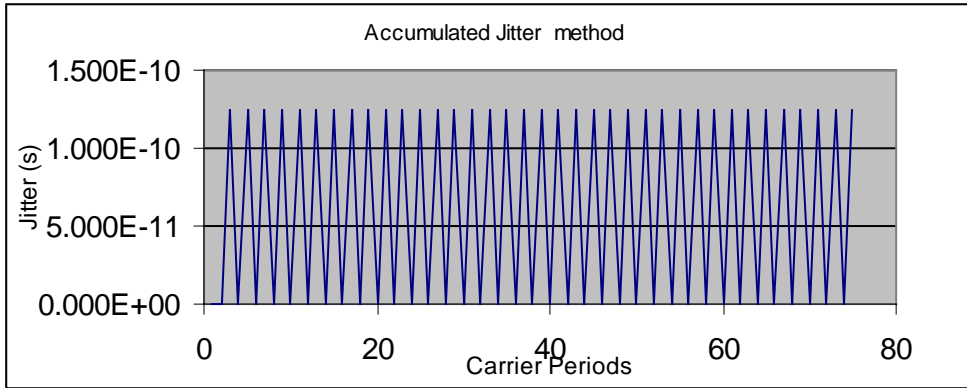
**Figure 1.4 – DTS-550 Jitter period deviations (expanded)**

The DTS-550 uses a frequency doubler to create carrier frequencies above 350MHz. When the frequency doubler is being used, jitter can only be applied every other cycle. Figure 1.5 and Figure 1.6 show simulated graphs of the higher carrier frequency with jitter using the DTS-550 Jitter method. Above 350MHz, as is the case in Figure 1.5, the first carrier period deviates from the nominal period by 0.125ns. The second carrier period is at nominal (jitter = 0). The third period the deviation is 0.125ns again. Alternating jitter every other cycle produces the solid pattern in Figure 1.5. Contrast this with Figure 1.3 for jitter below 350MHz.



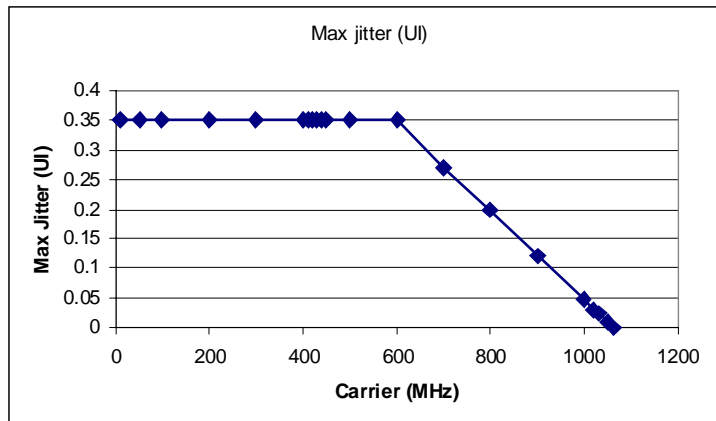
**Figure 1.5 – DTS-550 Jitter 600MHz**

The expanded jitter in Figure 1.6 shows the jitter added to every other cycle.



**Figure 1.6 – DTS-550 Jitter 600MHz (expanded)**

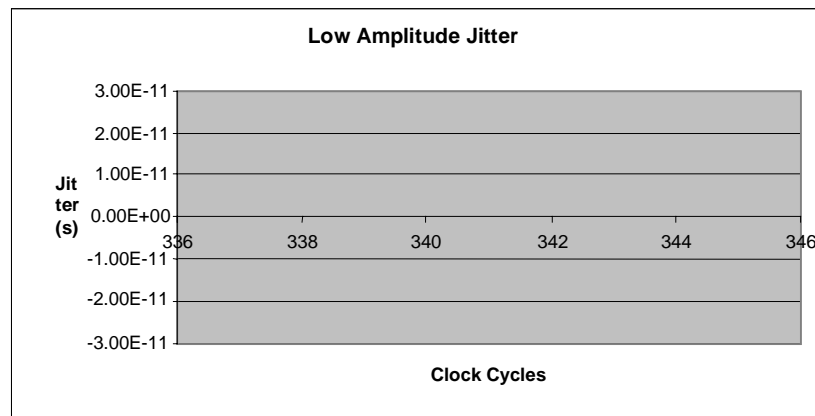
The jitter amplitude is entered as period jitter in the DTS-550. The amplitude must be less than 0.35UI p-p to avoid colliding with the previous pulse width edge. Additionally, the negative jitter requirement must not exceed the maximum frequency of the DTS-550. The graph in Figure 1.7 shows empirical data describing the maximum cycle to cycle jitter that can be applied to the carrier. Note that the allowable jitter at 1062.5MHz (the DTS-550 maximum frequency) is zero. The frequency doubling circuit is enabled above 350MHz.



**Figure 1.7 – Jitter Amplitude**

## Low Amplitude Jitter Generation in the DTS-550

For lower jitter amplitudes ( $\sim < 0.01\text{UI}$  variable), a second method is employed in the DTS-550 for producing jitter. The jitter data points are read from jitter memory in the same manner as described earlier, but instead of accumulating the latest jitter data point value, the jitter data point is added in only once to the period accumulator. This creates a momentary 'jittered' period and then a number of 'normal' periods would follow until the next jitter point is read and added in. This method works well with low amplitudes because the continuous method of jitter does not have the resolution to produce the small amplitude necessary. Figure 1.8 shows an expanded representation of a low amplitude jitter signal in which the jitter data point is being read in every four carrier cycle periods.



**Figure 1.8 - Low Amplitude Jitter (expanded)**

In this low amplitude mode of jitter generation, injected jitter may vary from every two periods to every 16 periods of the main carrier. This jitter injection frequency is indicated on the DTS-550 main panel in the Jitter Controls area.

The **Jittered Edge** control will display the number of clock periods that will occur before jitter is applied. '1' means jitter is applied every clock period. '2' means every other period, etc. The maximum number is every 16 periods.



**Figure 1.9 – DTS-550 Jitter Indicators**

The **Jitter Points** control indicates how many of the 4096 jitter distribution points in jitter memory are being used to create the jitter distribution pattern of jitter on the main clock output. This number will vary with main carrier and jitter frequency combinations. The access rate of the jitter memory, main carrier and jitter frequency all determine the number of points that are used.



This page intentionally left blank.

## CHAPTER 2 - SETUP

---

### HARDWARE DESCRIPTION

When first unpacking the DTS-550, check to see that the following items are present:

- DTS-550 Mainframe (1 channel)
- Power Cable
- Mouse
- Keyboard
- SMA Extenders
- 50 ohm terminators
- DTS-550 User's Manual (this document)

The DTS-550 hardware is based on a PC/AT architecture.

The floppy drive on the DTS-550 is a standard 3.5-inch, 1.44MB, PC/AT compatible drive. The drive is configured as the A: drive on the system computer.

---

### CAUTION

---

It is possible to damage the Clock/Data card through electrostatic discharge (ESD) on the I/O connectors. If the user is not wearing a static discharge strap, it is recommended to touch the frame of the unit prior to touching or connecting cables to any of the I/O connectors

\*\*\* The EXT INPUT is not used at this time. Functionality for this input will be available as an upgrade option in the future.

All of the front panel's SMA connectors have protective caps that must be removed before using. Attach the four SMA extenders to protect the threads on the device's SMA connectors from damage due to wear.

---

### CAUTION

---

To ensure high frequency signal integrity, it is recommended that the 50-Ohm termination caps included with the DTS-550 be installed on any unused output (OUTPUT, OUTPUT and SYNC) when operating the instrument.

The **SYNC OUT** can be configured to operate in one of four different modes\*\*\*. It can be used as a pattern marker for data\*\*\*, a follower output for the associated OUTPUT and OUTPUT signals, as a jitter sync output, or as an independent signal source.

The **OUTPUT** connector is programmed to output continuous clock type waveforms or to output digital data sequences. In either mode, pulse period, width, jitter and output levels are all individually controllable.

The **OUTPUT** connector produces an exact complement of what is seen on **OUTPUT**.

The three output connectors all have a status LED labeled **DISABLE**. This LED will be illuminated when the output has been disabled and will turn off when the output is enabled.

---

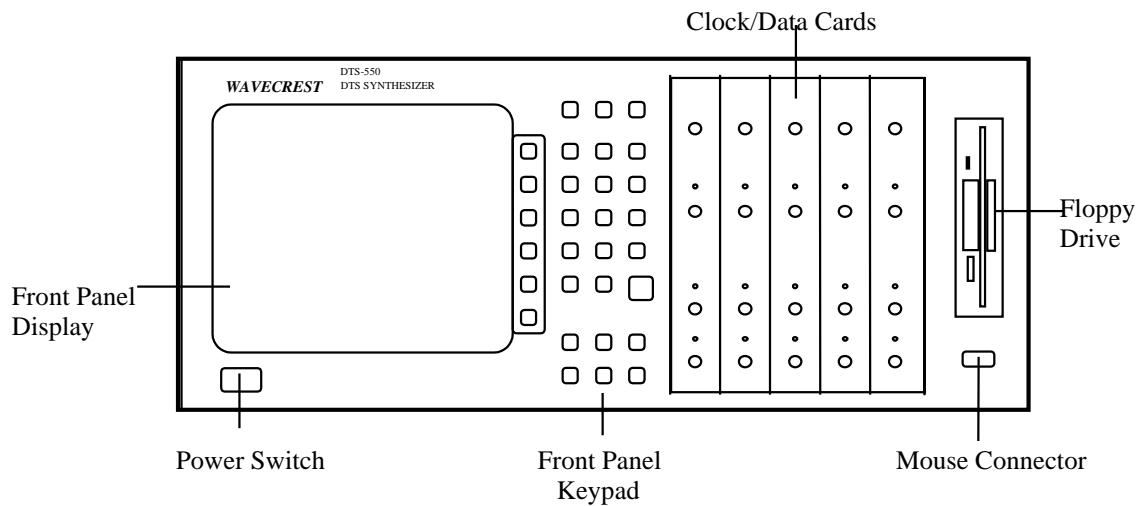
**CAUTION**

---

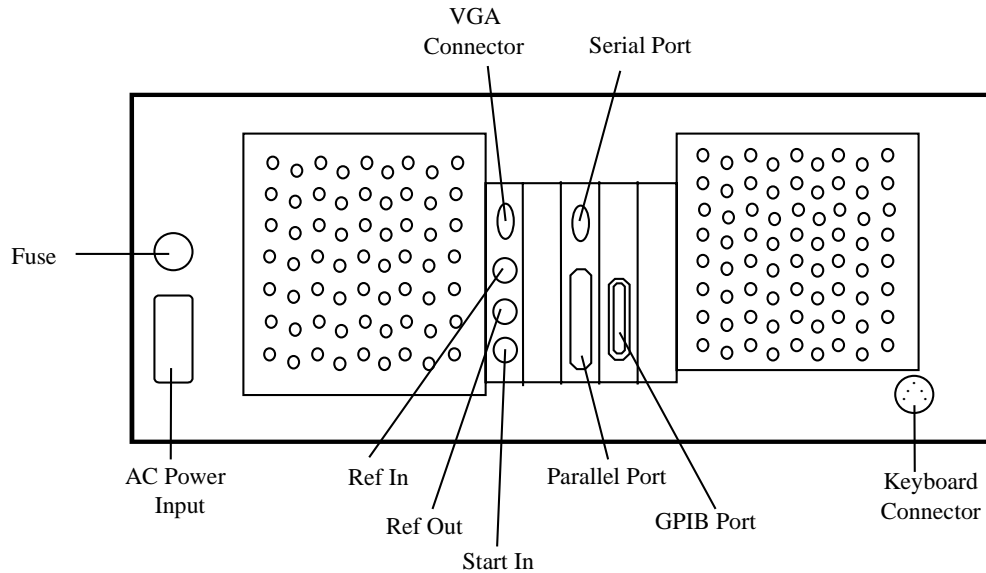
It is recommended that cables be connected to these outputs only when the status LEDs are lit (output disabled).

The female 15-pin D-sub connector is used to attach a standard VGA resolution (640 x 480) monitor.

The 24-pin GPIB connector is present on all systems. This connector is used to daisy-chain the instrument to other GPIB devices. At present, the application software only supports the unit as a listener, as opposed to being a talker or controller. The GPIB interface is discussed further in Chapter 4 of this manual.



**Figure 2.1 – DTS-550, front view**



**Figure 2.2 - DTS-550, rear view**

### **HARDWARE SETUP**

For proper ventilation, the DTS-550 should be placed on a level surface with approximately 3” of free space at the rear of the unit. Airflow is also required through the holes in the bottom of the chassis. Therefore, do not place papers or other objects underneath or directly in front of the unit. Using the legs will provide an additional 2” of airflow at the front and bottom of the instrument and will help to prevent overheating. Plug the power cord into the rear of the unit and to an appropriate AC outlet.

---

### **CAUTION**

---

Always use a three-conductor power cord and a properly grounded AC outlet with this instrument. Failure to ensure adequate earth grounding may cause instrument damage.

The DTS-550 requires a power source of 100-130VAC; 200-240VAC 900VA 50/60Hz and is protected by a user-replaceable 10A/250VAC fuse and by two internal 3A/250V fuses.

The supplied mouse should be attached to the 9-pin D-sub connector on the instrument front panel.

The supplied keyboard should be attached to the round AT connector on the back of the unit.

The positions of these connectors on the DTS-550 are indicated in Figure 2.2. At this point, the system should be ready to power-up. Proceed to either Chapter 3 or Chapter 5 for further information about the instrument’s usage and features.

## **SOFTWARE DESCRIPTION**

The DTS-550 comes with all necessary software pre-installed on the system's internal hard disk. Upon boot-up, the system will automatically run the DTS-550 application software. No setup or initialization is necessary in order to configure the software with the system.

The application software is based on National Instruments LabWindows<sup>®</sup> user interface. Chapter 3 discusses this interface in further detail and shows the common data selection and entry methods used with LabWindows<sup>®</sup>.

# CHAPTER 3 - THE USER INTERFACE

## Selecting Controls and Entering Data

The DTS-550 was designed for flexibility and ease-of-use. In addition to the front panel keypad, the unit includes a mouse and an external keyboard. This allows the instrument to be used effectively in any environment, as well as allowing the user to enter data or select controls in the fashion they are most comfortable with. This section will focus on using the instrument through the front panel and with the mouse, but will also list the external keyboard sequence.

## Front Panel Keypad Operation

The keys labeled **DATA ENTRY** on the front panel are general purpose and are used to enter data into selected controls. The digit keys **0-9** and the decimal point ( . ) key are used to enter a numeric value. A hyphen (-) is used to place a negative sign in front of a numeric value. The **EXP** ('E' with external keyboard) key is used to enter the exponent of a value expressed in scientific notation (ex.:  $2.5E-3 = 0.0025$ ). The  $\leftarrow$  key is used to backspace and overwrite the previous character and functions the same as the backspace delete ( $\leftarrow$ ) key on a standard keyboard. A value entered into a control does not take effect until the **ENTER** key is pressed. If the user tries to enter invalid data into a control, they will be warned of this after pushing **ENTER**.

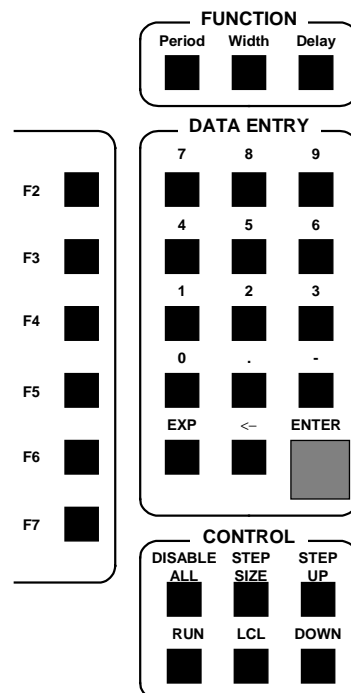


Figure 3.0 – DTS-550 Front Panel Keypad

An error message will appear when a value is entered that is out of range for a particular control. When this type of error occurs, an “out of range” dialog box will occur to let the user know the range of valid values. The “out of range” value entered will be replaced by the previous “in range” value after acknowledging the “out of range” message box. The user may then choose to reenter a value within the valid range.

**Note:** Sometimes a warning message may appear when editing a control that may cause another control’s value to be out of range. For example, when entering a new clock frequency the jitter amplitude control value may exceed the maximum for the new clock frequency just entered. A warning message will appear stating that the jitter amplitude value will be adjusted to the maximum.

Pressing the **FUNCTION** keys **PERIOD** and **WIDTH** will make the **FREQ/PERIOD** and **WIDTH/DC%** controls active respectively. Note that the **DELAY** function is not implemented.

The **Softkeys (F2-F7)** along the right side of the front panel display will vary in their function depending on what screen function is being displayed. Generally, the **F2** and **F3** keys are programmed as the **<- PREV** and **NEXT ->** functions to tab forward and backwards through the displayed controls. Specific functionality of each softkey is described in the various panel control descriptions. The softkeys can also be accessed with the external keyboard with function keys **F2-F7** or with a mouse.

The **LCL** key in the **CONTROL** group toggles the status of the Panel On/Off state (Local/Remote). See the GPIB command `:DISPLAY:PANEL ON|OFF`. The external keyboard sequence to toggle the Panel On/Off state is **<Ctrl-L>**. The other **CONTROL** function keys each perform a specific function and are described in the Main Panel control descriptions.

The Menu controls may only be operated with an external keyboard or mouse connected.

## Mouse Operation

When using a mouse, numeric and list controls may be activated (in any order) by a ‘left-mouse-click’. List controls values may be incremented/decremented by clicking on the down/up arrows on the left side of the controls. They also may be operated in a pop-up format by clicking on the control list area and selecting one of the items from the list with a single mouse click. Button controls (such as **Jitter On/Off**) will toggle states with a single click. Other button controls (such as **UP, DN, or Softkeys**) will perform special functions when clicked and are described later in this section. Menu functions are activated with a mouse by left-clicking on the desired menu function and then selecting the menu choice from the menu pop-up list with another click.

## Main Panel Operation

Upon system power-up, all application software required to run the DTS-550 is automatically loaded and executed. The initial display seen by the user will contain a number of controls used to specify OUTPUT, OUTPUT\_, and SYNC signals, jitter, and other parameters related to the instrument operation. All of the DTS-550 settings will revert to their last powered state if the instrument program was shut down by executing 'File' then 'Quit' from the main panel menu.

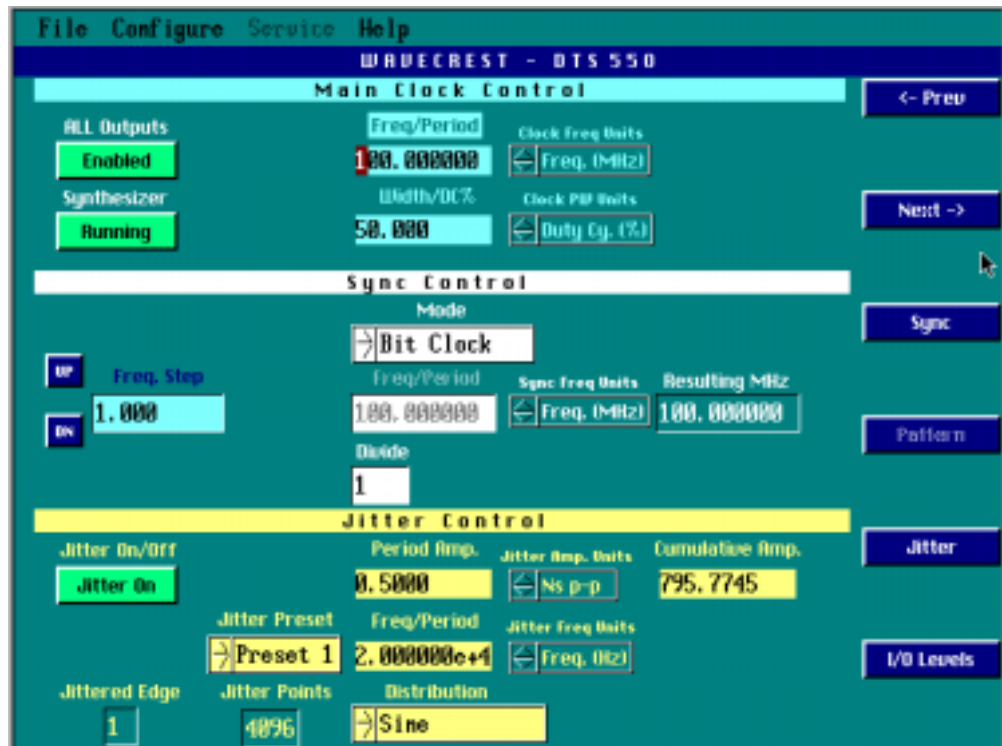


Figure 3.1 – DTS-550 Main Panel Screen

The **Main Panel** controls are organized into five main groups: Main Clock Control; Sync Control; Jitter Control, Main Menu and the Softkeys (F2-F7).



## Main Clock Control Functions



Figure 3.2 – DTS-550 Clock Control Screen

### All Outputs

The **All Outputs** control on the display is used to enable/disable outputs on the DTS-550. This control will be set to **Disabled** when indicators on all outputs are on. When toggled to **Enabled**, the relays that carry the SYNC OUT, OUTPUT, and OUTPUT signals on the output card are actuated and data is allowed to pass to the output connectors.

Control Type: **Toggle Button**  
 Value Range: **Enabled/Disabled**  
 Default Value: **Disabled**  
 Front Panel Key: **ALL** or **ENTER** with **All Outputs** control active will toggle output status.  
 External Keyboard: **< Ctrl-A >** or **ENTER** when control active

### Synthesizer

The **Synthesizer** control is used to toggle the run state of the DTS-550. When set to **Running**, the instrument's internal DTS synthesizer will generate the specified waveforms. When set to **Stopped**, no output waveforms will be generated by the unit.

Control Type: **Toggle Button**  
 Value Range: **Running/Stopped**  
 Default Value: **Stopped**  
 Front Panel Key: **RUN**  
 Or **ENTER** with **Synthesizer** control active will toggle run/stop status.  
 External Keyboard: **< Ctrl-R >**  
 Or **ENTER** when control active

**NOTE:** In order to receive any type of output signals from the unit, the **All Outputs** control needs to be set to **Enabled** and the **Synthesizer** control needs to be set to **Running**.

## Main Clock Control Functions

**Freq/Period** This control sets the frequency (MHz) or period (Ns) of the OUTPUT and  $\overline{\text{OUTPUT}}$  signals. The active control shown depends on the unit of measure selected in **Clock Freq Units** control. Changing one of these controls automatically changes the other control when displayed. (Freq = 1/Period.) Changing the main clock frequency/period will also affect other front panel controls as described in Table 3.1.

Control Type: **Numeric entry-float**  
 Value range: **10MHz <= Freq <= 1062.5MHz**  
**0.94117(ns) <= Period <= 100.0(ns)**  
 Default value: **Freq = 500 MHz**  
**Period = 2.0 ns**  
 Front Panel Key: **PERIOD** (selects Freq/Period control)  
**use numeric keypad.**  
 External Keyboard: **< Ctrl-P > and enter number.**

Control Group	Resulting Effect with new Freq/Period
<b>Width/DC%</b>	<b>DC%</b> = unchanged <b>Width</b> = new Period * DC%
<b>Jitter Period/Cumulative Amp</b>	<b>UI p-p</b> = unchanged <b>NS p-p</b> = <b>UI p-p</b> * new <b>Period</b> * scaling <b>DEG p-p</b> = unchanged
<b>Sync Divide</b> (Sync mode = Bit Clock)	If new <b>Freq</b> > 531.25MHz and <b>Divide</b> = 1 then <b>Divide</b> = 2
<b>Resulting Sync Output-</b> (Sync mode = Bit Clock)	If new <b>Freq</b> > 531.25MHz and <b>Divide</b> = 2 then <b>Resulting Sync output</b> = new <b>Freq</b> / 2

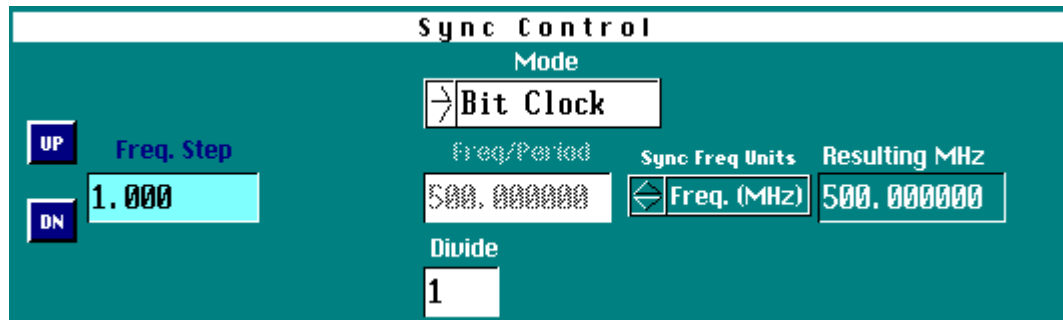
**Table 3.1 - Controls affected by Freq/Period control change**

## Main Clock Control Functions

**Width/DC%** This control sets the pulse width (Ns) or duty cycle (%) of the OUTPUT and OUTPUT signals. The active control shown depends on the unit of measure selected in **Clock PW Units** control. Since duty cycle is a function of the pulse width ( $DC\% = 100 * (\text{pulse width} / \text{period})$ ), a change in the **Width** control will also change the **DC%** control. In addition, if the **Freq/Period** controls are modified, this control will change so that the **DC%** control is held constant. (See also clock Freq/Period control description.)

Control Type: **Numeric entry-float**  
 Value range:  **$100 * (0.250/\text{Period (Ns)}) \leq DC\% \leq 100 * (\text{Period} - 0.250)/\text{Period}$**   
 **$0.250 \leq \text{Width (Ns)} \leq \text{Period} - 0.250$**   
 Default Value: **Width = 1.0ns**  
**DC% = 50%**  
 Front Panel Key: **WIDTH** (selects the **Width/DC%** control) **and use numeric keypad.**  
 External Keyboard: **< Ctrl-W > and enter number.**

## Sync Control Functions



**Figure 3.3 – Sync Control Functions**

**Mode** The **Sync Mode** control is used to specify the mode of operation of the SYNC OUT signal. There are currently four\* sync modes available to select from the control list.

**Sync Mode**  
**Pattern Sync\***

**Mode Description**

This mode will cause the SYNC OUT to act as a start of pattern marker.

**Jitter Sync**

This mode will cause the SYNC OUT to follow the **Jitter Frequency** value. The **Jitter Frequency** value must be greater than 510.8Hz. The **Jitter On/Off** control does not need to be enabled. The **Sync Freq/Period** and **Sync Divide** controls will be disabled with this selection.

**Bit Clock**

This mode will cause the SYNC OUT to follow the main **Clock Freq/Period** value. The **Sync Divide** control is enabled. The **Sync Freq/Period** control is disabled and will follow the **Clock Freq/Period** value.

**Independent**

The **Independent** mode will cause the SYNC OUT to operate as an independent signal source. If jitter is programmed, OUTPUT will move relative to SYNC OUT. This selection can also be used to generate a clock-type waveform on the SYNC OUT connector that has a different frequency from the OUTPUT signal. Both the **Sync Divide** and **Sync Freq/Period** controls are enabled.

Control Type:

**List Control**

Value range:

**Pattern Sync\*,**

**Jitter Sync,**

**Bit Clock,**

**Independent**

Default Value:

**Bit Clock**

Front Panel Key:

**F4** (Selects **Sync Mode** control). Use **STEP UP** and **STEP DOWN** to **change mode**.

External Keyboard:

**F4** (Selects **Sync Mode** control). **<Ctrl-U>**(Step up) or **Up Arrow** and **<Ctrl-N>**(Step Down) or **Down Arrow** to **change mode**.

\***Pattern Sync** currently not implemented.

## Sync Control Functions

### Freq/Period

This control sets the SYNC OUT frequency (MHz) or period (Ns). The active control shown depends on the unit of measure selected in **Sync Freq Units** control. Changing one of these controls automatically changes the other control when displayed. (Freq = 1/Period.). This control is enabled only when **Sync Mode** is set to **Independent**.

Control Type: **Numeric entry-float**  
Value range: **2.0840MHz <= Freq <= 531.25MHz**  
**1.882352(ns) <= Period <= 479.85(ns)**  
Default value: **Freq = 500 MHz**  
**Period = 2.0 ns**  
Front Panel Key: **Select control and use numeric keypad.**  
External Keyboard: **Select control and enter number.**

### Divide

This control sets the **Sync Divide** value. The **Sync Divide** control is only programmable when the **Sync Mode** is set to **Bit Clock** or **Independent**. When **Divide** is set between 1 and 255, the SYNC OUT waveform appears as a pulse-type waveform. The pulse width will be the same for all division factors from 1-255. However, when a value larger than 255 is used in this control, the SYNC OUT waveform will appear as a clock-type waveform with identical high and low times.

Control Type: **Numeric entry-integer**  
Value range: **1 <= Divide <= 255**  
**(Sync Mode =Independent-increments of 1)**  
**2 <= Divide <= 254**  
**(Sync Mode =Bit Clock-increments of 2)**  
**256 <= Divide <=4080 (increments of 16)**  
**(Sync Mode = Independent or Bit Clock)**  
Default value: **1**  
Front Panel Key: **Select control and use numeric keypad.**  
External Keyboard: **Select control and enter number.**

### Resulting MHz/ns

This control displays the SYNC OUT frequency or period value based on the **Sync Freq Units** control setting. This control is non-editable and is for informational use only.

## Sync Control Functions

**Step, UP, DN** The **STEP** and **UP** and **DN** controls are located in the Sync Control Group but pertain to other control groups as well. The **Step** control is used to specify the increment/decrement value of the currently selected control when using the Step **UP** & **DN**. This provides a convenient method of ‘stepping’ a control through a range of values from the front panel with out having to enter discrete control values. The **Step** control will be displayed only when a numeric or list control that supports the step function is selected. Some controls (list controls) have a fixed increment value and the **Step** control will be shown, but disabled. The **Step** control label will reflect the control that is currently selected (when then step function is shown). Selecting the **UP** button will increment the currently selected control by the value listed in the Step control field. Conversely, selecting **DN** will decrement the current control. When the upper or lower limit of the selected control is met, the value will no longer be incremented/decremented.

Front Panel Key: **STEP SIZE, STEP UP, DOWN**  
 External Keyboard: **<Ctrl-S>, <Ctrl-U> or Up Arrow, <Ctrl-N> or Down Arrow**

Step Control	Default Incr./Decree Value	Editable
Clock Freq	1	Y
Clock Period	0.01	Y
Clock Freq Units	1	N
Clock Width	0.01	Y
Clock DC%	1	Y
Clock PW Units	1	N
Sync Mode	1	N
Sync Freq	1	Y
Sync Period	0.01	Y
Sync Freq Units	1	N
Sync Divide	1	N
Jitter Period Amp UI	0.01	Y
Jitter Period Amp NS	0.01	Y
Jitter Period Amp DEG	1	Y
Jitter Amp. Units	1	N
Jitter Cumulative Amp UI	0.01	Y
Jitter Cumulative Amp NS	0.01	Y
Jitter Cumulative Amp DEG	1	Y
Jitter Preset	1	N
Jitter Freq	1e5	Y
Jitter Period	0.01	Y
Jitter Freq Units	1	N
Jitter Distribution	1	N
I/O Output Levels Preset	1	N
I/O Output Levels Termination	1	N
I/O Sync Levels Preset	1	N
I/O Sync Levels Termination	1	N

## Jitter Control Functions

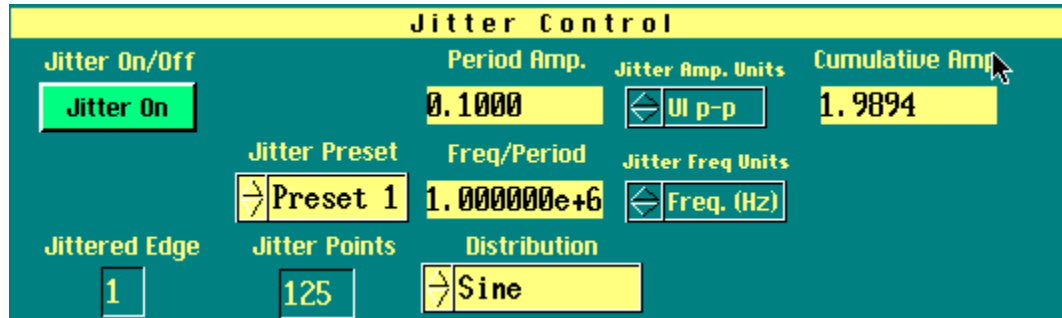


Figure 3.4 - Sync Control Functions

### Jitter On/Off

This control turns the jitter on or off on the OUTPUT and  $\overline{\text{OUTPUT}}$  signals. The current values in the jitter amplitude jitter frequency and jitter distribution controls determine the amount and type of jitter applied.

Control Type: **Toggle Button**  
 Value Range: **Jitter Off/ Jitter On**  
 Default Value: **Jitter Off**  
 Front Panel Key: **ENTER** with **Jitter On/Off** control active will toggle jitter on/off status.  
 External Keyboard: **< Ctrl-J >** or **ENTER** when control active

### Period Amp.

### Cumulative Amp.

The jitter amplitude applied to the OUTPUT and  $\overline{\text{OUTPUT}}$  signals is specified in either of these controls. **Period Amp.** Jitter is defined as the peak-to-peak jitter in one clock cycle (or period) of the **Clock Freq/Period** control value. **Cumulative Amp.** jitter is defined as the peak-to-peak jitter in one cycle of the **Jitter Freq/Period** control value. (See theory of jitter generation section.) The **Jitter Amp Units** control specifies how the amplitude will be displayed: **UI p-p** (Unit Interval peak-to-peak. Where a Unit Interval is defined as one clock cycle of the **Clock Freq/Period** control value); **ns p-p** (nanoseconds peak-to-peak) and **DEG p-p** (degrees peak-to-peak. Where one clock cycle equals  $360^\circ$ ). Changing the value in one jitter amplitude control will affect the other.

**NOTE:** The maximum value for the **Period Amp. Jitter** for a given frequency is specified in the specification in Appendix A. When a value is entered that exceeds the maximum the warning message “Jitter exceeds max allowed. Setting to max.” will appear. The both the **Period Amp** and **Cumulative Amp** controls will then be **automatically** set to the maximum allowable value for the given **Clock Freq/Period**.

Control Type:	<b>Numeric entry-float</b>
Value range:	<b>See Appendix A.</b>
Default value:	<b>0</b>
Front Panel Key:	<b>F6 (Selects Period Amp control). Use numeric keypad.</b>
External Keyboard:	<b>F6 (Selects Period Amp control) and enter number.</b>



## Jitter Control Functions

**Jitter Preset** The **Preset** control is used to select pre-defined jitter parameters that have been set by the user. Up to five sets of parameters may be stored for future use. When this control is selected, a pop-up list appears with options from **Preset 1** to **Preset 5**. A unique jitter amplitude, frequency, and distribution can be defined for each one of the presets. The **Jitter On/Off** state remains unchanged when changing between presets. In order to permanently save setup information, the user will need to execute the **File-Save Setup** option from the menu bar on the main display.

Control Type: **List Control**  
Value range: **Preset 1 – Preset 5,**  
Default Value: **Preset 1**  
Front Panel Key: **Select control and STEP UP and STEP DOWN**  
External Keyboard: **Select control and <Ctrl-U> or Up Arrow and <Ctrl-N> or Down Arrow**

**Freq/Period** This control sets the Jitter frequency (Hz) or Period (Sec). The active control shown depends on the unit of measure selected in **Jitter Freq Units** control. Changing one of these controls automatically changes the other control when displayed. (Freq = 1/Period.). This control affects the amount of **Jitter Cumulative Amp** while keeping the **Jitter Period Amp** amount constant. Also, when **Sync Mode = Jitter Sync** the SYNC OUT will follow the new jitter frequency/period entered.

Control Type: **Numeric entry-float**  
Value range: **0.01Hz <= Freq <= 5.0e+06Hz (5.0MHz)**  
**2.0e-07 Sec <= Period <= 1.0e+02 Sec**  
**Freq >= 510.8Hz. or Period <= 1.957e-03**  
**when Sync Mode = Jitter Sync.**  
Default value: **Freq = 1.0e+06Hz**  
**Period = 1.0e-06Sec**  
Front Panel Key: **Select control and use numeric keypad.**  
External Keyboard: **Select control and enter number.**

## **Jitter Control Functions**

**Distribution** This control selects the Jitter Distribution type that is loaded into jitter memory and applied to the OUTPUT and OUTPUT signals when jitter is enabled. There are six jitter distribution choices: **Sine, Triangle, Sawtooth, SSC1, Random** and **<From File>**. A graph representing each of the standard included distribution types is included in Appendix D. Selecting the **<From File>** option will bring up the Select Jitter File window, which allows the user to specify a jitter memory file (.jmf) to be used to specify the distribution shaped. (See Creating a Jitter Memory File.)


Control Type:	<b>List Control</b>
Value range:	<b>Sine, Triangle, Sawtooth, SSC1, Random, &lt;From File&gt;</b>
Default Value:	<b>Sine</b>
Front Panel Key:	<b>Select control and use STEP UP and DOWN</b>
External Keyboard:	<b>&lt;Ctrl-U&gt; or Up Arrow and &lt;Ctrl-N&gt; or Down Arrow</b>

**Jitter Edge** This non-editable control indicates how often jitter is applied to a clock period signal. A '1' indicates the every clock period will be jittered, whereas a '2' indicates that every other period will have jitter applied. A '4' indicates that every fourth period will be jittered, etc. For moderate to larger amounts of jitter amplitude, this control will indicate a '1' (for clock frequencies < 350MHz) or '2' (clock frequencies > 350 MHz). For small jitter amplitudes, this number will vary between '2' and '16' in even numbered increments. (See 'How Jitter is Generated'.)

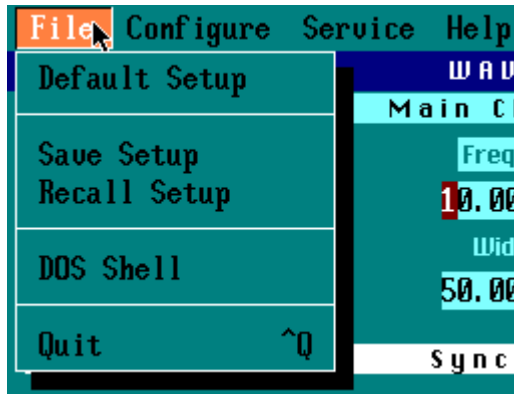
**Jitter Points** This non-editable control indicates the number of jitter points that are being used (out of a total of 4096 points) to represent the current jitter distribution type stored in jitter memory. This number may be used as a rough indicator to how well the jitter distribution 'shape' is being applied. This number generally will be lower when the ratio of clock frequency to jitter frequency is low. This number will vary from '1' to '4096' depending on the clock frequency to jitter frequency relationship.

## Main Panel Softkey Functions

Figure 3.5 - Main Panel

 A vertical stack of six blue rectangular softkey buttons with white text. From top to bottom, the buttons are labeled: '<- Prev', 'Next ->', 'Sync', 'Pattern', 'Jitter', and 'I/O Levels'. Each button is separated by a thin white border.	<b>&lt;-Prev (F2)</b>	This control moves the focus to the previous control in the tab sequence.
	<b>Next-&gt; (F3)</b>	This control moves the focus to the next control in the tab sequence.
	<b>Sync (F4)</b>	This control sets the <b>Sync Mode</b> control to be active.
	<b>Pattern (F5)*</b>	(currently disabled)
	<b>Jitter (F6)</b>	This control moves the focus to the <b>Periodic Jitter Amp</b> control.
	<b>I/O Levels (F7)</b>	This control pops up the <u>I/O levels</u> menu where SYNC OUT and OUTPUT / <u>OUTPUT</u> DC levels may be set. (See I/O Level functions.)

## Main Menu Functions



**Figure 3.6 - File Pull-down Menu**

- File-Default Setup** Selecting **Default Setup** returns all control values to their default settings. This includes all values under the sub-menu **I/O Levels**.
- External Keyboard: **<Alt-F>+D+ENTER**
- File-Save Setup** The **Save Setup** selection allows the user to save settings from a particular application for future use. Selecting this option brings up the **Select Instrument State File** window where the user selects a directory and filename to save program information to. **An external keyboard will be required to enter a new filename.** All saved state files should have a **.SAV** extension in their name. If the user selects a filename that already exists, they will see the message “File already exists, do you want to overwrite it?” and will have the option of overwriting the previous setup or selecting a new filename. When the proper destination filename is highlighted, click on **Select** to save DTS-550 setup information.
- External Keyboard: **<Alt-F>+S+ENTER**
- File-Dos Shell** The DOS shell command starts a DOS command processor so that you can type commands as if you were at a normal DOS prompt. When you have finished entering DOS commands type “exit” to return to the DTS-550 application.
- Keyboard: **<Alt-F>+D+D+ENTER**

## Main Menu Functions

**File-Recall Setup** The **Recall Setup** option allows the user to retrieve a previously saved state file. After selecting this option, the **Select Instrument State File** window appears and the user can select a previously saved state file (file extension SAV). File selection is done in the same manner as described under the **Save Setup** option.

External Keyboard: <Alt-F>+R+ENTER

**File-Quit** Selecting this option will cause the DTS-550 to immediately exit the application software and return to DOS, presenting the user with the standard DOS command prompt. The current settings of the instrument will be saved to file and restored when the DTS-550 application is run again.

External Keyboard: <Ctrl-Q> or <Alt-F>+Q+ENTER

**Configure-GPIB** The **GPIB** (See Figure 3.7) selection is used to change the GPIB address for the GPIB interface. Enter a number between 1 and 30 and press OK. The message “You must restart program to change settings.” will appear. Any changes made to the GPIB address will take affect after restarting the DTS-550 application. For further information regarding the characteristics and usage of the GPIB interface, see Chapter 5.

External Keyboard: <Alt-C>+G+ENTER



Figure 3.7 - Configure Pull-down Menu

**Help-<Rev. info>** Lists the DTS-550 software version number. For example, “DTS-550 , aVer1.17u” (See Figure 3.8).

External Keyboard: <Alt-H>



Figure 3.8 - Help Pull-down Menu

## I/O Level Functions

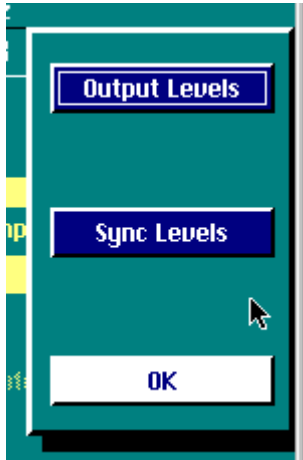


Figure 3.9 - I/O Levels

Selecting **I/O Levels (F7)** from the main window will bring up a pop-up window with 3 choices (Figure 3.9): **Output Levels (F5)**, **Sync Levels (F6)** and **OK (F7)**. It is recommended that when adjusting I/O levels, set the **Synthesizer** control to **Stopped** and the **All Outputs** control to **Disabled**. Failure to do so could cause damage to any devices connected to the I/O connectors.

The **I/O Levels** control is used to specify voltage and termination characteristics on the, SYNC OUT, OUTPUT and OUTPUT connectors. **I/O Levels** can be selected either with a mouse or by pressing the **F7** key on the front panel keypad. Pressing either **Output Levels (F5)** or **Sync Levels (F6)** will bring up a screen similar to the pop-up screen in Figure 3.10 (Output Levels window shown.). Since both the Output and Sync windows are identical, this section will describe the controls as they pertain to both.

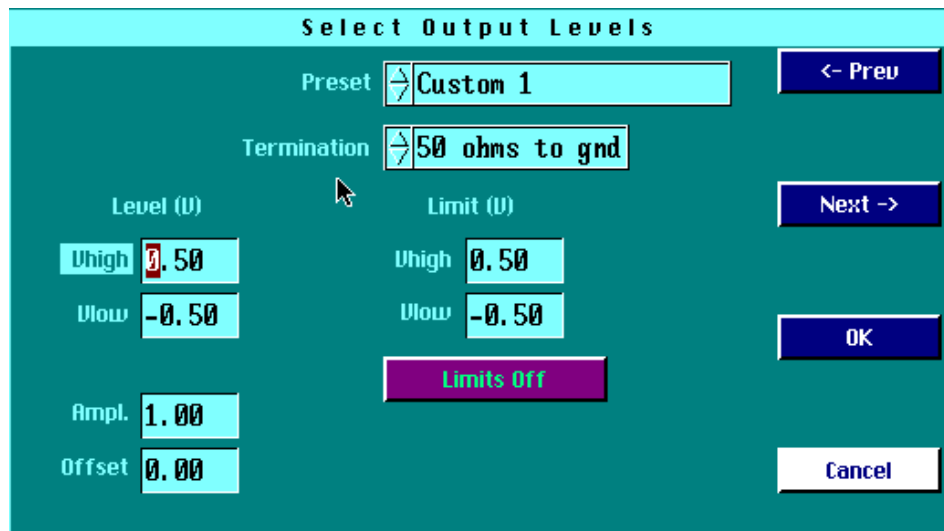


Figure 3.10 - Select Output Levels

The **<-Prev F2** and **Next -> F3** controls operate as control movement keys. (See main panel screen softkey functions.)

**Note:** In order to save any changes made to the Select Output/Sync Levels window, you must select '**OK (F4)**' after editing. This will save any changes to the screen and return to the main panel. Pressing '**Cancel (F5)** or **<ESC>**' will discard any changes made and return to the main panel.

## I/O Level Functions

### **Preset**

This list control displays 10\* predetermined and 3 user definable presets for the configuration of the impedance, high, low and voltage limit settings for the selected Output or Sync levels. First, the user should make a selection from the **Preset** control. Choices appear for many types of standard logic, including **ECL**, **PECL**, **TTL**, **CMOS (3.3V)**, and **CMOS (5V)**. Selecting any of these will automatically determine the input's voltage threshold and termination impedance, and these values will be filled in and cannot be selected for the user to change. There are also choices for three custom voltage levels available to the user. If any of the custom voltage levels are chosen, the termination impedance and voltage threshold controls become active and their values need to be specified.

Control Type:	<b>List Control</b>
Value range:	<b>ECL: 50 ohms to gnd*</b> <b>ECL: 50 ohms to -2V</b> <b>ECL: 50 ohms to +3V</b> <b>PECL: into open ckt</b> <b>TTL: 50 ohms to gnd,</b> <b>TTL: into open ckt,</b> <b>CMOS (3.3V): 50 to gnd,</b> <b>CMOS (3.3V): open ckt,</b> <b>CMOS (5V): open ckt,</b> <b>Custom 1 (user defined),</b> <b>Custom 2 (user defined),</b> <b>Custom 3 (user defined)</b>
Default Value:	<b>Custom 1</b>
Front Panel Key:	<b>Select control and use STEP UP and DOWN</b>
External Keyboard:	<b>&lt;Ctrl-U&gt; or Up Arrow and &lt;Ctrl-N&gt; or Down Arrow</b>

\*ECL: 50 ohms to gnd not available as Sync Level preset.

### **Termination**

This control sets the termination impedance for the selected channel. This list control is available only when the Custom 1-Custom 3 presets are selected.

Control Type:	<b>List Control</b>
Value range:	<b>50 ohms to gnd*</b> <b>50 ohms to -2V</b> <b>50 ohms to +3V</b> <b>into open ckt</b>
Default Value:	<b>50 ohms to gnd</b>
Front Panel Key:	<b>Select control and use STEP UP and DOWN</b>
External Keyboard:	<b>&lt;Ctrl-U&gt; or Up Arrow and &lt;Ctrl-N&gt; or Down Arrow</b>

## I/O Level Functions

### **Vhigh, Vlow Ampl., Offset**

Voltage levels can be specified in terms of **Vhigh** and **Vlow** or **Amplitude** and **Offset**. Changing one pair of these characteristics will automatically modify the other pair. With the **Limits On/Limits Off** control set to **Limits On**, the voltages specified under the **Level** heading cannot range outside of the values specified under the **Limits** heading. (See Output and Sync Preset Limit Tables in the GPIB command section for minimum/maximum voltage values.).

Control Type: **Numeric entry-float**  
Value range: **See Output and Sync Preset Limit Tables 4.2 & 4.5**  
Default value: **See Output and Sync Preset Limit Tables 4.2 & 4.5**  
Front Panel Key: **Select control and use numeric keypad.**  
External Keyboard: **Select control and enter number.**

### **Limit Vhigh,Vlow**

The **Vhigh** and **Vlow** controls under the **Limit (V)** heading are used to specify absolute maximum and minimum voltage levels for the outputs. These should be set to the extremes that, if violated, could cause damage to the circuit connected to the outputs.

Control Type: **Numeric entry-float**  
Value range: **See Output and Sync Preset Limit Tables 4.2 & 4.5**  
Default value: **See Output and Sync Preset Limit Tables 4.2 & 4.5**  
Front Panel Key: **Select control and use numeric keypad.**  
External Keyboard: **Select control and enter number.**

### **Limits On/Off**

This controls enables/disables the **Vhigh** and **Vlow Limit** values. When enabled, the Limit Vhigh and Vlow values may invalidate a previously entered value in the **Vhigh, Vlow, and Ampl. and Offset Level** controls. The value(s) that are outside of the limit values will be modified to conform to the new **Vhigh/Vlow Limit** range.

Control Type: **Toggle Button**  
Value Range: **Limits Off/Limits On**  
Default Value: **Limits Off**  
Front Panel Key: **Select control and ENTER to toggle.**  
External Keyboard: **Select control and ENTER to toggle.**



This page intentionally left blank.

# CHAPTER 4 - GPIB INTERFACE

---

## GPIB INTERFACE AND COMMANDS

Software control of the DTS-550 is accomplished using the GPIB port at the rear of the unit (see Figure 2.2). This port can be connected via a standard GPIB cable to any device that is capable of issuing GPIB commands.

### Address Configuration

Setup of the GPIB port address is done by selecting **Configure**, then **GPIB**, from the menu bar at the top of the display. To change the GPIB address, enter a valid address (1-30) and select **OK (Factory default address is 4)**. A message will appear stating that the **DTS550 must be restarted before the new GPIB address entered will take effect**. Clear this message by selecting **OK** and cycle the power to the DTS550. The new address will now be valid.

### Local and Remote Modes

At power on the DTS550 is running in the **local** operating mode. The GPIB port is automatically polled for commands. This does not affect the user's ability to enter commands using a mouse or from the front panel keypad. However, doing this is not advised since a GPIB command can override a front panel command without the knowledge of the front panel operator. This would not pose a problem for the DTS-550 but could lead to some unnecessary operator confusion. It is also possible to inadvertently disable the GPIB bus by opening various functions from the front panel (such as any of the menu options). It is recommended that the DTS550 front panel be locked out by sending the `:DISPlay:OFF` command through the GPIB port or by pressing the 'LCL' button on the front panel. The front panel may be 'unlocked' by pressing the 'LCL' key again or by issuing the `:DISPlay:ON` command. The message **'GPIB has disabled the display'** will appear when the display is in the **remote** or locked state. The front panel's displayed settings will not be updated when the DTS550 is in the **remote** (or display locked) state.

### Sending Commands over the GPIB

It is generally recommended that before sending commands to the DTS550, a GPIB device clear is sent. This has the effect of getting the device's attention and clearing any previous GPIB bus conditions. Also, the GPIB talker/controller should be configured to assert the EOI line on "End of Data". "End of String" mode is not sufficient. The `*RST` command should also be used to default the instrument settings of the DTS550 before a programming sequence is sent. `*CLS` should be sent before each command sequence to clear any error conditions. `*OPC` or `*OPC?` may be used to determine that a command has completed. (See Common Commands section.) This is recommended to prevent the DTS550 from missing issued GPIB commands.

### **GPIB Command Notation**

Any command description surrounded by “[ ]” are optional. Arguments in “<>” should be substituted with the data type specified. The “|” is used to indicate a logical OR function. Multiple commands may be sent in a single command by separating the individual commands with a “;” (semicolon). Some commands may have a query form as indicated by the “?” parameter option. **Multiple queries should not be sent in a single command string.** After issuing a query command the DTS550 should be ‘read’ before issuing any other commands. The returned data may not be valid if any commands are injected between issuing a query and reading the GPIB port.

### **Invalid Commands**

Errors generated by GPIB commands produce no warnings or error messages. The command that caused the problem is simply ignored. A command may be rejected if:

- The command contains a syntax error.
- The command cannot be identified.
- The command has too few or too many parameters.
- The command has a parameter that is out of range.
- The command is out of context.

## **BIT DEFINITIONS**

**CME** -Command error. Indicates whether the parser detected an error.

**ESB** - Event status bit. Indicates if any of the conditions in the Standard Event Status Register are set and enabled.

**EXE** - Execution error. Indicates whether a parameter was out of range, or inconsistent with current settings.

**MAV** -Message available. Indicates whether there is a response in the output queue.

**MSS** - Master summary status. Indicates whether the device has a reason for requesting service. This bit is returned for the \*STB? query.

**OPC** - Operation complete. Indicates whether the device has completed all pending operations.

**RQS** - Indicates if the device is requesting service. This bit is returned during a serial poll. RQS will be set to 0 after being read via a serial poll (MSS is not reset by \*STB?).

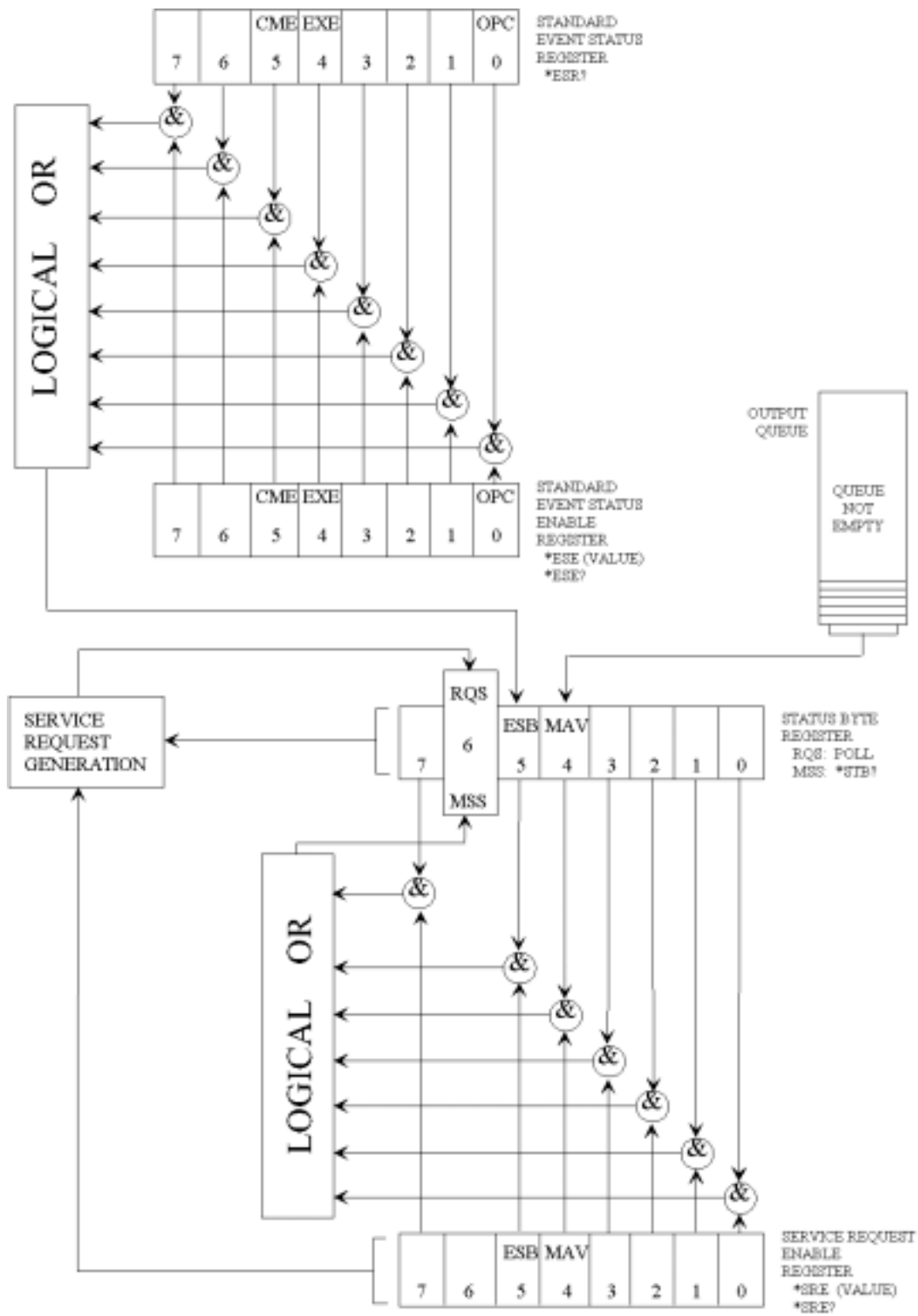


Figure 4.0 - Status Reporting

## Common Commands

### **\*CLS (Clear Status)**

Command

\*CLS

The \*CLS command clears the Event Status Register, the Status Byte Register, the trigger bit, the local bit and the error queue. The Event Status Register is read by the \*ESR? command. The Status Byte Register is read by the \*STB? command or a serial poll.

Example

This example clears the status and data registers.

```
Send( 0 , 4 , " *CLS " , 4 , EOI ) ;
```

## Common Commands

### **\*ESE (Event Status Enable)**

Command	<p><code>*ESE&lt;mask&gt;</code></p> <p>The *ESE command sets the Standard Event Status Enable Register bits. The Standard Event Status Enable Register contains a mask value for the bits to be enabled in the Standard Event Status Register. A one (1) in the Standard Event Status Enable Register will enable the corresponding bit in the Standard Event Status Register. A zero will disable the bit.</p>
	<p><code>&lt;mask&gt;</code> An integer, 0-255.</p>
Example	<p>In this example, the *ESE 64 command will enable OPC, user request, bit 0 of the Standard Event Status Enable Register.</p> <pre>Send(0,4,"*ESE 64",7,EOI);</pre>
Query	<p><code>*ESE?</code></p> <p>The *ESE query returns the current contents of the Standard Event Status Enable Register.</p>
	<p><code>&lt;mask&gt;</code> An integer, 0-255,</p>
Example	<p>In this example, the current contents of the Standard Event Status Enable Register are placed in the numeric variable named Event. The value of the variable is printed on the screen.</p> <pre>Send(0,4,"*ESE?",5,EOI); Received(0,4,&amp;Event,1,EOI); Printf("%d\n",Event);</pre>

## Common Commands

### **\*ESR? (Event Status Register)**

Query

\*ESR?

The \*ESR? query returns the contents of the Standard Event Status register.

**Note:** Reading the register clears the Standard Event Status Register and the ESB bit in the STB register.

Returned Format

An integer, 0 to 255.

Example

In this example, the current contents of the Standard Event Status Enable Register in the numeric variable, Event. The value of the variable is printed on the screen.

```
Send(0,4,"*ESR?",5,EOI);  
Receive(0,4,Event,1,EOI);  
Printf("%d\n",Event);
```



## Common Commands

### **\*IDN? (Identification Number)**

Query

\* IDN?

The \*IDN? query returns the company name, model number and software version by returning the string:

WAVECREST,DTS-550,<xx.xxx>

An \*IDN? query must be the last query in a message. Any queries after the \*IDN? in this program message will be ignored.

<xx . xxx> Specifies the software version and revision number.

Returned Format

WAVECREST,DTS-550,a1.18c

Example

This example places the unit's identification information in the string variable, Message, then prints the identification information to the screen.

```
char MESSAGE[50];  
Send(0,4,"*IDN?",5,EOI);  
Receive(0,4,MESSAGE,50,EOI);  
Printf("%s\n",MESSAGE);
```

## Common Commands

### **\*OPC (Operation Complete)**

Command	<p>*OPC</p> <p>The *OPC (operation complete) command will set the operation complete bit in the Standard Event Status Register when all pending device operations have finished.</p>
Example	<pre>Send(0,4,"*OPC",4,EOI);</pre>
Query	<p>*OPC?</p> <p>The *OPC? query places an ASCII "1" in the output queue when all pending device operations have finished.</p>
Example	<pre>Send(0,4,"*OPC?",5,EOI); result=0 /*wait for reset to finish*/ while ((result&amp;0X20 !=0){     ReadStatusByte(0,4,&amp;result); } Receive(0,4,data,1,EOI);</pre>
Returned format	"1" ASCII

# Common Commands

## **\*RCL (Recall)**

Command

\*RCL<specific setup #>

The \*RCL command restores the state of the DTS from a specified set of saved setups. There can be ten (10) different setups (0 through 9).

<specific setup>

An integer, 0-9, identifying the save/recall register that contains the setup you want to recall.

Example

This example recalls the DTS550 setup stored in configuration file 1.

```
Send( 0 , 4 , " *RCL1 " , 5 . EOI ) ;
```

**NOTE:** See common command \*SAV for information of specific information recalled/saved.

## Common Commands

### **\*RST (Reset)**

#### Command

\*RST

The \*RST command places the instrument in a known state (See default table below). Use the interface clear (IFC) bus command to perform a hardware reset. This command should be used to initialize the DTS550 to a known state before beginning a programming sequence.

#### Example

This example resets the DTS550 to a known default state (See default table below).

```
int result;

Send(0,4,"*CLS",4,EOI);
Send(0,4,"*RST;*OPC",9,EOI);
result=0 /*wait for reset to finish*/
while ((result&0X20 !=0){
    ReadStatusByte(0,4,&result);
}
/*reset complete*/
```

## Common Commands

### Default setup:

<b>Main Clock Control</b>	<b>State</b>
Outputs	OFF = 0
Run/Stop	Stopped = 0
Display	Enabled = 1(Local)
Freq/Period	500Mhz / 2.0ns
DC% / PulseWidth	50% / 1.0ns
<b>Sync Control</b>	
Mode*	Bit clock
Freq. / Period	500 Mhz / 2.0ns
Divide	1
<b>Jitter Control</b>	
Periodic Amplitude	0.0 UI / Sec / Deg
Cumulative	0.0 UI / Sec / Deg
Preset**	1
Freq. / Period	1MHz / 1e-6 Sec
Distribution	Sine
Jitter On/Off	Off = 0
<b>Main &amp; Sync I/O Levels</b>	
Vhigh	0.5V
Vlow	-0.5
Vamp	1.0V
VOFFset	0V
Preset***	Custom1
TerminatiON	50 Ohms to Gnd
Vhigh Limit	0.5V
Vlow Limit	-0.5V
Limits On/Off	Off

**Table 4.0 - Default Settings**

\*Independent Mode: Frequency = 500Mhz; Div = 1

\*\*All jitter presets (1-5) are set to default values also.

\*\*\*All custom presets (1-3) are set to default values also.

## Common Commands

### **\*SAV** (Save)

**Command** \*SAV<specific setup>#

The \*SAV command stores the current settings to file. This setup is saved (and recalled) by specifying a specific setup number from 0 to 9.

<specific setup> An integer, 0-9, specifying the file used to save the current setup.

**Example** This example stores the current instrument settings to configuration file 6.

```
Send(0,4,"*SAV6",5,EOI);
```

## Common Commands

### **\*SRE (Service Request Enable)**

Command	<p><code>*SRE &lt;mask&gt;</code></p> <p>The <code>*SRE</code> command sets the Service Request Enable Register bits. The Service Request Enable Register contains a mask value for the bits to be enabled in the Status Byte Register.</p> <p>A one in the Service Request Enable Register (Figure 4.0) will enable the corresponding bit in the Status Byte Register, a zero will disable the bit.</p>
	<p><code>&lt;mask&gt;</code> An integer, 0-255, representing a mask value for the bits to be enabled in the Service Request Enable Register.</p>
Example	<p>This example enables a service request to be generated when a message is available in the output queue. When a message is available, the MAV bit will be high.</p> <pre>Send(0, 4, "*SRE16", 7, EOI);</pre>
Query	<p><code>*SRE?</code></p> <p>The <code>*SRE</code> query returns the current contents of the Service Request Enable register.</p>
Returned Format	<p><code>&lt;mask&gt;</code></p> <p><code>&lt;mask&gt;</code> An integer, 0-255, representing a mask value for the bits enabled in the service Request Enable Register.</p>
Example	<p>This example places the current contents of the Service Request Enable Register in the numeric variable, Enable, then prints the value of the variable to the screen.</p> <pre>Send(0, 4, "*SRE?", 5, EOI); Receive(0, 4, ENABLE, 1, EOI); Printf("%d\n", ENABLE);</pre>

## Common Commands

### **\*STB? (Status Byte)**

#### Query

\*STB?

The \*STB query returns the current value of the instrument's status byte. The MSS (Master Summary Status) bit and not RQS is reported on bit 6. The MSS indicates whether or not the device has at least one reason for requesting service. Refer to table 3-4 for the meaning of the bits in the status byte.

**Note:** To read the instrument's status byte with RQS reported on bit 6, use the GPIB Serial Poll.

#### Returned Format

<value>

<value> An integer, 0 through 255, representing a mask value for the bits enabled in the Status Byte

#### Example

This example reads the contents of the Status Byte into the numeric variable, Status, then prints the value of the variable to the screen.

```
Send(0,4,"*STB?",5,EOI);  
Receive(0,4,STATUS,1,EOI);  
Printf("%d\n",STATUS);
```



## Bus Commands

### **DCL (Device Clear)**

Command

DCL (ibclr)

The DCL command clears the input buffer and output queue, resets the parser and clears any pending commands.

## Device Commands

### **:DISPlay:PANel**

Command	<pre>:DISPlay:PANel 0 1 OFF ON</pre> <p>The PANel command sets the DTS550 remote / local operation. A '0' or 'OFF' sets the Remote operation (front panel is locked). A '1' or 'ON' sets the Local operation (front panel is operational).</p>
Example	<p>This command will put the DTS550 in remote operation (front panel locked.)</p> <pre>Send(0,4,":DISP:PAN OFF",13,EOI);</pre>
Query	<pre>:DISPlay:PANel?</pre> <p>The :DISPlay:PANel query returns the current value of the instrument's local / remote status.</p>
Example	<p>This command will return the operating mode (local/remote) of the DTS550 and print the result to the display.</p> <pre>char InBuffer[25]; Send(0,4,":DISP:PAN?",13,EOI); Receive(0,4,InBuffer,10,EOI); printf("%s\n", InBuffer);</pre>
Returned Format	0 1 (short form response) OFF ON (long form response)

## Device Commands

### **:JITTer:AMPLitude:Cumulative**

Command	<code>:JITTer:AMPLitude:Cumulative [UI] SEC DEG &lt;value&gt; MIN MAX</code>
	<p>This command sets the cumulative peak-to-peak jitter amplitude of the current jitter preset. The amplitude units may be sent in UI (default), Seconds or Degrees. The Minimum or Maximum value of jitter amplitude may be set by using the parameters 'MIN' or 'MAX' respectively. Jitter amplitude cumulative and periodic are coupled. (See also <code>:JITTer:AMPLitude:Periodic</code>)</p>
<code>[UI] SEC DEG</code>	Unit of measure of the jitter amplitude in UI(default), Seconds or Degrees.
<code>&lt;value&gt;</code>	Value of jitter amplitude in UI, Seconds or Degrees.
Example	<p>This example sets the Cumulative Jitter amplitude to 0.1 UI. Note that the default units (UI) was implied and may be omitted.</p> <pre>Send(0,4,":JITT:AMPL:C 0.1",16,EOI);</pre> <p>This example sets the Cumulative Jitter amplitude to the maximum (Degrees).</p> <pre>Send(0,4,":JITT:AMPL:C DEG MAX",16,EOI);</pre>
Query	<code>:JITT:AMPL:C [UI] SEC DEG ?  MIN?  MAX?</code>
	<p>The basic query will return the current cumulative jitter amplitude. Adding the parameters 'MIN' or 'MAX' will return the minimum or maximum jitter amplitude possible with the current instrument settings.</p>
Example	<p>This example will query the DTS550 for the maximum cumulative jitter in Degrees and print the result.</p> <pre>char InBuffer[50];  Send(0,4,":JITT:AMPL:C DEG MAX?",13,EOI); Receive(0,4,InBuffer,50,EOI); printf("%f\n", InBuffer);</pre>
Returned Format	Amplitude value in exponential format.

## Device Commands

### :JITTer:AMPLitude:Periodic

Command	<pre>:JITTer:AMPLitude:Periodic [UI] SEC DEG &lt;value&gt; MIN MAX</pre> <p>This command sets the periodic peak-to-peak jitter amplitude of the current jitter preset. The amplitude units may be sent in UI (default), Seconds or Degrees. The Minimum or Maximum value of jitter amplitude may be set by using the parameters 'MIN' or 'MAX' respectively. Jitter amplitude cumulative and periodic are coupled. (See also :JITTer:AMPLitude:Cumulative)</p>
[UI] SEC DEG	Unit of measure of the jitter amplitude in UI(default), Seconds or Degrees.
<value>	Value of jitter amplitude in UI, Seconds or Degrees.
Example	<p>This example sets the Periodic Jitter amplitude to 2.34e-09 seconds. (Short form shown.)</p> <pre>Send(0,4,":JITT:AMPL:P SEC 2.34e-9",24,EOI);</pre> <p>This example set the Periodic Jitter amplitude to the maximum (UI).</p> <pre>Send(0,4,":JITT:AMPL:P UI MAX",16,EOI);</pre>
Query	<pre>:JITT:AMPL:P [UI] SEC DEG ? MIN? MAX?</pre> <p>The basic query will return the current periodic jitter amplitude. Adding the parameters 'MIN' or 'MAX' will return the minimum or maximum jitter amplitude with the current instrument settings.</p>
Example	<p>This example will query the DTS550 for the periodic jitter in Seconds and print the result.</p> <pre>char InBuffer[50]; Send(0,4,":JITT:AMPL:P SEC?",17,EOI); Receive(0,4,InBuffer,50,EOI); printf("%f\n", InBuffer);</pre>
Returned Format	6.437884e-10 amplitude value in exponential format.

## Device Commands

### **:JITTer:DISTRibution**

Command	<pre>:JITTer:DISTRibution SIN SAW TRI SSC1 RAN FILE &lt;filename&gt;  1-5 6 &lt;filename&gt;</pre> <p>This command sets the jitter distribution for the current jitter preset. Valid distribution types are SIN (sine); SAW(sawtooth); TRI(triangle); SSC1 (Spread Spectrum Curve) and RAN(random). A custom jitter file may also be specified by 'FILE' and the .jmf (jitter memory file) filename (See Chapter 5). The distribution types may also be specified by numbers 1 through 6. (short form)</p>
Example	<p>This example sets the jitter distribution to the triangle distribution.</p> <pre>Send(0,4,":JITTer:DISTRibution TRI",25,EOI);</pre>
Query	<pre>:JITTer:DISTRibution?</pre> <p>The query will return the current jitter distribution.</p>
Example	<p>This example will query the DTS550 for the current jitter distribution and print the result.</p> <pre>char InBuffer[50];  Send(0,4,":JITTer:DISTRibution?",22,EOI); Receive(0,4,InBuffer,50,EOI); printf("%s\n", InBuffer);</pre>
Returned Format	<pre>SIN SAW TRI SSC1 RAN FILE &lt;filename.jmf&gt; (long form) 1 2 3 4 5 6 &lt;filename.jmf&gt; (short form).</pre>

## Device Commands

### **:JITTer:FREQuency**

**Command**            `:JITTer:FREQuency <value>|MAX|MIN`

This command sets the Jitter frequency of the current jitter preset. ‘MAX’ sets the maximum jitter frequency. ‘MIN’ sets the lowest jitter frequency. Jitter frequency and jitter period are coupled. (See also `:JITTer:PERiod.`)

`<value>`            The desired jitter frequency in Hz.

**Example**            This example sets the jitter frequency to 1500000Hz. (1.5 MHz)

```
Send(0,4," :JITTer:FREQuency 1.5e6",23,EOI);
```

**Query**              `:JITTer:FREQuency?|MIN?|MAX?`

The basic query will return the current jitter frequency for the current jitter preset. Adding the ‘MIN’ or ‘MAX’ parameters will return the minimum or maximum jitter frequency allowable.

**Example**            This example will query the DTS550 for the current jitter frequency and print the result.

```
char InBuffer[50];

Send(0,4," :JITTer:FREQuency?",18,EOI);
Receive(0,4,InBuffer,50,EOI);
printf("%f\n", InBuffer);
```

**Returned Format**    Frequency (Hz) value in exponential format.

## Device Commands

### **:JITTer:PERiod**

Command	<code>:JITTer:PERiod &lt;value&gt;   MAX   MIN</code>
	This command sets the Jitter period of the current jitter preset. 'MAX' sets the maximum jitter period. 'MIN' sets the lowest jitter period. Jitter period and jitter frequency are coupled. (See also <code>:JITTer:FREQuency</code> .)
<code>&lt;value&gt;</code>	The desired jitter period in Seconds.
Example	This example sets the jitter period to 1.25e-6. (0.00000125 Seconds)  <code>Send(0,4,":JITTer:PERiod 1.25e-6",22,EOI);</code>
Query	<code>:JITTer:PERiod?   MIN?   MAX?</code>  The basic query will return the current jitter period for the current jitter preset. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum jitter period allowable.
Example	This example will query the DTS550 for the current jitter period and print the result.  <code>char InBuffer[50];</code>  <code>Send(0,4,":JITTer:PERiod?",15,EOI);</code> <code>Receive(0,4,InBuffer,50,EOI);</code> <code>printf("%f\n", InBuffer);</code>
Returned Format	Period (Seconds) value in exponential format.

## Device Commands

### **:JITTer:PRESet**

Command	<pre>:JITTer:PRESet 1-5</pre> <p>This command sets the Jitter preset. Valid preset numbers are 1 – 5. The jitter amplitude; jitter frequency and jitter distribution associated with the selected preset will now take affect. The jitter state will remain unaffected.</p>
Example	<p>This example sets the jitter preset to 3.</p> <pre>Send(0,4,":JITTer:PRESet 3",16,EOI);</pre>
Query	<pre>:JITTer:PRESet?</pre> <p>The query will return the current jitter preset.</p>
Example	<p>This example will query the DTS550 for the current jitter preset and print the result.</p> <pre>char InBuffer[10];  Send(0,4,":JITTer:PRESet?",15,EOI); Receive(0,4,InBuffer,1,EOI); printf("%d\n", InBuffer);</pre>
Returned Format	ASCII number from 1 to 5.



## Device Commands

### **:JITTer:STATe**

Command	<code>:JITTer:STATe 0 1 OFF ON</code>  This command enables/disables the jitter. A '1' or 'ON' will enable the current jitter preset control values. A '0' or 'OFF' will disable jitter.
Example	This example enables the jitter.  <code>Send(0,4,":JITTer:STATe ON",16,EOI);</code>
Query	<code>:JITTer:STATe?</code>  The query will return the current jitter enable/disable state.
Example	This example will query the DTS550 for the current jitter enable/disable state and print the result.  <code>char InBuffer[10];  Send(0,4,":JITTer:STATe?",14,EOI); Receive(0,4,InBuffer,1,EOI); printf("%s\n", InBuffer);</code>
Returned Format	<code>0 1</code> (short form response) <code>OFF ON</code> (long form response)

## Device Commands

### **:OUTputs**

**Command**           :OUTputs 0|1|OFF|ON

This command enables/disables all outputs (main out, main out\_ and sync out.). A '1' or 'ON' will enable the outputs. A '0' or 'OFF' will disable the outputs.

**Example**           This example enables the outputs.

```
Send(0,4,":OUTputs ON",11,EOI);
```

**Query**             :OUTputs?

The query will return the current output enable/disable state.

**Example**           This example will query the DTS550 for the current output enable/disable state and print the result.

```
char InBuffer[10];  
  
Send(0,4,":OUTputs?",15,EOI);  
Receive(0,4,InBuffer,1,EOI);  
printf("%s\n", InBuffer);
```

**Returned Format**   0|1 (short form response) OFF|ON (long form response)

## Device Commands

### **:PULSe:FREQuency**

Command	<code>:PULSe:FREQuency &lt;value&gt;  MAX  MIN</code>  This command sets the main output / output_ clock frequency. 'MAX' sets the maximum frequency. 'MIN' sets the lowest frequency. (Pulse frequency and pulse period are coupled. See also :PULSe:PERiod.)
<value>	The main clock out frequency in Hz.
Example	This example sets the main clock to 450MHz .  <code>Send(0,4," :PULSe:FREQuency 450e6",22,EOI);</code>
Query	<code>:PULSe:FREQuency?  MIN?  MAX?</code>  The basic query will return the current main clock frequency in Hz. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum clock frequency.
Example	This example will query the DTS550 for the current clock frequency and print the result.  <code>char InBuffer[50];</code>  <code>Send(0,4," :PULSe:FREQuency?",17,EOI);</code> <code>Receive(0,4,InBuffer,50,EOI);</code> <code>printf("%f\n", InBuffer);</code>
Returned Format	Frequency (Hz) value in exponential format

## Device Commands

### :PULSe:LEVel:AMPLitude

Command           : PULSe:LEVel:AMPLitude <value> | MAX | MIN

This command sets the main output / output\_ amplitude level in the three custom output level presets (See also :PULSe:LEVel:PRESet). 'MAX' sets the maximum amplitude allowable with the current preset. 'MIN' sets the lowest amplitude. Pulse level amplitude, offset, high level and low level are coupled. Amplitude = high level – low level. (See also :PULSe:LEVel:OFFset :PULSe:LEVel:VHigh & :PULSe:LEVel:VLow). High and low limits will also affect this command operation when the limit function is enabled.

Programming sequence:

1. Select the desired custom (1-3) output level preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired amplitude value. (It may be necessary to change the offset, Vhigh or Vlow values first before setting the amplitude.)

<value>           The main clock out amplitude in Volts

Example           This example sets the main clock output amplitude to 0.8V .

```
Send(0,4,":PULSe:LEVel:AMPLitude 0.8",22,EOI);
```

Query             :PULSe:LEVel:AMPLitude? | MIN? | MAX?

The basic query will return the current output amplitude in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum output amplitude voltage.

Example           This example will query the DTS550 for the current output amplitude and print the result.

```
char InBuffer[50];  
  
Send(0,4,":PULSe:LEVel:AMPLitude?",23,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

Returned Format   Amplitude (Volts) value in exponential format

## Device Commands

### **:PULSe:LEVel:HighLIMit**

**Command**                    `:PULSe:LEVel:HighLIMit <value>|MAX|MIN`

This command sets the main output / output\_high limit level in the three custom output level presets (See also `:PULSe:LEVel:PRESet`). 'MAX' sets the maximum high level limit allowable with the current preset settings. 'MIN' sets the minimum high level limit. Pulse level high limit & low limit are coupled. The high limit setting will not be in effect unless the level limit is enabled. (See also `:PULSe:LEVel:LowLIMit` & `:PULSe:LEVel:LIMit`).

Programming sequence:

1. Select the desired custom (1-3) output level preset.
2. Disable the limit function.
3. Program the desired high limit value. (It may be necessary to change the low limit value first before setting the high limit.)

`<value>`                    The main clock output level high limit in volts.

**Example**                    This example sets the output level high limit to 1.2 Volts.

```
Send(0,4,":PULSe:LEVel:HighLIMit 1.2",26,EOI);
```

**Query**                      `:PULSe:LEVel:HighLIMit?|MIN?|MAX?`

The basic query will return the current output high limit in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum output level high limit voltage.

**Example**                    This example will query the DTS550 for the current level high limit and print the result.

```
char InBuffer[50];  
  
Send(0,4,":PULSe:LEVel:HighLIMit?",23,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

**Returned Format**        High limit (V) value in exponential format

## Device Commands

### **:PULSe:LEVel:LIMit**

Command

`:PULSe:LEVel:LIMit 0|1|OFF|ON`

This command enables/disables the output level limit function. A '1' or 'ON' will enable the limit function. A '0' or 'OFF' will disable the limit function. Enabling the limit function may affect the vhigh, vlow, offset & amplitude settings if their values exceed the high limit & low limit values. (See also `:PULSe:LEVel:HighLIMit` & `:PULSe:LEVel:LowLIMit`.)

Programming sequence (to enable limits):

1. Select the desired custom (1-3) output level preset.
2. Program the desired high limit & low limit values.
3. Enable the limit function. Output levels will conform to the high limit & low limit values.

Example

This example enables the limit function.

```
Send(0,4," :PULSe:LEVel:LIMit ON",21,EOI);
```

Query

`:PULSe:LEVel:LIMit?`

The query will return the current output level limit function state.

Example

This example will query the DTS550 for the current output limit function state and print the result.

```
char InBuffer[10];  
  
Send(0,4," :PULSe:LEVel:LIMit?",19,EOI);  
Receive(0,4,InBuffer,1,EOI);  
printf("%s\n", InBuffer);
```

Returned Format

`0|1` (short form response) `OFF|ON` (long form response)

## Device Commands

### **:PULSe:LEVel:LowLIMit**

**Command**                    :PULSe:LEVel:LowLIMit <value>|MAX|MIN

This command sets the main output / output\_low limit level in the three custom output level presets (See also :PULSe:LEVel:PRESet). 'MAX' sets the maximum low level limit allowable with the current preset settings. 'MIN' sets the minimum low level limit. Pulse level low limit & high limit are coupled. The low limit setting will not be in effect unless the level limit function is enabled. (See also :PULSe:LEVel:HighLIMit & :PULSe:LEVel:LIMit).

Programming sequence:

1. Select the desired custom (1-3) output level preset.
2. Disable the limit function.
3. Program the desired low limit value. (It may be necessary to change the high limit value first before setting the low limit.)

<value>                    The main clock output level low limit in volts.

**Example**                    This example sets the main clock output level low limit to -0.75 Volts.

```
Send(0,4," :PULSe:LEVel:LowLIMit -0.75",27,EOI);
```

**Query**                     :PULSe:LEVel:LowLIMit?|MIN?|MAX?

The basic query will return the current output\_low limit in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum output level low limit voltage.

**Example**                    This example will query the DTS550 for the current output level low limit and print the result.

```
char InBuffer[50];

Send(0,4," :PULSe:LEVel:LowLIMit?",23,EOI);
Receive(0,4,InBuffer,50,EOI);
printf("%f\n", InBuffer);
```

**Returned Format**        Low limit (V) value in exponential format

## Device Commands

### **:PULSe:LEVel:OFFSet**

**Command**                   : PULSe:LEVel:OFFSet <value>MAX|MIN

This command sets the main output / output\_offset level in the three custom output level presets (See also :PULSe:LEVel:PRESet). 'MAX' sets the maximum offset allowable with the current preset. 'MIN' sets the lowest offset. Pulse level amplitude, offset, high level and low level are coupled.  $\text{Offset} = (\text{high level} + \text{low level}) / 2$ . (See also :PULSe:LEVel:AMPLitude, :PULSe:LEVel:VHIgh & :PULSe:LEVel:VLOW). High and low limits will also affect this command operation when the limits are enabled.

Programing sequence:

1. Select the desired custom (1-3) output level preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired offset value. (It may be necessary to change the amplitude, vhigh or vlow values first before setting the offset.)

<value>                   The main clock out level offset in Volts

**Example**                   This example sets the main output level offset to 0.75V .

```
Send(0,4," :PULSe:LEVel:OFFSet 0.75",24,EOI);
```

**Query**                   : PULSe:LEVel:OFFSet?|MIN?|MAX?

The basic query will return the current main output level offset in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum clock output level offset voltage.

**Example**                   This example will query the DTS550 for the current output level offset and print the result.

```
char InBuffer[50];  
  
Send(0,4," :PULSe:LEVel:OFFSet?",20,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

**Returned Format**       Offset (Volts) value in exponential format.



## Device Commands

### **:PULSe:LEVel:PRESet**

Command	<pre>:PULSe:LEVel:PRESet ECLGND   ECL-2V   ECLOPEN   PECL+3V   PECLOPEN   TTLGND   TTLOPEN   CMOS3GND   CMOS3OPEN   CMOS5OPEN   CUST1   CUST2   CUST3   1-13</pre> <p>This command sets the main clock output level preset. Valid presets settings are listed in Table 4.1. Note that all other output level commands (Vhigh, Vlow, amplitude, offset, high limit, low limit &amp; limit enable) will be disabled when any of the non-custom presets are selected.</p>
Example	<p>This example sets the output level preset to ECLGND. (ECL voltage levels with 50 ohms to ground termination.)</p> <pre>Send(0,4," :PULSe:LEVel:PRESet ECLGND",26,EOI);</pre>
Query	<pre>:PULSe:LEVel:PRESet?</pre> <p>The query will return the current output level preset.</p>
Example	<p>This example will query the DTS550 for the current output level preset and print the result.</p> <pre>char InBuffer[10];  Send(0,4," :PULSe:LEVel:PRESet?",20,EOI); Receive(0,4,InBuffer,1,EOI); printf("%s\n", InBuffer);</pre>
Returned Format	<pre>ECLGND   ECL-2V   ECLOPEN   PECL+3V   PECLOPEN   TTLGND   TTLOPEN   CMOS3GND   CMOS3OPEN   CMOS5OPEN   CUST1   CUST2   CUST3 (long form response) 1-13 (short form response)</pre>

### Output Level Preset Settings (Fixed)

Preset Name  #	Termination	V high	V low	Amplitude	Offset	High limit	Low limit	Limit Enable
<b>ECLGND 1</b>	50Ω to GND	-0.90	-1.70	0.80	-1.30	NA	NA	NA
<b>ECL-2V 2</b>	50Ω to -2V	-0.90	-1.70	0.80	-1.30	NA	NA	NA
<b>ECLOPEN 3</b>	Open Circuit	-0.90	-1.70	0.80	-1.30	NA	NA	NA
<b>PECL+3V 4</b>	50Ω to +3V	4.20	3.20	1.00	3.70	NA	NA	NA
<b>PECLOPEN 5</b>	Open Circuit	4.20	3.20	1.00	3.70	NA	NA	NA
<b>TTLGND 6</b>	50Ω to GND	2.65	0.15	2.50	1.40	NA	NA	NA
<b>TTLOPEN 7</b>	Open Circuit	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CMOS3GND 8</b>	50Ω to GND	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CMOS3OPEN 9</b>	Open Circuit	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CMOS5OPEN 10</b>	Open Circuit	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CUST1 11</b> <b>CUST2 12</b> <b>CUST3 13</b>	Selectable: See Custom Output Level Preset Limits Table 4.2							

**Table 4.1 - Output Level Presets**

### Custom Preset Output Level Limits

<b>Level Parameter</b>	<b>Termination:</b> GND 1 (50Ω to GND) V-2  2 (50Ω to -2V) V+3 3 (50Ω to +3V)	Termination: OPEN 4 (Into Open Ckt)
<b>Vhigh</b>	Min: Vlow+0.25V Max: 2.50V	Min: Vlow + 0.50V Max: 5.00V
<b>Vlow</b>	Min: -2.00V Max: Vhigh-0.25V	Min: -4.00V Max: Vhigh + 0.50V
<b>Amplitude</b>	Min: 0.25V Max: 3.00V	Min: 0.50V Max: 6.00V
<b>Offset</b>	Min: -1.875V Max: 2.375V	Min: -3.75V Max: 4.75V
<b>Highlimit</b>	Min: Low limit + 0.25V Max: 2.50V	Min: Low limit + 0.50V Max: 5.00V
<b>Lowlimit</b>	Min: -2.00V Max: High limit - 0.25V	Min: -4.00V Max: High limit + 0.50V
<b>Limits Enable/Disable</b>	Functional	Functional

**Table 4.2 - Custom Output Level Limits**

## Device Commands

### **:PULSe:LEVel:TERMination**

Command	<code>:PULSe:LEVel:TERMination GND V-2 V+3 OPEN 1-4</code>  This command sets the output level termination when one of the 3 custom presets is selected. (See also <code>:PULSe:LEVel:PRESet</code> ). Valid termination settings are listed in the Custom Preset Output Limits.
Example	This example sets the output level termination to OPEN. (into Open circuit.)  <pre>Send(0,4," :PULSe:LEVel:TERMination OPEN",29,EOI);</pre>
Query	<code>:PULSe:LEVel:TERMination?</code>  The query will return the current output level termination.
Example	This example will query the DTS550 for the current output level termination and print the result.  <pre>char InBuffer[10];  Send(0,4," :PULSe:LEVel:TERMination?",25,EOI); Receive(0,4,InBuffer,1,EOI); printf("%s\n", InBuffer);</pre>
Returned Format	GND V-2 V+3 OPEN (long form response) 1-4 (short form response)

## Device Commands

### **:PULSe:LEVel:VHIgh**

**Command**                    :PULSe:LEVel:VHIgh <value> |MAX|MIN

This command sets the output / output\_ high level voltage in the three custom output level presets (See also :PULSe:LEVel:PRESet). 'MAX' sets the maximum high level allowable with the current preset. 'MIN' sets the lowest high level. Pulse level amplitude, offset, high level and low level are coupled. High level = Offset + (Amplitude/2) = Low level + Amplitude. (See also :PULSe:LEVel:OFFset, :PULSe:LEVel:AMPLitude; & :PULSe:LEVel:VLOW). High and low limits will also affect this command operation when the limit function is enabled.

Programing sequence:

1. Select the desired custom (1-3) output level preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired high level value. (It may be necessary to change the offset, amplitude or Vlow values first before setting the amplitude.)

<value>                    The main clock out high level in Volts

**Example**                    This example sets the main clock output high level to 0.75V .

```
Send(0,4," :PULSe:LEVel:VHIgh 0.75",23,EOI);
```

**Query**                    :PULSe:LEVel:VHIgh? |MIN? |MAX?

The basic query will return the current main clock output high level in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum clock output high level voltage.

**Example**                    This example will query the DTS550 for the clock output high level and print the result.

```
char InBuffer[50];

Send(0,4," :PULSe:LEVel:VHIgh?",19,EOI);
Receive(0,4,InBuffer,50,EOI);
printf("%f\n", InBuffer);
```

**Returned Format**        High level (Volts) value in exponential format

## Device Commands

### :PULSe:LEVel:VLOw

Command :PULSe:LEVel:VLOw <value> |MAX|MIN

This command sets the output / output\_ low level voltage in the three custom output level presets (See also :PULSe:LEVel:PRESet). 'MAX' sets the maximum low level allowable with the current preset. 'MIN' sets the lowest low level allowable. Pulse level amplitude, offset, low level and low level are coupled. Low level = Offset - (Amplitude/2) = High level --Amplitude. (See also :PULSe:LEVel:OFFset, :PULSe:LEVel:AMPLitude; & :PULSe:LEVel:VHIgh). High and low limits will also affect this command operation when the limit function is enabled.

Programing sequence:

1. Select the desired custom (1-3) output level preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired low level value. (It may be necessary to change the offset, amplitude or Vhigh values first before setting the amplitude.)

<value> The main clock out low level in Volts

Example This example sets the main clock output low level to -0.85V .

```
Send(0,4," :PULSe:LEVel:VLOw -0.85",23,EOI);
```

Query :PULSe:LEVel:VLOw? |MIN? |MAX?

The basic query will return the current main clock output low level in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum output low level voltage.

Example This example will query the DTS550 for the output low level and print the result.

```
char InBuffer[50];  
  
Send(0,4," :PULSe:LEVel:VLOw?",18,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

Returned Format Low level (Volts) value in exponential format

## Device Commands

### **:PULSe:PERiod**

**Command**            :PULSe:PERiod <value> |MAX|MIN

This command sets the main output / output\_clock period. 'MAX' sets the maximum period. 'MIN' sets the lowest period. (See also :PULSe:FREQuency.)

<value>            The main clock out period in Seconds.

**Example**            This example sets the main clock period to 2.2222e-9.

```
Send(0,4," :PULSe:PERiod 2.2222e-9",23,EOI);
```

**Query**              :PULSe:PERiod? |MIN? |MAX?

The basic query will return the current main clock period in Seconds. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum clock period.

**Example**            This example will query the DTS550 for the current clock period and print the result.

```
char InBuffer[50];  
  
Send(0,4," :PULSe:PERiod?",14,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

**Returned Format**    Period (Seconds) value in exponential format

## Device Commands

### **:PULSe:WIDTh**

Command	<code>:PULSe:WIDTh &lt;value&gt;   MAX   MIN</code>
	This command sets the main output / output_clock pulse width. 'MAX' sets the maximum pulse width. 'MIN' sets the lowest pulse width. (Pulse Duty cycle and pulse width are coupled. See also <code>:PULSe:DutCYCle</code> .)
<code>&lt;value&gt;</code>	The clock out pulse width in seconds.
Example	This example sets the main clock pulse width to 1.2e-9 seconds.  <code>Send(0,4," :PULSe:WIDTh 1.2e-9",19,EOI);</code>
Query	<code>:PULSe:WIDTh?   MIN?   MAX?</code>
	The basic query will return the current main clock pulse width. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum pulse width allowable for the current clock settings (See also <code>:PULSe:FREQuency</code> ).
Example	This example will query the DTS550 for the current clock pulse width and print the result.  <code>char InBuffer[50];</code>  <code>Send(0,4," :PULSe:WIDTh?",13,EOI);</code> <code>Receive(0,4,InBuffer,50,EOI);</code> <code>printf("%f\n", InBuffer);</code>
Returned Format	Pulse width (Seconds) value in exponential format.



## Device Commands

### **:RUN**

Command

`:RUN 0 | 1 | OFF | ON`

This command enables/disables the run state of the DTS550 (main out, main out\_ and sync out.). A '1' or 'ON' will enable the waveforms. A '0' or 'OFF' will disable the waveforms.

Example

This example enables the waveforms.

```
Send(0,4," :RUN ON",11,EOI);
```

Query

`:RUN?`

The query will return the current run enable/disable state.

Example

This example will query the DTS550 for the current run enable/disable state and print the result.

```
char InBuffer[10];  
  
Send(0,4," :RUN?",5,EOI);  
Receive(0,4,InBuffer,1,EOI);  
printf("%s\n", InBuffer);
```

Returned Format

`0 | 1 (short form response) OFF | ON (long form response)`

## Device Commands

### :SYNC:DIVide

Command           :SYNC:DIVide <value> |MAX|MIN

This command will set the sync output divide value when the sync mode is set to INDEpendent or BITclock mode. (See also :SYNC:MODE.) The sync divide will scale the sync frequency/period down by the divisor entered. (See table below for sync divisor ranges.)

<value>           The sync out divisor value integer (no units).

Example            This example sets the sync divisor to 4.

```
Send(0,4," :SYNC:DIVide 4",19,EOI);
```

Query              :SYNC:DIVide? |MIN? |MAX?

The basic query will return the current sync divide value. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync divide allowable for the current settings.

Example            This example will query the DTS550 for the current sync divide value and print the result.

```
char InBuffer[50];

Send(0,4," :SYNC:DIVide?",13,EOI);
Receive(0,4,InBuffer,50,EOI);
printf("%f\n", InBuffer);
```

Returned Format    Sync divide, integer value.

SYNC Mode	Divisor values
INDEpendent	1-255 & 256-4080 (mod 16)
BITClock -(Pulse freq <= 531.25Mhz)	1-255 & 256-4080(mod 16)
BITClock -(Pulse freq > 531.25Mhz)	2-254 (mod 2) & 256-4080(mod 16)
JITter mode	NA

**Table 4.3 - Sync Mode**

## Device Commands

### **:SYNC:FREQuency**

Command	<code>:SYNC:FREQuency &lt;value&gt;MAX MIN</code>  This command sets the sync output frequency when the sync mode is set to INDPendent. (See also <code>:SYNC:MODE</code> ). 'MAX' sets the maximum frequency. 'MIN' sets the lowest frequency. (Sync frequency and sync period are coupled. See also <code>:SYNC:PERiod</code> .)
<value>	The sync out frequency in Hz.
Example	This example sets the sync output to 450MHz .  <code>Send(0,4," :SYNC:FREQuency 450e6",22,EOI);</code>
Query	<code>:SYNC:FREQuency? MIN? MAX?</code>  The basic query will return the current sync frequency in Hz. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync frequency.
Example	This example will query the DTS550 for the current sync frequency and print the result.  <code>char InBuffer[50];  Send(0,4," :SYNC:FREQuency?",16,EOI); Receive(0,4,InBuffer,50,EOI); printf("%f\n", InBuffer);</code>
Returned Format	Frequency (Hz) value in exponential format.

## Device Commands

### :SYNC:LEVel:AMPLitude

Command           :SYNC:LEVel:AMPLitude <value>|MAX|MIN

This command sets the sync amplitude level in the three custom sync level presets (See also :SYNC:LEVel:PRESet). 'MAX' sets the maximum amplitude allowable with the current preset. 'MIN' sets the lowest amplitude. Sync level amplitude, offset, high level and low level are coupled. Amplitude = high level – low level. (See also :SYNC:LEVel:OFFset, :SYNC:LEVel:VHigh; & :SYNC:LEVel:VLOw). High and low limits will also affect this command operation when the limit function is enabled.

Programming sequence:

1. Select the desired custom (1-3) sync level preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired amplitude value. (It may be necessary to change the offset, Vhigh or Vlow values first before setting the amplitude.)

<value>           The sync out amplitude in Volts

Example            This example sets the sync output amplitude to 0.8V .

```
Send(0,4,":SYNC:LEVel:AMPLitude 0.8",25,EOI);
```

Query             :SYNC:LEVel:AMPLitude?|MIN?|MAX?

The basic query will return the sync output amplitude in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync output amplitude voltage.

Example            This example will query the DTS550 for the current sync output amplitude and print the result.

```
char InBuffer[50];  
  
Send(0,4,":SYNC:LEVel:AMPLitude?",22,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

Returned Format    Amplitude (Volts) value in exponential format.

## Device Commands

### **:SYNC:LEVel:HighLIMit**

**Command**            `:SYNC:LEVel:HighLIMit <value>|MAX|MIN`

This command sets the sync high limit level in the three custom sync level presets (See also `:SYNC:LEVel:PRESet`). ‘MAX’ sets the maximum high level limit allowable with the current preset settings. ‘MIN’ sets the minimum high level limit. Sync level high limit & low limit are coupled. The high limit setting will not be in effect unless the level limit is enabled. (See also `:SYNC:LEVel:LowLIMit` & `:SYNC:LEVel:LIMit`).

Programming sequence:

1. Select the desired custom (1-3) sync level preset.
2. Disable the limit function.
3. Program the desired high limit value. (It may be necessary to change the low limit value first before setting the high limit.)

`<value>`            The sync high limit in volts.

**Example**            This example sets the sync level high limit to 1.2 Volts.

```
Send(0,4,":SYNC:LEVel:HighLIMit 1.2",25,EOI);
```

**Query**              `:SYNC:LEVel:HighLIMit?|MIN?|MAX?`

The basic query will return the current sync high limit in volts. Adding ‘MIN’ or ‘MAX’ parameters will return the minimum or maximum sync high limit voltage.

**Example**            This example will query the DTS550 for the current sync high limit and print the result.

```
char InBuffer[50];

Send(0,4,":SYNC:LEVel:HighLIMit?",23,EOI);
Receive(0,4,InBuffer,50,EOI);
printf("%f\n", InBuffer);
```

**Returned Format**    High limit (V) value in exponential format

## Device Commands

### **:SYNC:LEVel:LIMit**

Command	<pre>:SYNC:LEVel:LIMit 0 1 OFF ON</pre> <p>This command enables/disables the sync level limit function. A '1' or 'ON' will enable the limit function. A '0' or 'OFF' will disable the limit function. Enabling the limit function may affect the Vhigh, Vlow, offset &amp; amplitude settings if their values exceed the high limit &amp; low limit values. (See also :SYNC:LEVel:HighLIMit &amp; :SYNC:LEVel:LowLIMit.)</p> <p>Programming sequence (to enable limits):</p> <ol style="list-style-type: none"><li>1. Select the desired custom (1-3) sync level preset.</li><li>2. Program the desired high limit &amp; low limit values.</li><li>3. Enable the limit function. Sync levels will conform to the high limit &amp; low limit values.</li></ol>
Example	<p>This example enables the sync limit function.</p> <pre>Send(0,4,":SYNC:LEVel:LIMit ON",20,EOI);</pre>
Query	<pre>:SYNC:LEVel:LIMit?</pre> <p>The query will return the current sync level limit function state.</p>
Example	<p>This example will query the DTS550 for the current sync limit function state and print the result.</p> <pre>char InBuffer[10];  Send(0,4,":SYNC:LEVel:LIMit?",18,EOI); Receive(0,4,InBuffer,1,EOI); printf("%s\n", InBuffer);</pre>
Returned Format	<pre>0 1 (short form response) OFF ON (long form response)</pre>

## Device Commands

### **:SYNC:LEVel:LowLIMit**

**Command**            `:SYNC:LEVel:LowLIMit <value> |MAX|MIN`

This command sets the sync low limit level in the three custom sync level presets (See also `:SYNC:LEVel:PRESet`). ‘MAX’ sets the maximum low level limit allowable with the current preset settings. ‘MIN’ sets the minimum low level limit. Pulse level low limit & high limit are coupled. The low limit setting will not be in effect unless the level limit function is enabled. (See also `:SYNC:LEVel:HighLIMit` & `:SYNC:LEVel:LIMit`).

Programming sequence:

1. Select the desired custom (1-3) sync level preset.
2. Disable the limit function.
3. Program the desired low limit value. (It may be necessary to change the high limit value first before setting the low limit.)

`<value>`            The sync low limit in volts.

**Example**            This example sets the sync low limit to  $-0.75$  Volts.

```
Send(0,4," :SYNC:LEVel:LowLIMit -0.75",26,EOI);
```

**Query**              `:SYNC:LEVel:LowLIMit? |MIN? |MAX?`

The basic query will return the current sync low limit in volts. Adding ‘MIN’ or ‘MAX’ parameters will return the minimum or maximum sync low limit voltage.

**Example**            This example will query the DTS550 for the current sync low limit and print the result.

```
char InBuffer[50];

Send(0,4," :SYNC:LEVel:LowLIMit?",21,EOI);
Receive(0,4,InBuffer,50,EOI);
printf("%f\n", InBuffer);
```

**Returned Format**    Low limit (V) response in exponential format.

## Device Commands

### :SYNC:LEVel:OFFSet

#### Command

:SYNC:LEVel:OFFSet <value>MAX|MIN

This command sets the sync offset in the three custom sync level presets (See also :SYNC:LEVel:PRESet). 'MAX' sets the maximum offset allowable with the current preset. 'MIN' sets the lowest offset. Pulse level amplitude, offset, high level and low level are coupled.  $\text{Offset} = (\text{high level} + \text{low level}) / 2$ . (See also :SYNC:LEVel:AMPLitude, :SYNC:LEVel: VHIgh; & :SYNC:LEVel:VLOW). High and low limits will also affect this command operation when the limits are enabled.

Programming sequence:

1. Select the desired custom (1-3) sync preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired offset value. (It may be necessary to change the amplitude, vhigh or vlow values first before setting the offset.)

<value> The sync offset in Volts

#### Example

This example sets the sync offset to 0.75V .

```
Send(0,4," :SYNC:LEVel:OFFSet 0.75",23,EOI);
```

#### Query

:SYNC:LEVel:OFFSet?|MIN?|MAX?

The basic query will return the current sync offset in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync level offset voltage.

#### Example

This example will query the DTS550 for the current sync offset and print the result.

```
char InBuffer[50];  
  
Send(0,4," :SYNC:LEVel:OFFSet?",19,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

#### Returned Format

Offset (Volts) value in exponential format.



## Device Commands

### **:SYNC:LEVel:PRESet**

**Command**

```
:SYNC:LEVel:PRESet ECL-2V|ECLOPEN|PECL+3V|  
PECLOPEN|TTLGND|TTLOPEN|CMOS3GND|CMOS3OPEN|  
CMOS5OPEN|CUST1|CUST2|CUST3|1-12
```

This command sets the sync level preset. Valid presets settings are listed in Table 4.4 . Note that all other sync level commands (Vhigh, Vlow, amplitude, offset, high limit, low limit & limit enable) will be disabled when any of the non-custom presets are selected.

**Example**

This example sets the sync level preset to CUST1. (Custom preset #1)

```
Send(0,4," :SYNC:LEVel:PRESet CUST1",24,EOI);
```

**Query**

```
:SYNC:LEVel:PRESet?
```

The query will return the current sync level preset.

**Example**

This example will query the DTS550 for the current sync level preset and print the result.

```
char InBuffer[10];  
  
Send(0,4," :SYNC:LEVel:PRESet?",19,EOI);  
Receive(0,4,InBuffer,1,EOI);  
printf("%s\n", InBuffer);
```

**Returned Format**

```
ECL-2V|ECLOPEN|PECL+3V|PECLOPEN|TTLGND|TTLOPEN  
|CMOS3GND|CMOS3OPEN|CMOS5OPEN|CUST1|CUST2|CUST3  
(long form response) 1-12 (short form response)
```

### Sync Level Preset Settings (Fixed)

Preset Name  #	Termination	V high	V low	Amplitude	Offset	High limit	Low limit	Limit Enable
<b>ECL-2V 1</b>	50Ω to -2V	-0.90	-1.70	0.80	-1.30	NA	NA	NA
<b>ECLOPEN 2</b>	Open Circuit	-0.90	-1.70	0.80	-1.30	NA	NA	NA
<b>PECL+3V 3</b>	50Ω to +3V	4.20	3.20	1.00	3.70	NA	NA	NA
<b>PECLOPEN 4</b>	Open Circuit	4.20	3.20	1.00	3.70	NA	NA	NA
<b>TTLGND 5</b>	50 Ω to GND	2.65	0.15	2.50	1.40	NA	NA	NA
<b>TTLOPEN 6</b>	Open Circuit	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CMOS3GND 7</b>	50 Ω to GND	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CMOS3OPEN 8</b>	Open Circuit	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CMOS5OPEN 9</b>	Open Circuit	2.65	0.15	2.50	1.40	NA	NA	NA
<b>CUST1 10</b> <b>CUST2 11</b> <b>CUST3 12</b>	Selectable: See Custom Sync Level Preset Limits Table 4.5							

**Table 4.4 - Sync Level Preset Limits**

## Custom Preset Sync Level Limits

<b>Level Parameter</b>	<b>Termination:</b> 50Ω to GND 50Ω to -2V 50Ω to +3V	<b>Termination:</b> Into Open Circuit
<b>Vhigh</b>	Min: Vlow+0.25V Max: 2.50V	Min: Vlow + 0.50V Max: 5.00V
<b>Vlow</b>	Min: -1.00V Max: Vhigh-0.25V	Min: -2.00V Max: Vhigh + 0.50V
<b>Amplitude</b>	Min: 0.25V Max: 3.00V	Min: 0.50V Max: 6.00V
<b>Offset</b>	Min: -1.875V Max: 2.375V	Min: -3.75V Max: 4.75V
<b>Highlimit</b>	Min: Low limit + 0.25V Max: 2.50V	Min: Low limit + 0.50V Max: 5.00V
<b>Lowlimit</b>	Min: -2.00V Max: High limit - 0.25V	Min: -4.00V Max: High limit + 0.50V
<b>Limits Enable/Disable</b>	Functional	Functional

**Table 4.5 - Custom Preset Sync Level Limits**

## Device Commands

### **:SYNC:LEVel:TERMination**

Command	<code>:SYNC:LEVel:TERMination GND V-2 V+3 OPEN 1-4</code>  This command sets the sync termination when one of the 3 custom presets is selected. (See also <code>:SYNC:LEVel:PRESet</code> ). Valid termination settings are listed in the Custom Preset Sync Limits.
Example	This example sets the sync termination to OPEN. (into Open circuit.)  <code>Send(0,4,":SYNC:LEVel:TERMination OPEN",28,EOI);</code>
Query	<code>:SYNC:LEVel:TERMination?</code>  The query will return the current sync termination.
Example	This example will query the DTS550 for the current sync termination and print the result.  <code>char InBuffer[10];</code>  <code>Send(0,4,":SYNC:LEVel:TERMination?",25,EOI);</code> <code>Receive(0,4,InBuffer,1,EOI);</code> <code>printf("%s\n", InBuffer);</code>
Returned Format	<code>GND V-2 V+3 OPEN</code> (long form response) <code>1-4</code> (short form response)

## Device Commands

### :SYNC:LEVel:VHIgh

Command :SYNC:LEVel:VHIgh <value>|MAX|MIN

This command sets the sync high voltage in the three custom sync level presets (See also :SYNC:LEVel:PRESet). 'MAX' sets the maximum high level allowable with the current preset. 'MIN' sets the lowest high level. Sync amplitude, offset, high level and low level are coupled. High level = Offset + (Amplitude/2) = Low level + Amplitude. (See also :SYNC:LEVel:OFFset, :SYNC:LEVel:AMPLitude; & :SYNC:LEVel:VLOW). High and low limits will also affect this command operation when the limit function is enabled.

Programming sequence:

1. Select the desired custom (1-3) sync preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired high level value. (It may be necessary to change the offset, amplitude or Vlow values first before setting the amplitude.)

<value> The sync high level in Volts

Example This example sets the sync high level to 0.75V .

```
Send(0,4,":SYNC:LEVel:VHIgh 0.75",22,EOI);
```

Query :SYNC:LEVel:VHIgh?|MIN?|MAX?

The basic query will return the current sync high level in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync high level voltage.

Example This example will query the DTS550 for the sync high level and print the result.

```
char InBuffer[50];  
  
Send(0,4,":SYNC:LEVel:VHIgh?",18,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

Returned Format High level (Volts) response in exponential format.

## Device Commands

### SYNC:LEVel:VLOW

Command `:SYNC:LEVel:VLOW <value>|MAX|MIN`

This command sets the sync low level voltage in the three custom sync level presets (See also `:SYNC:LEVel:PRESet`). 'MAX' sets the maximum low level allowable with the current preset. 'MIN' sets the lowest low level allowable. Pulse level amplitude, offset, low level and low level are coupled.  $\text{Low level} = \text{Offset} - (\text{Amplitude}/2) = \text{High level} - \text{Amplitude}$ . (See also `SYNC:LEVel:OFFSet`, `SYNC:LEVel:AMPLitude` & `SYNC:LEVel:VHigh`). High and low limits will also affect this command operation when the limit function is enabled.

Programming sequence:

1. Select the desired custom (1-3) sync level preset.
2. Disable the limit function if enabled. (Leave enabled if limit function desired.)
3. Program the desired low level value. (It may be necessary to change the offset, amplitude or Vhigh values first before setting the amplitude.)

<value> The sync low level in Volts

Example This example sets the sync low level to -0.85V .

```
Send(0,4,"SYNC:LEVel:VLOW -0.85",22,EOI);
```

Query `SYNC:LEVel:VLOW?|MIN?|MAX?`

The basic query will return the current sync low level in volts. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync low level voltage.

Example This example will query the DTS550 for the sync low level and print the result.

```
char InBuffer[50];  
  
Send(0,4,"SYNC:LEVel:VLOW?",17,EOI);  
Receive(0,4,InBuffer,50,EOI);  
printf("%f\n", InBuffer);
```

Returned Format Low level (Volts) value in exponential format.

## Device Commands

### **:SYNC:MODE**

Command	<code>:SYNC:MODE JITter BITclock INdependent 2-4</code>  This command sets the sync mode operation. JITter mode=sync matches programmed jitter frequency; BITclock mode=follows main clock output frequency (with divide function); INdependent mode=operates independently from main output (with divide function.)
Example	This example sets the sync mode to independent mode.  <code>Send(0,4,":SYNC:MODE INdependent",22,EOI);</code>
Query	<code>:SYNC:MODE?</code>  The query will return the current sync mode.
Example	This example will query the DTS550 for the sync mode and print the result.  <code>char InBuffer[20];  Send(0,4,":SYNC:MODE?",15,EOI); Receive(0,4,InBuffer,1,EOI); printf("%s\n", InBuffer);</code>
Returned Format	JITter BITclock INdependent (long form response) 2-4 (short form response)

## Device Commands

### **:SYNC:PERiod**

Command	<code>:SYNC:PERiod &lt;value&gt; MAX MIN</code>  This command sets the sync period when the sync mode is set to INDpendent mode. 'MAX' sets the maximum period. 'MIN' sets the lowest period. (See also <code>:SYNC:FREQuency</code> .)  <value> The sync period in seconds.
Example	This example sets the sync period to 2.2222e-9 seconds.  <code>Send(0,4," :SYNC:PERiod 2.2222e-9",22,EOI);</code>
Query	<code>:SYNC:PERiod? MIN? MAX?</code>  The basic query will return the current sync period in Seconds. Adding 'MIN' or 'MAX' parameters will return the minimum or maximum sync period.
Example	This example will query the DTS550 for the current sync period and print the result.  <code>char InBuffer[50];  Send(0,4," :SYNC:PERiod?",14,EOI); Receive(0,4,InBuffer,50,EOI); printf("%f\n", InBuffer);</code>
Returned Format	Period (Seconds) value in exponential format.



## Device Commands

### **:SYSTem:HEADer**

Command	<code>:SYSTem:HEADer 0 1 OFF ON</code>  This command enables/disables the header being returned in a response from the DTS550.
Example	This example disables the headers from being placed in responses to queries.  <code>Send(0,4,":SYSTem:HEADer OFF",18,EOI);</code>
Query	<code>:SYSTem:HEADer?</code>  The basic query will return the header enabled status.
Example	This example will query the DTS550 for the current header status and print the result.  <code>char InBuffer[50];  Send(0,4,":SYSTem:HEADer?",15,EOI); Receive(0,4,InBuffer,50,EOI); printf("%f\n", InBuffer);</code>
Returned Format	<code>0 1</code> (short form response) <code>OFF ON</code> (long form response)

## Device Commands

### **:SYSTem:LONGform**

Command	<code>:SYSTem:LONGform 0 1 OFF ON</code>  This command selects the long or short form of the header being returned in a response from the DTS550.
Example	This example enables the longform of the headers in responses to queries.  <code>Send(0,4,":SYSTem:LONGform ON",19,EOI);</code>
Query	<code>:SYSTem:LONGform?</code>  The basic query will return the header longform status.
Example	This example will query the DTS550 for the current longform status and print the result.  <code>char InBuffer[50];  Send(0,4,":SYSTem:LONGform?",16,EOI); Receive(0,4,InBuffer,50,EOI); printf("%f\n", InBuffer);</code>
Returned Format	<code>0 1</code> (short form response) <code>OFF ON</code> (long form response)

## Device Commands

### **:SYSTem:SVER?**

Query

:SYSTem:SVER?

The command queries the DTS550 for the installed firmware date and time.

Example

This example will query the DTS550 for the current firmware date & time and print the result.

```
char InBuffer[50];  
  
Send(0,4,":SYSTem:SVER?",19,EOI);  
Receive(0,4,InBuffer,1,EOI);  
printf("%s\n", InBuffer);
```

Returned Format

ASCII text

## **CHAPTER 5 - ADVANCED FEATURES**

### **Creating A Jitter Memory File**

A jitter memory file (JMF extension) is used to specify a custom jitter distribution for a particular waveform or pattern. The JMF file only specifies the jitter distribution, not the jitter amplitude or frequency. These attributes are defined in the Select Output Jitter window.

The JMF file is prepared in much the same way as a pattern memory file. First, a jitter text file is prepared by creating a standard text file containing a series of real numbers between -1.0 and +1.0. These numbers represent a percentage (where 1.0 = 100%) of maximum jitter amplitude at a particular slice of time. For example, to create an 8-step phase shift with a triangle distribution, the following would be entered into a standard text file:

```
0.0
+0.5
+1.0
+0.5
0.0
-0.5
-1.0
-0.5
```

For this example, the distribution would vary from 0 to 100% of the amplitude specified in the Select Output Jitter window.

Jitter memory in the DTS-550 is 4K vectors deep, where each vector corresponds to one point in the data text file. The number of data points in a jitter text file must be a power of two (2, 4, 8, 16...), but can number no more than 4K (4096), the size of the jitter memory. In addition, if the text file contains fewer than 4K data points, the data is repeated until jitter memory in the DTS is filled. Jitter memory is scanned once every period as indicated in the Select Output Jitter window. Therefore, if only 8 points are in a jitter text file, this distribution pattern will be repeated 512 times, one after another, in memory. The effective frequency of this pattern is actually 512 times higher than the frequency listed in the Select Output Jitter window. This “frequency scaling” can be useful when the user wants to exceed the 5MHz limit imposed in the Jitter Frequency field.

To compile the data, switch to the directory containing the source text file and use the following command (note that the /A argument is optional):

```
JMC/F [/A] <input filename> <output filename>.JMF
```

While compiling, messages will appear on the display indicating the status of the compile process. Pay particular attention to any error messages that may occur. If the utility is run from a directory other than the one containing the source text file, pathnames will need to be included as part of the input and output filenames.

To apply the new jitter distribution to a waveform or data pattern, select **<From File>** in the **Distribution** field of the **Select Output Jitter** window. This will bring up the **Select Jitter File** window. Highlight the directory and filename containing the desired jitter distribution, then click on **Select**. The selected filename should now appear in the **Jitter File** field of the **Select Output Jitter** window.

## **CHAPTER 6 - AN APPLICATION EXAMPLE**

This chapter walks the user step-by-step through a typical application example for the DTS-550. Setup files, scope settings and all other necessary setup information is provided so that the user can focus on the functionality of the instrument, not details, regarding the setup for this particular example.

The following equipment will be necessary for this example:

<b>Qty</b>	<b>Description</b>	<b>Comments</b>
1	DTS-550 DTS Synthesizer	Only one output channel necessary
1	Digitizing Oscilloscope	Should have 1GHz or higher sampling frequency
2	Cables	SMA plug on one end, other end to mate with scope input channels

### **Description of Setup Information**

Freq = 106.25MHz

Duty Cycle = 60%

Sync Mode = Bit Clock,

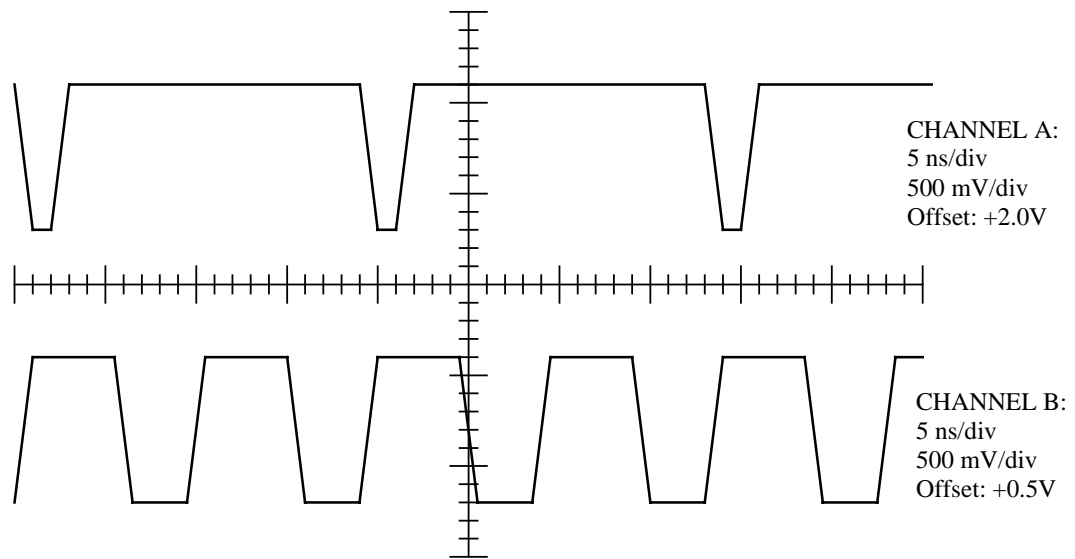
Sync Divide = 2

Jitter On/Off = Off

Output Levels = CUSTOM1: 50Ω to GND

- 1) First run one cable from the SYNC OUT connector (SMA female connector) on Channel 1 to one of the data input channels (hereafter referred to as CH A) on the oscilloscope. This signal will be used to trigger the scope but it will also be helpful to have the ability to view the waveform. Therefore, use a data input channel and not a trigger input.
- 2) Run the other cable from the OUTPUT connector (SMA female connector) on Channel 1 of the CLOCK/DATA card to another input channel (hereafter referred to as CH B) on the oscilloscope.
- 3) Power up both the scope and the DTS-550.
- 4) Adjust the scope for a timebase of 5ns/div and a signal level of 500mV/div on both CH A and CH B. Set the scope to trigger on positive going edges on CH A.
- 5) Adjust the scope GND references for CH A and CH B so that both traces can be viewed simultaneously on separate areas of the screen.
- 6) Select Default Setup from the File <ALT-F> menu selection. This command will restore all the default settings for the DTS550 controls.
- 7) On the main display, set the **Frequency** and **DC%** controls to 106.25 (MHz) and 60 (%) respectively.

- 8) Toggle the **All Outputs (DISABLE ALL or <Ctrl-A>)** and **Synthesizer (RUN or <Ctrl-R>)** controls to **Enabled** and **Running**. Clock-type waveforms similar to the ones shown in Figure 5.1 should now be seen on the scope. Note that the waveforms in Figure 5.1 are idealized and contain none of the overshoot or ringing that will be seen on the actual output. Make any adjustments necessary on the scope so that several cycles of both waveforms can be seen.



**Figure 6.1 - Example waveforms**

- 9) Push the **PERIOD** button on the DTS front panel. The input cursor will jump to the **Frequency/Period** control and the user will be allowed to modify the value in this field. Change the number in this field to 200 (MHz), then press the **ENTER** key. To view the main frequency as period rather than as frequency, set the **Clock Freq. Units** control value to **Period** (ns) by using the **UP** and **DN** buttons on the main screen or the **STEP UP** and **DOWN** controls on the front panel. The clock period now will read 5.00 (ns). In addition, the **Width** (ns) field will change to hold the **DC%** constant. The output seen on the scope should now match this new frequency. Adjust the timebase on the scope if necessary to view several cycles of the output waveform.
- 10) To view the clock **Width/DC%** as a function of time, change the **Clock PW Units** control to the right of the **Width/DC%** control to **Width**. The Clock pulse width will now display 3.0 (ns).
- 11) To see how the **Period Step** function works, either click on the **UP** and **DN** icons on the main screen with the mouse, or push the **STEP UP** and **DOWN** keys on the front panel. This will increment or decrement the value in the **Period** (ns) field by the amount shown in the **Period Step** field. Before proceeding, set the period back to 9.41176ns (106.25MHz frequency).

- 12) Push the **WIDTH** button on the front panel. The input cursor will jump to the **Width** (ns) field which can now be modified. Change the value in this field to 3.0, then press **ENTER**. Note that the **Width** field changed to reflect the new value. View the scope to see that the pulse width of the OUTPUT signal has been decreased.
- 13) At this time, the step function should be labeled **Width Step**. Step the value in the **Width** (ns) field up and down to see how this affects the output. Before proceeding, change the pulse width back to 5.647ns (60% duty cycle).
- 14) Now try applying some phase modulation (jitter) to the output waveform. To do this, press the **Jitter** (or **F6**) button on the front panel. This will select the **Jitter Period Amp** control.
- 15) There are two means of defining jitter amplitude presented in this window. They are related and changing one value will affect the other. Use the **DATA ENTRY** or external keyboard keys to change the value in the amplitude control to 0.05 UI (unit interval). The jitter amplitude may also be view in Ns or degrees by changing the **Jitter Ampl. Units** control. Note that changing the Period Jitter will also change the **Cumulative Amp** value as well. The **Jitter Ampl. Units** control setting affects the Cumulative Amp. units as well.
- 16) Jitter frequency is by the **Freq. (Hz) / Period (sec)** field. Select **Freq./Period** and set its value to 0.5 (Hz). To view the jitter frequency as a period change the **Jitter Freq Units** control value to **Period (Hz)**.
- 17) Click on the **Distribution** field and select **Sine**. (Sine should be selected already.). This will cause the jitter to vary with a sinusoidal distribution between the minimum and maximum amplitude values at the jitter frequency selected. Other distribution types may be selected by using the **UP/DN** buttons or **<Ctrl-U>** and **<Ctrl-N>** .
- 18) Click on the **Jitter Off/Jitter On** field and set it to **Jitter On**. This will enable jitter to be applied to the output. Click on **OK** to exit the **Select Output Jitter** window. You should now see the OUTPUT signal vary with respect to the SYNC OUT signal in the manner defined by the jitter setup.
- 19) To save this instrument setup as a setup file that may be recalled later, select 'Save Setup' from the 'File' menu. When the screen asking for a Select Instrument State file name appears, enter a '1' from the keypad / external keyboard. It is not necessary to type the '.sav' to the end of the desired name. Press the Enter key or 'Select' on the screen. The current instrument settings have now been saved to the file '1.sav'. These settings may be recalled at another time with the 'File-Recall Setup' command.



This page intentionally left blank.

# APPENDIX A – DTS-550/550A SPECIFICATIONS

## Clock/Data Outputs – Output, $\overline{\text{Output}}$

### Timing Characteristics

#### Period

**Programmable Range** ..... 952.4ps – 100ns (1050MHz - 10MHz)

**Programming Resolution** ..... 25 bits resolution (0.015ps)

**Accuracy** .....  $\pm 50$  ppm (0.005%) of clock period

#### Width

**Range**..... 250ps to ( Period – 250ps)

**Resolution** ..... 4ps

#### Jitter (50MHz to 1000MHz, 50% duty cycle)

**1Sigma**..... 25 ps max, 10 ps typical

**Pk-to-pk** ..... 150 ps max

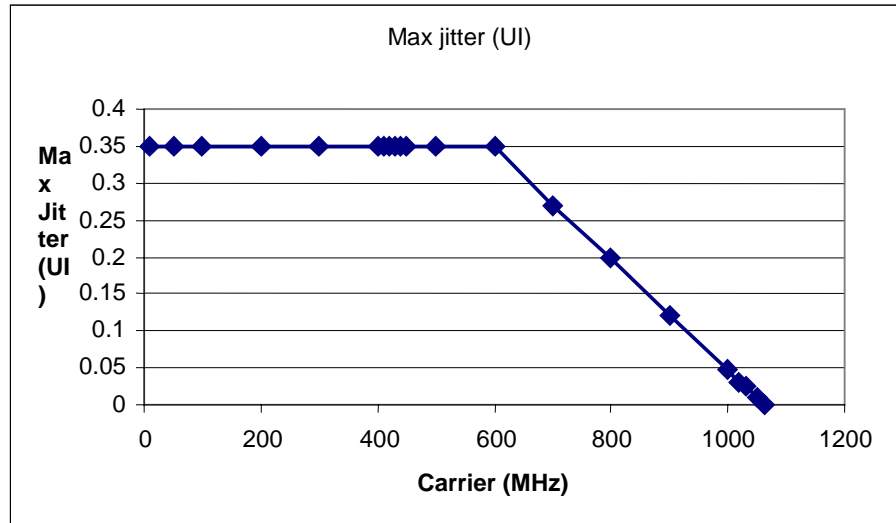
#### Programmable Jitter

**Programmable Jitter Period** ..... 200ns to 67s (5MHz to 0.015Hz)

**Jitter Programming Resolution** ..... 0.015Hz

**Programmable Jitter Distributions** ..... Sine, Sawtooth, Triangle, SSC1, Pseudo Random, User Defined.

### Maximum Jitter Amplitude



**Jitter Amp. Resolution**..... 20 bit resolution (4 ps)

**Cumulative Jitter Amp. Accuracy** ..... 50 ppm(0.005%) of clock period +/- Jitter edge placement

<b>Jitter Edge Placement</b> (50 MHz to 1000 MHz, 50% duty cycle)	
1Sigma .....	25 ps max, 10 ps typical
Pk-to-pk.....	150 ps max
Jitter Update Rate .....	70megaSamples/s (max)
Jitter Data Length.....	4096 words

**Output Level**

Selectable Termination.....	50Ω to GND, 50Ω to -2V 50Ω to +3V open circuit
-----------------------------	---

**Amplitude**

Preset Levels.....	ECL, PECL, TTL, 3.3V CMOS, 5V CMOS
Swing.....	0.25V min: +3.0V max. (50Ω) 0.5V min: +6.0V max. (open)
Max. High Level.....	+2.5V (50Ω), + 5.0V (open) +2.25V (50Ω) above 800MHz
Min. Low Level .....	-2.0V (50Ω), -4.0V (open)
Resolution.....	20mV (50Ω), 40mV (open)
Swing Accuracy .....	+/-6% of Amplitude Swing +/-20mV (50Ω) +/-6% of Amplitude Swing +/-40mV (open)
Offset Accuracy .....	+/-6% of Amplitude Swing +/-20mV (50Ω) +/-6% of Amplitude Swing +/-40mV (open)
Undershoot, OverShoot.....	<15% of Swing
Rise, Fall times .....	200ps max (1V swing into 50Ω)
(20% / 80%) .....	250ps max (2V swing into 50Ω)

Output vs.  $\overline{\text{Output}}$  Skew .....30ps max

**Sync Output**

**Period**

Range .....	1.882ns – 1.97ms (531.25MHz - 510HZ)
Programming Resolution .....	25 bits resolution (0.015ps)
Accuracy.....	± 50 ppm (0.005%) of sync period
Divisor range.....	÷1 to ÷255 (step 1) ÷256 to ÷4080 (step 16)

**Modes**

Jitter Sync .....	Generates a sync marker coinciding with value of Jitter Frequency control. (Max. Freq. = 5MHz - Min. Freq. 510Hz)
Bit clock.....	Follows Main Frequency with divisor values of ÷1 to ÷255 (step 1) and ÷256 to ÷4080 (step 16). Minimum divisor is ÷2 for Main Frequencies > 531.25 MHz.
Independent .....	Operates independent from Main Frequency channel operation. (Max Frequency = 531.25MHz (531.25MHz ÷1) Min Frequency = 510MHz (2.08MHz ÷4080)

<b>Selectable Termination</b> .....	50Ω to GND, 50Ω to -2V open circuit
<b>Amplitude</b>	
<b>Preset Levels</b> .....	ECL, PECL, TTL, 3.3V CMOS, 5V CMOS
<b>Swing</b> .....	0.25V min: +3.0V max. (50Ω) 0.5V min: +6.0V max. (open)
<b>Max. High Level</b> .....	+2.5V (50Ω), + 5.0V (open) +2.25V (50Ω) above 200MHz
<b>Min. Low Level</b> .....	-1.0V(50Ω), -2.0V(open)
<b>Resolution</b> .....	20mV (50Ω), 40mV (open)
<b>Swing Accuracy</b> .....	+/-8% of Amplitude Swing +/-20mV (50Ω) +/-8% of Amplitude Swing +/-40mV (open)
<b>Offset Accuracy</b> .....	+/-8% of Amplitude Swing +/-20mV (50Ω) +/-8% of Amplitude Swing +/-40mV (open)
<b>Undershoot, OverShoot</b> .....	<25% of Swing
<b>Rise, Fall times</b> .....	800ps max (1V swing into 50Ω)
<b>(20% / 80%)</b> .....	1.0ns max (2V swing into 50Ω)

### Environmental Conditions

<b>Input Line Voltage</b> .....	100-130; 200-240 VAC 900VA MAX 50/60 Hz
<b>Operating Temperature</b> .....	0-40°C
<b>Storage Temperature Limits</b> .....	-20°C to 70°C
<b>Standards</b> .....	Complies with EN6010-01

### Computer Hardware

<b>Computer</b> .....	486 PC/AT compatible DOS 6.2
<b>Display, Internal</b> .....	7" B& W
<b>External Display Out</b> .....	VGA 16 color
<b>Keyboard</b> .....	Standard 101- key
<b>Mouse</b> .....	Microsoft compatible
<b>Floppy Drive</b> .....	1.44 Mbytes, 3.5" DSHD
<b>Hard Drive</b> .....	2.0 GBytes
<b>GPIB</b> .....	IEEE 488.2

This page intentionally left blank.

## APPENDIX B - TROUBLESHOOTING HINTS

There are potential problems that can be encountered when using the DTS-550. The following is a list of some of the more common ones observed and their solutions:

- *I cannot see a phase difference between the SYNC OUT and OUTPUT connectors even when a large amount of delay has been specified.*

Make sure that the **Sync Setup** field is set to **Independent**. Any of the other settings will lock synchronization between the SYNC OUT and OUTPUT connectors and no phase delay will be seen.

- *As settings are adjusted, the output suddenly becomes distorted or hangs at a specific logic level.*

Toggle the **Synthesizer** and **All Output** controls inactive, then active. This will reset the internal waveform synthesizer as well as the output relays and will clear up most problems of this type. In certain instances when the output becomes unstable or incoherent, toggling the **Synthesizer** and **All outputs** fields will not clear up the problem. At this point it may be necessary to exit the application software and reboot the system. This should be done by first entering <CTRL-Q> on the keyboard, and then using a hard reset (power off, power on) to reboot rather than using the <CTRL-ALT-DEL> key sequence.

- *After running for some time, the DTS-550 will unexpectedly power-down.*

The high-density, high-speed logic present in the DTS box requires constant cooling to avoid damage. Air-flow sensors in the box will shut the unit down if a steady airflow is not present. Check to see that adequate ventilation space is present at the front and rear of the unit and that nothing is underneath the unit such as paper or books. This problem may also occur if an attempt is made to power the unit with the enclosure loosened or removed.

- *When entering values in the Delay(ns) or Phase(deg) fields, the system will not accept these values immediately or appears to hang-up.*

The internal calculations required to implement these functions take a relatively long time to complete, especially as compared to other functions on the DTS unit. Be patient, and use the smallest value possible when entering information into these two fields.

- *When using some of the controls, the phase of my data pattern shifts radically.*

There is no guarantee that phase will be maintained. There are many operations (RUN/STOP for example) that will cause the OUTPUT signal to regenerate at a different phase relative to SYNC OUT. Even repeated recall of a saved setup will not guarantee SYNC OUT and OUTPUT will consistently have the same relative phase.

- *My mouse works erratically or doesn't work at all.*

Try plugging the mouse into the serial port on the back of the DTS-550. It's possible that the system settings have been corrupted and the mouse port on the back is now the primary serial port. If this doesn't work, try using another PC/AT compatible mouse with the system.

This page intentionally left blank.

## **APPENDIX C - ACCESSORIES AND OPTIONS**

### **USING AN EXTERNAL KEYBOARD**

In addition to using the front panel keypad and mouse to enter commands and data, a standard PC keyboard can be attached to the DTS-550. An AT-compatible keyboard using a standard 5-pin DIN plug is required, and must be attached to the keyboard connector at the rear of the unit (see Fig. 2.2). In certain instances, such as when a new filename has to be entered, the use of a keyboard is mandatory.

In general, using the **ALT** or **CTRL** key with a letter key will either pull-down a menu from the upper menu bar or go to the field beginning with that letter. Certain combinations will also cause an action to be performed. The following indicates how the **ALT** and **CTRL** key combinations function while in the main window:

#### **ALT KEYS:**

- <ALT-C> - pull-down **Configure** menu
- <ALT-D> - move between **DN** and **All Outputs Disabled**
- <ALT-E> - pull-down **Edit** menu
- <ALT-F> - pull-down **File** menu
- <ALT-H> - pull-down **Help** menu
- <ALT-I> - move to **I/O Levels**
- <ALT-J> - move to **Jitter**
- <ALT-P> - move between **Period** and **Pattern**
- <ALT-U> - move to **Step Up**
- <ALT-V> - pull-down **View** menu
- <ALT-W> - change **Step** to **Width Step**

#### **CTRL KEYS:**

- <CTRL-A> - toggle **All Outputs**
- <CTRL-C> - move to **Channel**
- <CTRL-D> - move to **Delay**
- <CTRL-L> - feature not implemented
- <CTRL-N> - **Step Down**
- <CTRL-P> - move to **Period**
- <CTRL-Q> - **Quit**
- <CTRL-R> - toggle **Synthesizer**
- <CTRL-S> - move to **Step**
- <CTRL-U> - **Step Up**
- <CTRL-W> - move to **Width**
- <CTRL-Z> - feature not implemented

Function keys on the keyboard perform the same actions as the softkeys on the front panel keypad. For example, use **F7** to bring up the I/O Levels window. In addition, **F1** is used to bring up the Help menu.

When working on the main screen or in any other window:

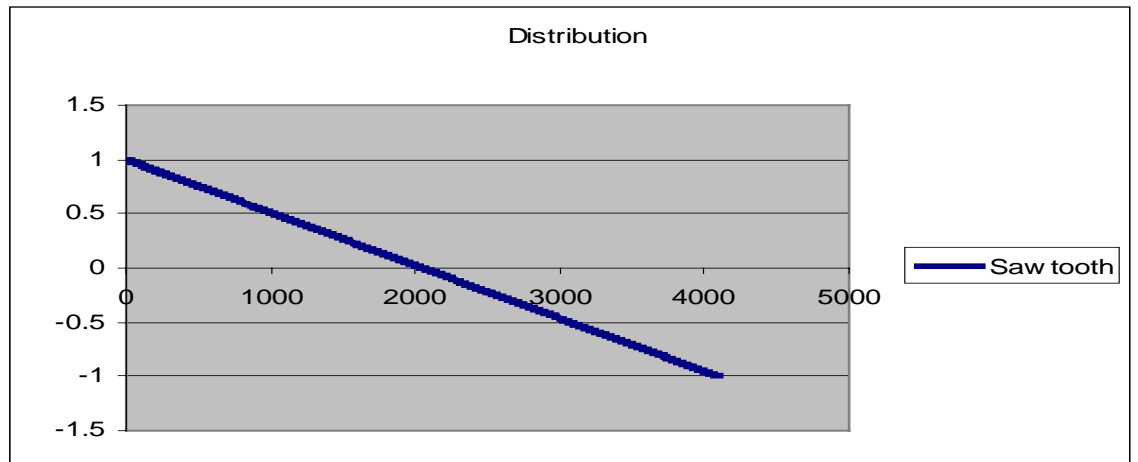
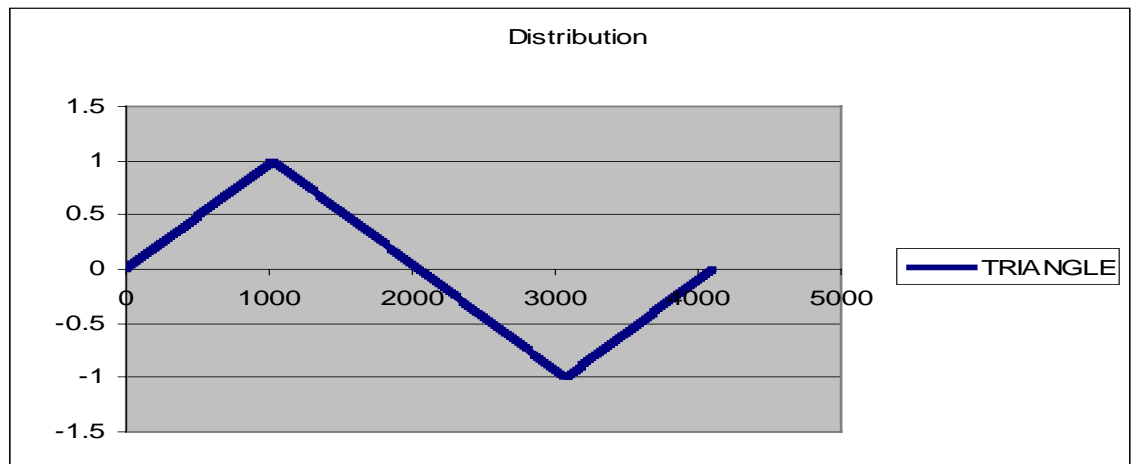
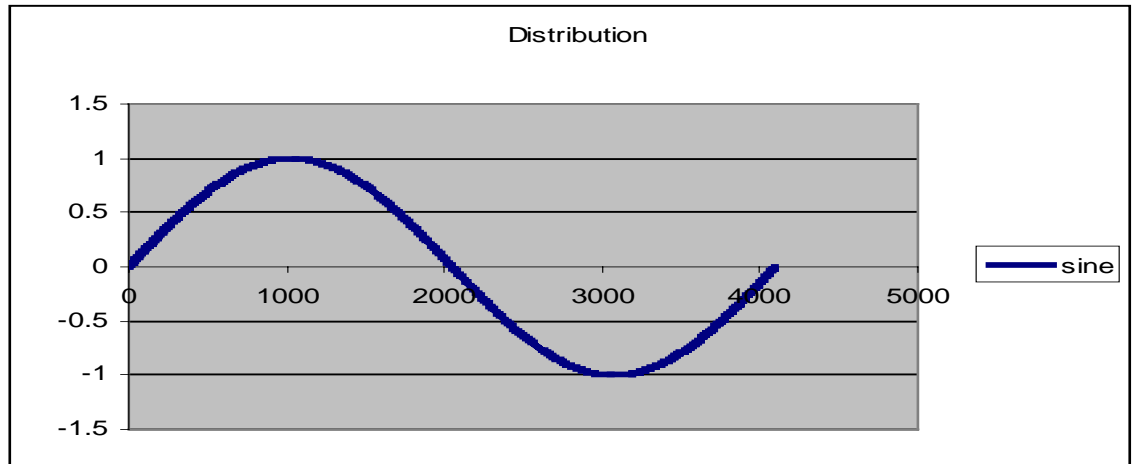
- <**TAB**> to move from control to control, highlighting the control to be changed.
- Press the <**space bar**> to display options.
- Use the arrow keys to move to the desired option, then press <**ENTER**> to select that option.

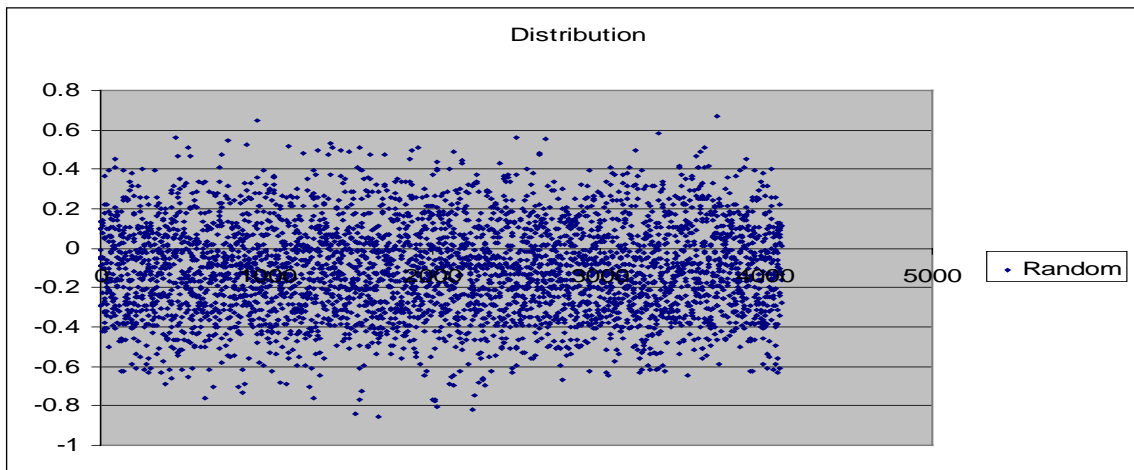
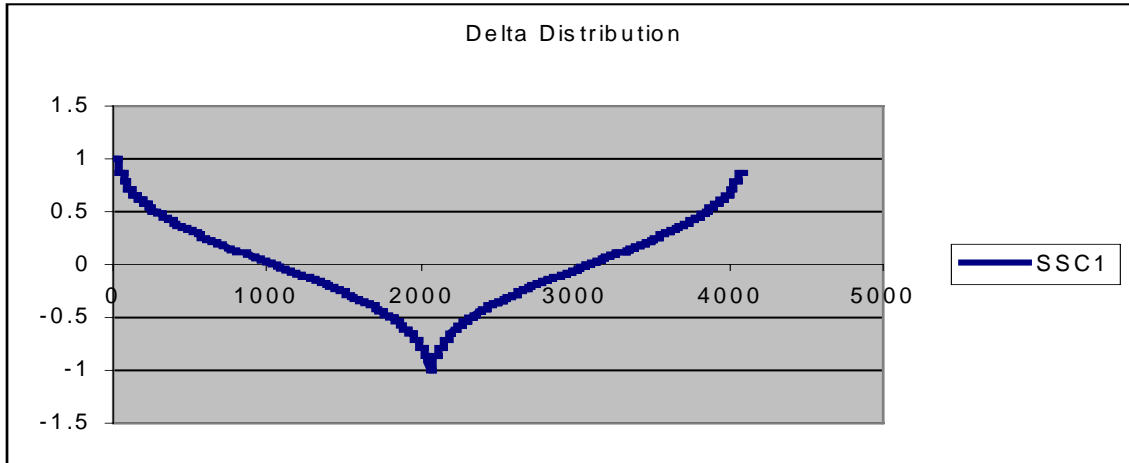


## **USING AN EXTERNAL MONITOR**

In addition to the monochrome display that is incorporated into the DTS-550, an external color or monochrome VGA display (640 x 480 resolution) can be attached. The monitor cable is attached to the VGA port (15-pin female D-sub connector) at the rear of the unit. See Figure 2.2 for the location of this connector.

## Appendix D – Standard Jitter Distributions





## **Appendix E - Maintenance & Service**

### **Calibration**

Periodic calibration is recommended for the DTS-550A/550. Please contact WAVECREST for calibration information 1-800-733-7128

### **Spare Parts:**

<b>Part No.</b>	<b>Description</b>
110413	Littlefuse® 3AG, 10A, 250V Fuse
110554	Keyboard
110553	Mouse
030100-36	36" Signal Cables
360108-01	SMA Adapters
400172-00	110V Power Cord, North America
400172-01	220V Power Cord, United Kingdom

### **Service & Repair:**

The DTS-550A/550 is a sophisticated test instrument that should only be repaired by a qualified service technician. Please contact WAVECREST for service & repair information. 1-800-733-7128

This page intentionally left blank.

**WAVECREST Corporation**

World Headquarters  
7275 Bush Lake Road  
Edina, MN 55439  
(612) 831-0030  
FAX: (612) 831-4474  
Toll Free: 1-800-733-7128  
[www.wavecrestcorp.com](http://www.wavecrestcorp.com)

**WAVECREST Corporation**

West Coast Office:  
1735 Technology Drive, Suite 400  
San Jose, CA 95110  
(408) 436-9000  
FAX: (408) 436-9001  
1-800-821-2272