

# WEIGH-TRONIX



## WPI-135 SimPoser<sup>®</sup> Software User's Manual



# Table of Contents

Table of Contents .....	3
Introduction .....	5
About This Manual .....	5
WPI-135 SimPoser™ and the WPI-135 Indicator .....	5
WPI-135 SimPoser™ Software .....	6
Minimum and Recommended Computer Requirements .....	6
WPI-135 SimPoser™ Installation .....	6
Starting WPI-135 SimPoser .....	7
Commands Overview .....	7
Toolbar Buttons Overview .....	8
WPI-135 SimPoser™ Operation .....	9
Toolbar Commands .....	9
File .....	9
Toolbar .....	9
Editors .....	10
Download .....	10
Simulate .....	10
Help .....	11
Toolbar Buttons .....	12
Configure Button .....	12
Program Button .....	27
Format Button .....	32
SetPoint Button .....	36
Simulate Button .....	45
Close Button .....	45
Sample Application Program Summaries .....	46
Appendix 1: Display Samples .....	48
Appendix 2: WT-BASIC Interpreter Command Set .....	51
Appendix 3: Subroutine Examples .....	92
Appendix 4: Error Messages .....	96
Appendix 5: ASCII Chart .....	98
Appendix 6: Alphabetic Listing of WT-BASIC Commands .....	99



# Introduction

## About This Manual

This manual covers the information you need to install and operate the WPI-135 SimPoser™ software package.

Major sections of this manual are headed by titles in a black bar like *Introduction* above. Subheadings appear in the left column. Instructions and text appear on the right side of the page. You will occasionally see notes, tips, and special instructions in the left column. This information will usually pertain to text in the opposite column.

## WPI-135 SimPoser™

WPI-135 SimPoser™ software works exclusively with the WPI-135 indicator. The word SimPoser™ comes from two root words—*Simulator* and *Composer*. These two words describe the strengths of this program. Built into the software is a computer simulation of the WPI-135. WPI-135 SimPoser lets you compose application programs and configuration setups for the WPI-135. You then test an application on the simulator. This makes quick fixes easy and assures identical function when you download the program to a WPI-135.

Application programming is done in the WT-BASIC computer language. These application programs and the configuration are saved in a computer file and can be recalled simply by opening that file. Because of this, you can design, buy or trade application programs.

Downloading a program to the WPI-135 is as easy as clicking a mouse button. The information you send to the WPI-135 is instantly active and the WPI-135 is ready to perform the task you have set it up to do. The program becomes part of the WPI-135's permanent memory and cannot be lost due to power failure. If you download a new program to the WPI-135, the old program is replaced with the new program. There are no chips to change and no long turnaround times. Each program can take the place of expensive software/hardware specials. Turnaround times can be measured in hours or days instead of weeks or months.

## Minimum and Recommended Computer Requirements

WPI-135 SimPoser is PC based and requires a certain level of computer power to function. With the minimum setup listed below, the system will work but slowly. The recommended configuration will run the program very well.

### Minimum Configuration

- IBM® compatible AT®/PC with an Intel Pentium microprocessor
- 16 megabytes of RAM
- 120 megabyte hard disk drive (program takes up a minimum of 5 meg.)
- 3.5" 1.44 megabyte floppy disk drive
- VGA color monitor
- Mouse
- Microsoft Windows® 95

### Recommended Configuration

- IBM® compatible AT®/PC notebook computer with an Intel Pentium II microprocessor
- 64 megabytes of RAM
- 120 megabyte hard disk drive (program takes up a minimum of 5 meg.)
- 3.5" 1.44 megabyte floppy disk drive
- VGA color monitor
- Internet access
- Mouse
- Microsoft Windows® Win95, Win98, or WinNT

This section of the manual is divided into the following sections:

- SimPoser Installation
- Starting WPI-135 SimPoser
- Commands Overview
- SimPoser Operation

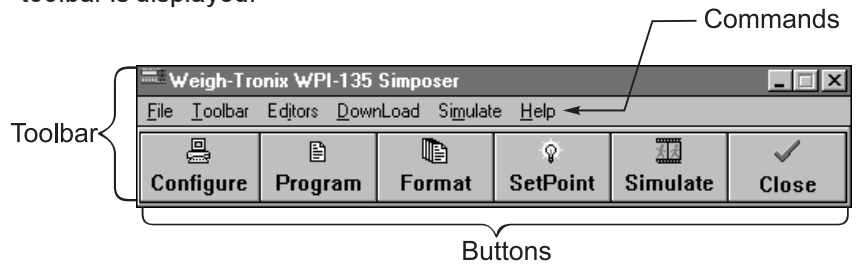
## WPI-135 SimPoser Installation

Place the WPI-135 SimPoser diskette in the disk drive and in Windows 3.1 click on File - Run - a:\setup, or in Windows 95 click on Start - Run - a:\setup. Follow all the on-screen prompts as the software is installed.

## Starting WPI-135 SimPoser

*You can move the toolbar anywhere on your screen by clicking and dragging the title bar.*

1. Once WPI-135 SimPoser is installed, double click the icon with the left mouse button. A license message appears and upon accepting, this toolbar is displayed:



**Figure 1**  
WPI-135 SimPoser's toolbar

## Commands Overview

The toolbar consists of six commands and six buttons. Some of these commands are common to all Windows programs and cause a drop down menu to appear. Some commands access the special features of the WPI-135 SimPoser program. A quick overview for each command is given below. Specific instructions appear in the *Operations* section of this manual. For help with Windows operations see your Windows documentation.

### File

This drop down menu lets you create, open, save and print files and exit the program. This menu also contains a list of the last nine files you have opened. This allows you to click on a file name and open it quickly.

### Toolbar

This allows you to select a small version of the toolbar.

### Editors

This is a navigation aid for accessing the different editing windows.

### Download

Click on this command, then **COM 1** or **COM 2** to download the active application program to the WPI-135.

### Simulate

Click on this command, then *Start* to bring up the WPI-135 simulator on the computer screen. All the parameters and instructions you have created and saved will be active and running on the display. This allows you to test your programs before downloading to the WPI-135.

### Help

Click on this command or press F1 to find indexed help documentation.

## Toolbar Buttons Overview

### Configure button

The toolbar has six buttons. You select the function you want by clicking on the appropriate button with the mouse cursor. Below are brief descriptions of each button's function. Complete instructions are in the next section, *WPI-135 SimPoser Operation*.

When you click on this button with your mouse, a tabbed notebook appears containing the customizable features you can set to suit your needs. Below is a list of items in this dialog box:

- Parameters
- Motion/AZT
- Filters
- ROC
- Time Out
- Counting
- Analog Output
- Serial Ports
- Network
- Misc
- Bargraph
- Units
- Key Enable
- Display Values
- Display Modes

### Program button

The Program button opens a WT-BASIC text editing window. Use this window to create application programs using the WT-BASIC computer language. In these programs you can configure the system to run a specialized counting program, setup a truck in/out system, design special graphics for use on the display, and much more.

### Format button

Click on this button to see a screen for setting up print formats. Design your custom format, choose the format # (from 1 to 32) and save it by clicking on the Save button. Any or all of these formats can be recalled in the program you create in the Program window. For example, you might use this feature for spreadsheet reports for managers or ISO documentation.

### Setpoint

Click on this button to bring up a dialog box for configuring setpoints.

### Simulate

Click on this button to bring up the WPI-135 simulator on the computer. This does the same thing as the Simulate command described earlier.

### Close

Click on this button to exit the WPI-135 SimPoser program.

The WPI-135 can be sealed for legal-for-trade use and the software protected from change by a switch located under the nylon plug on the rear panel.



# WPI-135 SimPoser Operation

## Toolbar Commands

### File

*This manual assumes that you know the basic Windows™ procedures. If not, see your Windows™ documentation.*

#### Helpful Hints:

1. SAVE often!
2. Do not have any other Windows™ programs running.
3. SAVE often!

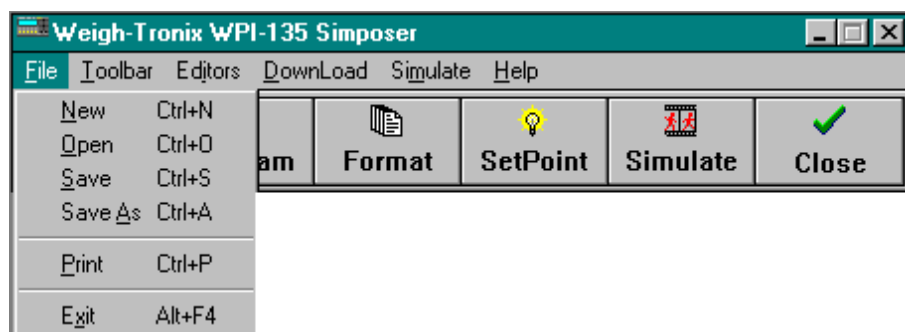
### Toolbar

*If the WPI-135 SimPoser software freezes up or crashes while you are working on an application program, you may be able to recover it even if you have not saved it using the SAVE command. If you have performed a download or simulation, the WPI-135 SimPoser program creates a temporary copy of the application in  
C:\wt\wpi135\SIM\tempcfg.cfg.  
To recover the program, open this file in WPI-135 SimPoser and do a FILE, SAVE AS,{filename} and use the file name you want for your application program.*

Following are specific instructions for each of the commands on the WPI-135 SimPoser toolbar.

1. Click on **File**. . .

The drop down menu shown in Figure 2 appears. With this menu you can call for a new file, open an existing file, save a file you are working on, save a file under a new name, print, or exit the WPI-135 SimPoser program. There is also a file history list which appears after you've opened a file. This list will hold up to nine of the most recently opened file names. You can click on the name of the file to open it.



**Figure 2**  
File menu

2. Click on the **Toolbar** command. . .

The following menu is shown.



Click on **Open Small Toolbar** to replace the large toolbar with the smaller one shown below. . .

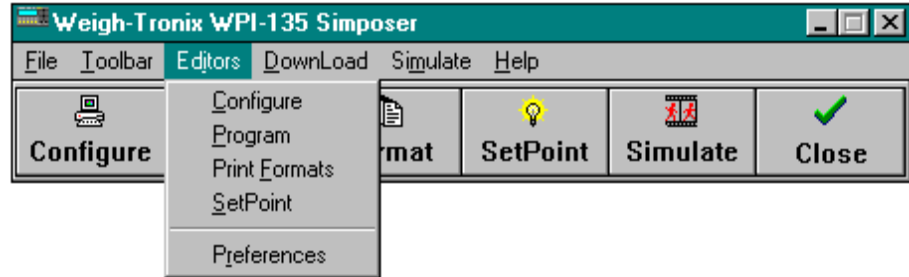


Notice that the small toolbar does not have the command line. You can return to the large toolbar at any time by clicking on the button on the far right side. You cannot close the WPI-135 SimPoser program from this toolbar. You must first return to the larger version. The other buttons on the small toolbar do the same things as their larger counterparts.

## Editors

- Click on the **Editors** command. . .

The following is displayed:



Click on the editing window you want. This is a duplication of the buttons and is handy for navigation of open windows.

The **Preferences** menu item allows you to configure the SimPoser program so that a **Tools** command appears on all the SimPoser toolbars. For more information about using this function see a Windows or DOS manual.

## Download

COM1 or COM 2 under Download refer to the serial output from your computer, NOT the serial ports on your WPI-135

*HINT: If you have an application with continuous output, you should disable it before you download. To do this, power down the indicator, hold in the **CLEAR** key and power the unit back up, then release the **CLEAR** key.*

- Click on the **Download** command. . .

The following menu is displayed:



When you have created a custom program, use this command to choose which Com port to use to download the program to your WPI-135. F11 and F12 keys can be used as hot keys for these functions.

## Simulate

- Click on the **Simulate** command. . .

A simulation of the WPI-135 appears on screen. It will behave the same way as a real WPI-135 loaded with the program you have active in WPI-135 SimPoser. Use this to test your program before downloading to the real WPI-135.

*If you are going to print from the simulator mode you need to add this line to your autoexec.bat file on your computer and reboot before trying to print:*

**SET WTPORT 2=1**

*What this means is that WTPORT2 is the simulated WPI-135 serial port number 2 and =1 is the communication port from your computer.*

**Never set both WTPORTS to the same computer COM port number.**

The following computer key strokes take the place of pressing front panel keys when using the simulation:

COMPUTER KEY STROKE	EVENT QUEUED
ALT-S	SELECT_KEY
ALT-Z	ZERO_KEY
ALT-T	TARE_KEY
ALT-C	CLEAR_KEY
ALT-U	UNITS_KEY
ALT-P	PRINT_KEY
ALT-X	EXIT
ENTER	ENTER_KEY
ESCAPE	ESC_KEY
1..9	NUMERIC_KEY
A..Z	ENTRY_KEY
F1..F10	F1..F10_KEY
.	DECIMAL POINT

F6-F10 are accessible through a remote keyboard or the simulator.

Move the mouse to change weight on the simulator.

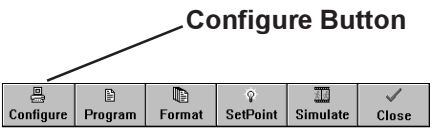
- Move the mouse to the right to increase weight value
- Move the mouse to the left to decrease weight value
- To add more weight with shorter mouse movements, click and hold down the left mouse button while moving the mouse to the right
- To add small increments of weight, hold down the right mouse button, while moving the mouse to the right
- To exit the simulation, press both mouse buttons at the same time or press **ALT + X**

## **Help**

Click on the Help command to bring up an indexed Adobe PDF (Portable Document Format) help manual on your computer screen.

This is the last item on the command line of the WPI-135 SimPoser toolbar. The next section describes the toolbar buttons.

## Toolbar Buttons

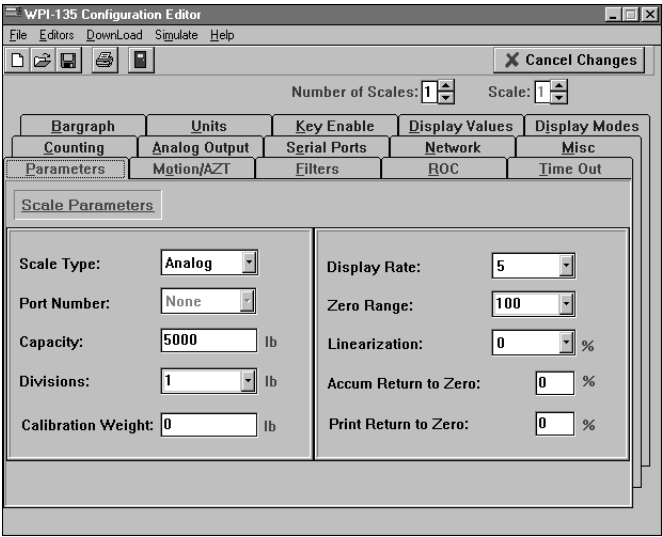


### TIP

*Combo Box = A Windows feature which allows you to type in a value or select one from a drop down list.*

*Text Box = A box into which you type a value or word.*

Click on the first toolbar button, **Configure**. The dialog box in Figure 3 appears.



**Figure 3**  
Configure dialog box

At the top of the screen are listed the number of scales connected and the active scale number. Use these boxes to set the number of scales you have and choose which scale you are configuring.

This dialog box contains many tabs representing different areas of scale function. Click on a tab to bring that function into view. The first tab is **Parameters**.

### Parameters tab

Following is a brief description of each item you see in the Parameters tab in Figure 3.

<i>Scale Type</i>	Select analog or Quartzzell® weight sensor.
<i>Port No.</i>	For Quartzzell® only. Select the serial port you wish to connect the Quartzzell to.
<i>Capacity</i>	Set the capacity for the chosen scale.
<i>Divisions</i>	Set the division size for the chosen scale platform. Values must be a multiple or submultiple of 1, 2, or 5. If you type an incorrect value, the program will automatically select the closest correct value.
<i>Calibration Weight</i>	The amount of test weight used to calibrate the scale. We recommend entering the test weight most commonly used to calibrate this scale capacity by your organization. (Minimum of 25% of capacity.)

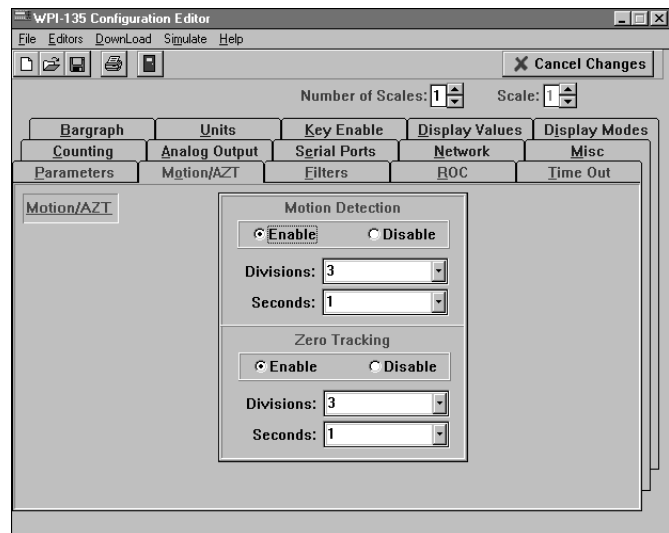
By default, when the **PRINT** key is pressed, a print operation and an accumulation take place. If you do not want the accumulation to occur, a WT-BASIC program assigning only the DOPRINT command to the **PRINT** key needs to be downloaded to the WPI-135. A WT-BASIC program can also define an ACCUM. soft key and assign accumulation to that key only.

The default print format is 0 and is transmitted out of port #1. (Format 0 = Gross, Tare, Net)

<b>Display Rate</b>	Set the display update rate (the number of times/second the display is updated.) Choose values between .1 (slowest, once every 10 seconds) and 5 (fastest) updates per second.
<b>Zero Range</b>	Select the percentage of scale capacity you can zero.
<b>Linearization</b>	Choose a number from -10 to +10 to pull the center point of span back to a linear value.
<b>Accum Return to Zero %</b>	To accumulate weight, the weight must be above this percentage of scale capacity and stable. Before you can perform another accumulation operation the weight must return to zero.
<b>Print Return to Zero %</b>	To print, the weight must be above this percentage of scale capacity and stable. Before you can perform another print operation the weight must return to zero.

#### Motion/AZT tab

**Motion/AZT** is the next tab. The dialog box is shown in Figure 4.



**Figure 4**  
Motion Detection and AZT dialog box

In this dialog box you enable or disable motion detection and Automatic Zero Tracking or AZT.

If you enable motion detection you can set the motion detection window size in divisions and the time window in seconds. The default for motion detection is three divisions and one second.

For AZT the division size you pick defines a range above and below zero. When scale weight is inside this range for the number of seconds you picked, 1/2 of the weight will be zeroed. The indicator will repeat removing 1/2 the weight every X seconds. X being the number of seconds you have picked. This will be repeated as long as the condition exists or until the indicator display reaches zero. The default is three divisions and one second.

## Filters tab

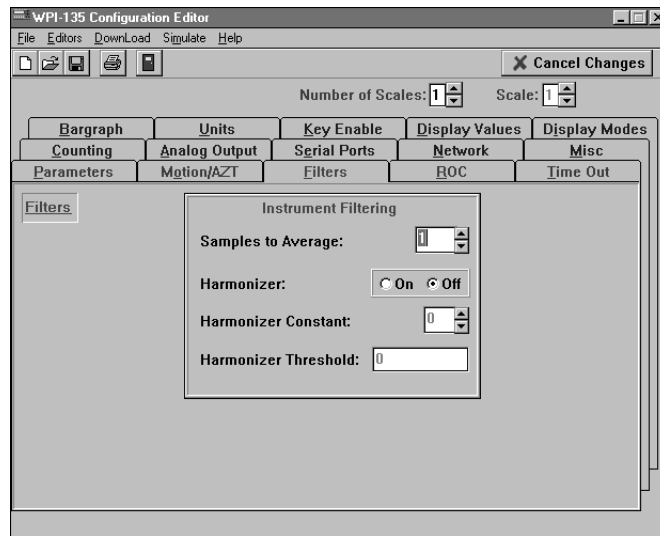
*Harmonizer threshold is based on actual weight in calibration units, not division size.*

*The Harmonizer Constant choices are 0-10 in the WPI-135 SimPoser program but it may be best to make the selection in the "real world" through the front panel.*

*Pounds is the default calibration unit.*



**Filters** is the next tab. This dialog box is shown in Figure 5.



**Figure 5**  
Filtering dialog box

Use this to set up the Harmonizer™ filtering. A full explanation of the Harmonizer™ is given below.

Samples to Average is the number of conversions you want to average. For example, with an analog base, if you pick 30, the unit will average the weight values from the last 30 conversions or ½ second and uses that value for displayed data.

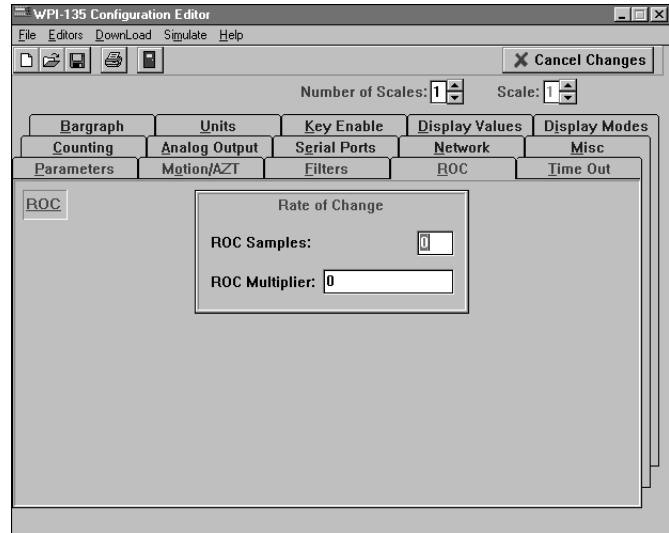
On an analog base, the Samples to Average should be set between 12 and 60. With a QDT base the value should be set to 1.

The next choice you have is for turning the Harmonizer filtering on or off. If you turn the Harmonizer filtering on you need to set the Harmonizer Constant. Typical values are between 1 and 8. Set the number low for small vibration problems and higher for more dampening effect.

The purpose of the Harmonizer Threshold is so the indicator will respond quickly to large weight changes. Harmonizer Threshold is the amount of weight change, in calibration units, beyond which the Harmonizer will be temporarily disabled. For example, if you set this to 10 lbs, a change in weight greater than 10 lbs between samples will disable the Harmonizer until the weight change during the sample time drops below 10 pounds.

## ROC tab

**ROC** is the next tab. This is the Rate of Change dialog box and it is shown in Figure 6.



**Figure 6**  
Rate of Change dialog box

*ROC is the rate of material flowing on or off a scale. If the flow of material is constant, the value displayed is zero. If the flow increases, the value is positive. If the flow decreases, the value is negative.*

$$\frac{\text{Cal Unit}}{\text{Custom Unit weight in Calibration Units}} = \frac{1}{8} = 0.125$$

The Rate of Change dialog box allows the user to set up a WPI-135 Indicator to calculate Rate of Change for flow rate, or weight/time, applications.

### ROC Samples

ROC Samples is the number of samples over which the rate of change of weight is determined. The WPI-135 converts weight from A to D at 60 times per second. If ROC Samples is set to 60, the WPI-135 is determining the rate of weight change over one full second.

### ROC Multiplier

The ROC Multiplier allows you to enter a conversion factor to translate weight to some other unit of measure, such as gallons/hour or tons/minute, etc. or some other weight unit based upon the active unit of measure during a specific time. There is an example on the next page.

---

#### ROC Examples:

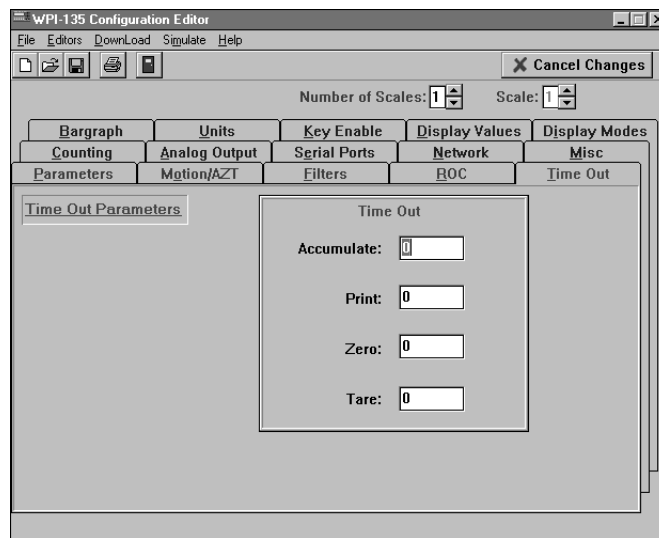
If pounds is your calibration unit, pick a sample value of 60 and a multiplier of 1. The display will show the rate of change in pounds/second.

For gallons of water/second set the sample value at 60 and the multiplier to 0.125. Water = 8 lbs/gallon (8 lbs is close enough for our example) so their are 0.125 gallons per pound. See formula to the left.

To get gallons/minute, do not change the sample size but rather multiply the 0.125 by 60 to get a value equal to gallons/minute (7.5). The display will then show you a rate of change in gallons per minute. (This is the flow over the last second not over a whole minute's time.)

---

**Time Out** is the next tab, shown in Figure 7.



**Figure 7**  
Time Out dialog box

Use this dialog box to set Accumulate Timeout, Print Timeout, Zero Timeout, Tare Timeout. This is the amount of time the WPI-135 will wait for motion to cease and perform the function after the corresponding key is pressed and/or the event is queued up.

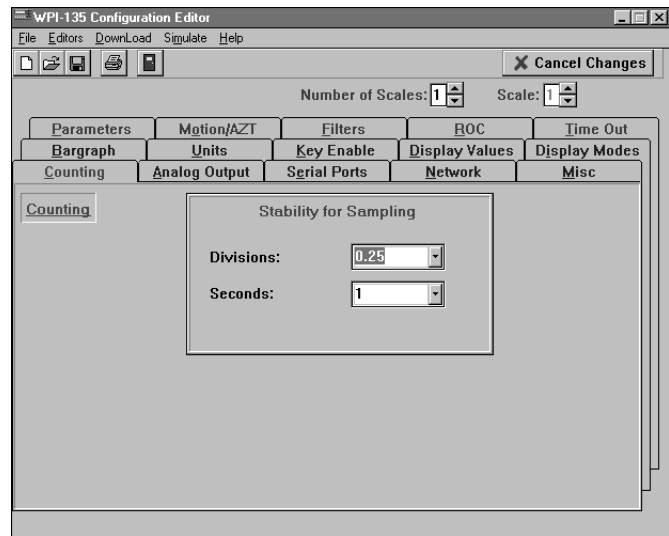
---

**Example:** If Zero Timeout is set to 3 seconds, when the **ZERO** key is pressed the unit will zero the scale if there is no motion. If there is motion and motion ceases within 3 seconds the unit will zero the scale. If motion doesn't cease the key press is aborted.

---

The same idea applies to the other three parameters in this dialog box.





**Figure 8**  
Counting dialog box

When an application program with counting operation is required, the Counting dialog box allows the user to select parameters relating to the stability of the scale during the parts sampling process.

#### Divisions

Select the number of scale divisions for stability. During the parts sampling process, this parameter determines how many divisions motion can occur on the scale while allowing the sampling process to occur. The smaller the number of divisions, the more stable the scale will need to be before the WPI-135 will go into sampling mode. If the stability window is exceeded, the sampling process cannot occur and no piece weight is established, therefore no counting can occur.

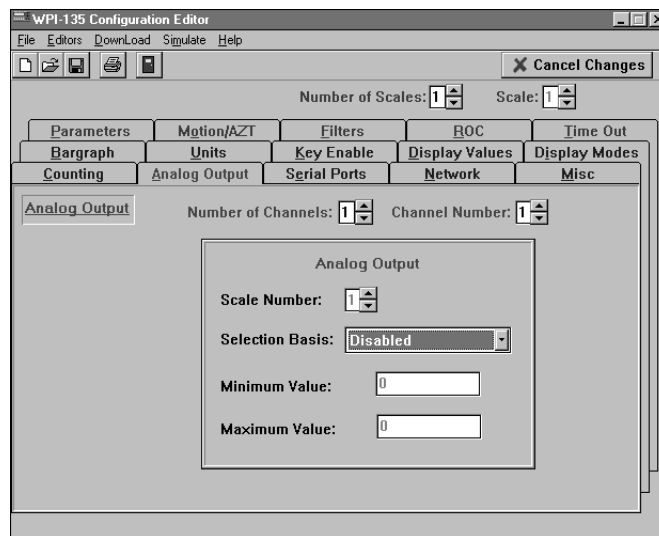
#### Seconds

Select the number of seconds the weight must be within the Divisions range before the WPI-135 will go into Sampling mode. In the setting above, the display must remain stable within 0.25 scale Divisions for one Second before sampling will occur and thus establish an accurate piece weight.

*Motion and AZT settings do not affect the sampling process, but they do affect the counting process.*

**Analog Output Basis list:**

*Disabled*  
*Gross Weight*  
*Net Weight*  
*Tare Weight*  
*Minimum Weight*  
*Maximum Weight*  
*Rate of Change*  
*Gross Weight Total*  
*Net Weight Total*  
*Count Total*  
*Transaction Total*  
*Count*  
*Variable*  
*Piece Weight.*

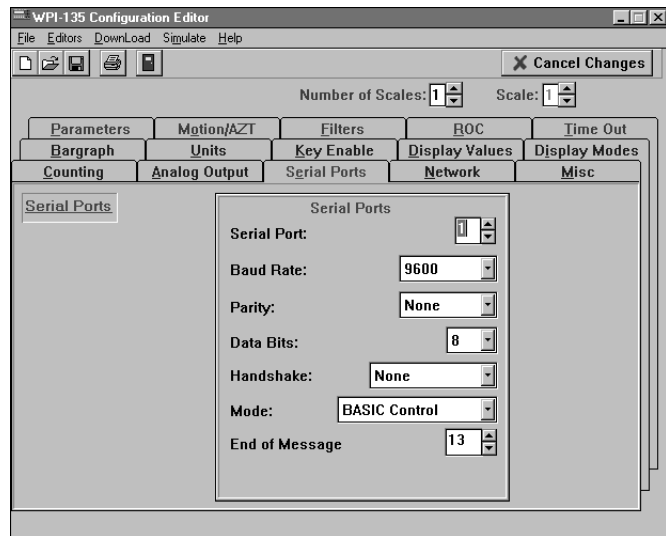


**Figure 9**  
Analog Output dialog box

In this dialog box you set the following:

1. The number of analog output channels you have
2. The channel you want to configure
3. The scale number you want the output based on
4. What the analog output will be based on (see the list at left)
5. The basis value which will cause the minimum output from the analog board.
6. The basis value which will cause the maximum output from the analog board.

**Serial Ports** is next. The serial ports dialog box is shown in Figure 10.



**Figure 10**  
Serial ports dialog box

*Consult your peripheral device manual for proper serial port parameter selections.*

The Serial Ports dialog box allows the user to set the parameters for the four WPI-135 serial ports. Each parameter is described below.

#### **Serial Port**

This selection switches between Serial Port 1,2,3 or 4 . Use the up/down arrow box to select the port or position the cursor inside the text box and type the number directly into the box.

#### **Baud Rate**

Use the combo box to select the Baud Rate from the list of selections.

Caution: If you use a baud rate above 19.2k, you need to use an error detection or correction protocol as well. Baud rate choices available are:

300	9600
1200	19200
2400	38400
4800	56700

#### **Parity**

Use the combo box to select the Parity setting from the list of selections. Choices available are shown in bold below:

	Stop Bits	Data Bits	Parity
<b>None</b>	1 or 2	7 or 8	None
<b>Odd</b>	1 or 2	7	Odd
<b>Even</b>	1 or 2	7	Even
<b>Set</b>	2	7	None
<b>Clear</b>	1	8	None

*CTS is a hardware handshake (ready/busy) which requires two extra wires in your cable.*

*Xon/Xoff is a software handshake requiring no additional hardware.*

*Software must support this protocol in all devices.*

*A keyboard is an input device that a WPI-135 is listening to or receiving data from. You may share a serial port with a printer that just listens or receives data from the WPI-135.*

*Ports 2 and 4 have a port level setting. Port 2 can be configured for RS-232/RS-485 or 20mA current loop. Port 4 can be configured for RS-232/RS-485 or infrared.*

### Data Bits

Use the combo box to select the Data Bits from the list of selections. Choices are 7 or 8.

### Handshake

Use the combo box to select the Handshake protocol from the list of selections.

Selections include:

None -	No Handshake protocols are selected.
CTS -	Clear to Send protocol is selected.
Xon/Xoff -	Xon/Xoff protocol selected
Both -	Both CTS and Xon/Xoff protocols selected.

### Mode

Use the combo box to select the mode from the list of selections. The following mode selections are available:

BASIC Control -	Control of the serial port is through the WT-BASIC program executing in the WPI-135. When BASIC Control is selected, the End of Message box appears. Select or enter the ASCII value for the end of message character to denote the end of the serial transmission. For example, setting the end of message character to 13 would indicate that the transmission would end on the reading of a "carriage return" (ASCII character 13).
Keyboard -	Control of the serial port is through an attached keyboard. As in Basic Control above, with this selection, an End of Message character must be entered.
Disabled -	The serial port is turned off.
Multi-Drop-	When an RS-485 network is used multi-drop mode automatically enables the RS-485 line drivers while transmitting. It tri-states the drivers when the system is done transmitting. No standard protocol is implemented. The programmer must write their own.

Computer Mode- For future use.

### End of Message (EOM)

Enter the decimal number that represents the ASCII character the WPI-135 expects as a signal for end of data transmitted to it from a communicating device such as a PLC or computer. When an EOM is received a COM1\_MESSAGE, COM2\_MESSAGE, COM3\_MESSAGE, or COM4\_MESSAGE event is queued in the WT-BASIC program. You must then write BASIC code for the COM1\_MESSAGE, COM2\_MESSAGE, COM3\_MESSAGE, or COM4\_MESSAGE event containing the GETCOM\$ BASIC command.

### WT Standard Network Input

Address	Value
0	reqZero
2	reqTare
4	reqPrint
6	reqAccum
8	setTare
10	setCurrentUnits (see CURUNIT)

*Req is motion inhibited.  
Set is not motion inhibited.*

### WT Standard Network Output

Address	Value
0	Gross
2	Net
4	Tare
6	Motion
8	Center of zero
10	Active value (see ActValue keyword)
12	Overload- Underload

*Outputs: Gross, Net, Tare  
return the current active  
numeric value.*

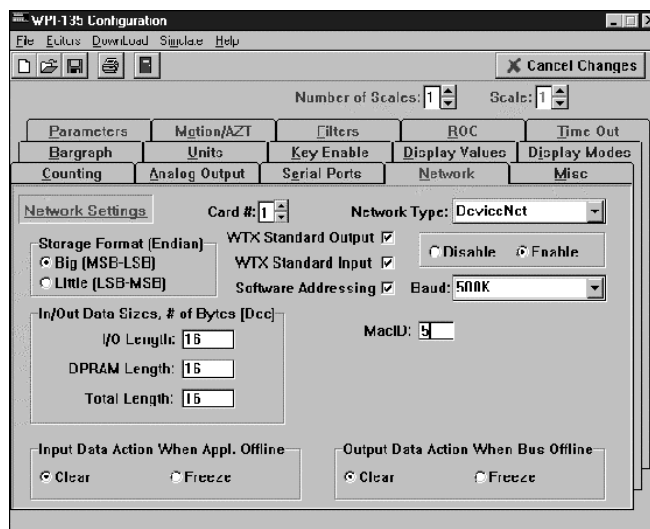
*Motion and Center of Zero  
return a True (1) or False (0).*

*ActValue returns 0-13, which  
represents the current active  
value on the display.*

*Overload-Underload returns  
0 = Okay, 1 = Under, 2 = Over*

**For more detailed information  
on network connections  
and programming please  
reference the WPI-135  
NETWORK INSTALLATION  
GUIDE P/N 29762-0015.**

**Network** is the next tab, shown in Figure 11.



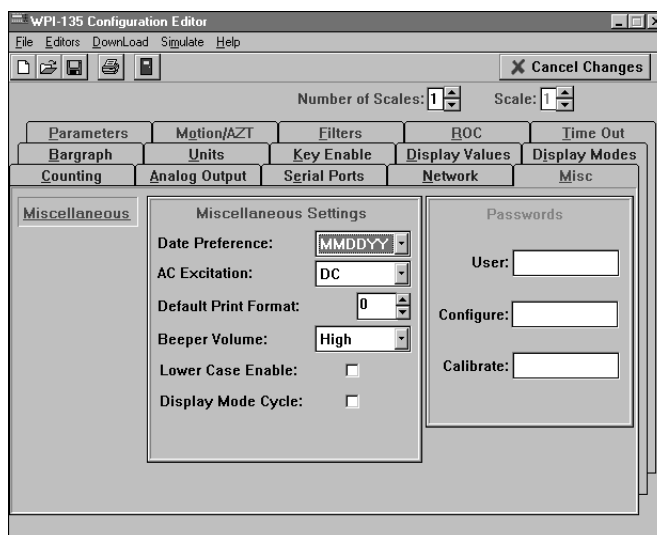
**Figure 11**  
**Network dialog box**

Use this tab to configure your network card. In this box you can set the following:

1. Card number to configure
2. Network type. Figure 11 shows DeviceNet™ as an example.
3. Enable or disable WTX (Weigh-Tronix) standard output (see table at left)
4. Enable or disable WTX (Weigh-Tronix) standard input (see table at left)
5. Enter the appropriate MacID address for the network module used.
6. Disable or enable the network card
7. Configuration for the network card (data size) memory (all three values should be identical). The numbers for input and output memory represent the default data type for a given network. Using the BASIC command MAP, will require you to increase the allocated memory. Example: 16 Bytes will be declared for DeviceNet.
8. Input Data Action when Appl. (Application) offline:
  - Clear - sets all memory to zero (0) if the network goes offline
  - Freeze - leaves the memory alone if the network goes offline
9. Output Data Action when Network Bus offline:
  - Clear - sets all memory to zero (0) if the network goes offline
  - Freeze - leaves the memory alone if the network goes offline
10. Storage Format-Big Endian or Little Endian. This defines the order of data storage (MSB-LSB or LSB-MSB).

Misc tab

**Misc** is the next tab, shown in Figure 12.



**Figure 12**  
Miscellaneous dialog box

The time and date may be sent in a variety of formats to the display, printer or computer. Format examples are: AM/PM, 24 hour, numerical reference, or spelled day and month.

Default format 0 outputs the following information:

**G** 12 LB  
**T** 4 LB  
**N** 8 LB

Display modes requiring BASIC text will show blank screen space if no WT-BASIC program exists to support screen text.

Record miscellaneous parameters for the configuration system in this dialog box. The first four selections use the combo box for entry and the last three use the Text box. The following items are included:

#### **Date Preference**

Allows you to set the WPI-135 System Clock in Month-Day-Year format, Day-Month-Year format or Year-Month-Day format.

#### **AC Excitation**

You can select the analog weight sensor excitation to be a DC level of 10 volts or one of three frequencies for AC excitation which is useful for reducing weight shifts due to extreme temperature changes.

#### **Default Print Format**

Allows you to select which one of the 32 print formats you want to designate as the format used when the Print key is pressed. The default is format 0 and it is only sent to port #1.

#### **Beeper Volume**

Allows you to turn the WPI-135 internal beeper off, or select from three volume levels.

#### **Lower Case Enable**

If you enable this option, letters in the soft key labels and all factory defined messages may be displayed in lowercase. If not enabled, all labels and messages are displayed in upper case.

#### **Display Mode Cycle**

Enable this mode for product demonstration of the display modes. If enabled, pressing the decimal key on the front panel causes the display to cycle through the display modes. Deselect this option to disable it and when using the unit as a weight indicator.

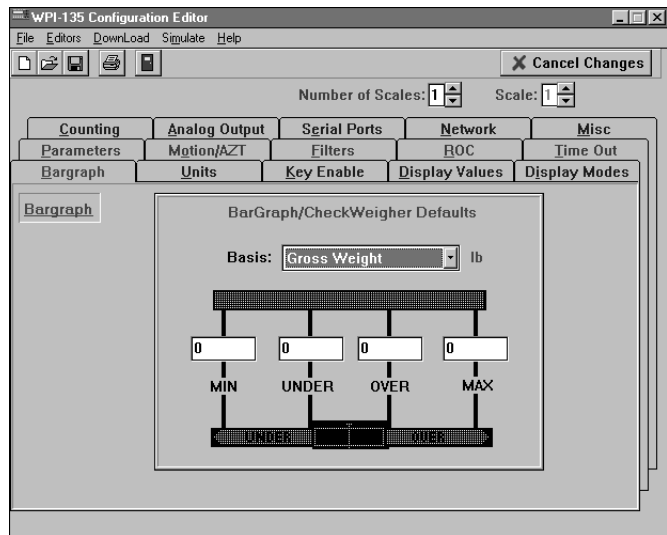
## Passwords

The next three boxes allow you to change the following passwords:

- User - (Default is 111.) Allows access to basic user parameters through the WPI-135 keyboard.
- Configure - (Default is 2045.) Allows access to configuration parameters.
- Calibrate - (Default is 30456.) Allows access to WPI-135 calibration routines.

## Bargraph tab

The next tab is **Bargraph**, shown in Figure 13.



**Figure 13**  
Bargraph/Checkweigher dialog box

Use this dialog box to enter parameters relating to bargraph or checkweigher functions, if the WPI-135 is configured to operate in these modes. To operate in bargraph or checkweigher mode, the proper display mode must be selected.

### Basis

Use the combo box to select one of thirteen choices for the Basis upon which the bargraph or checkweigher will be operating. It also is associated with the selected calibration unit selected in the Units dialog box.

### Min

Enter the Minimum value of the Basis selection for which the bargraph or checkweigher will begin registering movement. If the WPI-135 is set up with a bargraph, this value will determine when the graph on the display will begin moving. If the WPI-135 is set up with a checkweighing display, the "Under" portion of the checkweigher graph will begin receding when this value is exceeded. The Minimum value can be set to any value, including negative values.

### Under

Enter the Under value of the Basis selection. This value applies only to Checkweigher configurations and represents the Lower Acceptance Value

for the checkweigher application. At this point, the "Under" portion of the checkweigher graph will disappear and the needle in the "Accept" range will be at it's extreme left position.

### Over

Enter the Over value of the Basis selection. This value applies only to Checkweigher configurations and represents the Upper Acceptance Value for the checkweigher application. At this point, the needle in the "Accept" range will be at it's extreme right position and the "Over" area will not yet be visible.

### Max

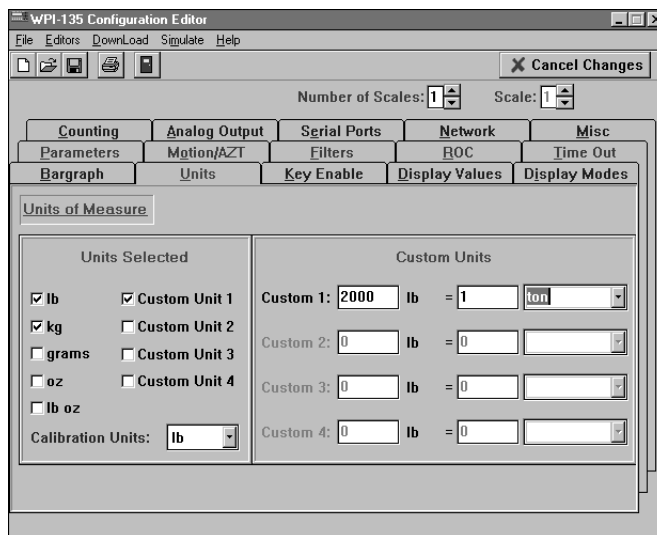
Enter the Maximum value of the Basis selection for which the bargraph or checkweigher will end registering movement. At this point, both the bargraph and checkweigher modes reach their maximum position and do not register further movement.

### Units tab

Setpoint and configuration parameters such as capacity, bargraph, checkweigher are based on the calibration unit, not on displayed unit of measure.

Figure 14 shows an example of setting 2000 lb = 1 ton for Custom Unit 1.

The next tab is **Units**. This is shown in Figure 14.



**Figure 14**  
Units dialog box

Following is a brief description of each of the unit of measure items you see in Figure 4.

*lb, kg, grams, oz,  
lb/oz, Custom  
Unit 1, 2, 3, 4*

Select the units of measure you want to use. Those you enable will be available to you as the **UNITS** key is pressed on the WI- 130. Conversion factors are preassigned by the factory for lb, kg, grams, oz, and lb/oz.

*Calibration Unit*

Select the unit of measure used in the calibration of your scale. Choose from lb, kg, grams, or oz.

*Custom Units*

If you select a custom unit, enter the number of calibration units equal to a number of custom units and choose a label for the custom unit or type in your own label.

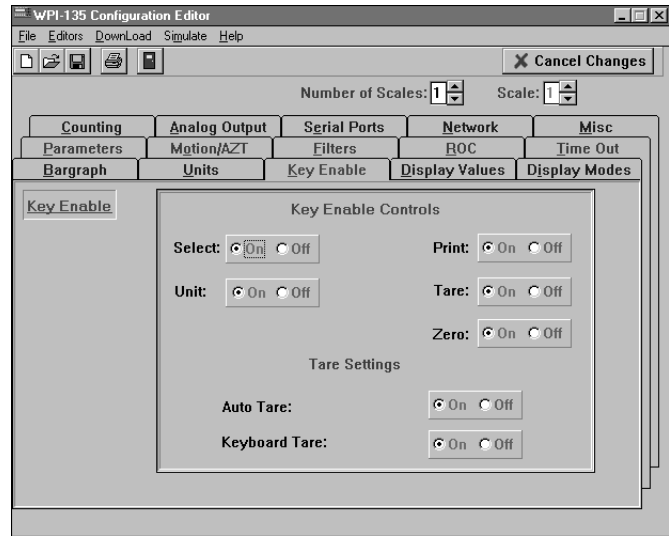


### Key Enable tab

*Perform an Auto Tare by placing a container on the scale and pressing the yellow **TARE** key.*

*The Keyboard Tare allows you to enter numbers, via the keypad, of known tare values. Key in values based on the calibration unit only.*

The next tab is **Key Enable**, shown in Figure 15. This dialog box lets you enable or disable the keys listed. You can also enable or disable the autotare function or the keyboard tare.



**Figure 15**  
Key Enable dialog box

### Display Values tab

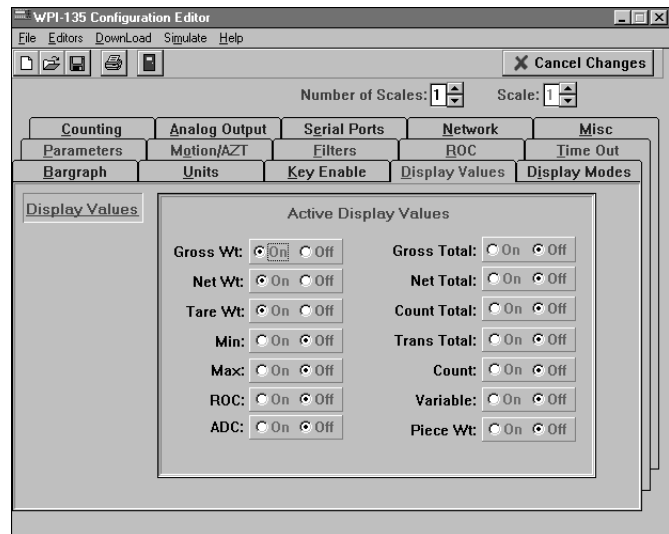
*ROC stands for Rate Of Change.*

*Min = Minimum captured weight*

*Max = Maximum or peak captured weight.*

*Variable = a value defined by WT-BASIC programming using the keyword or command (SHOWVAR)*

**Display Values** is the next tab, shown in Figure 16

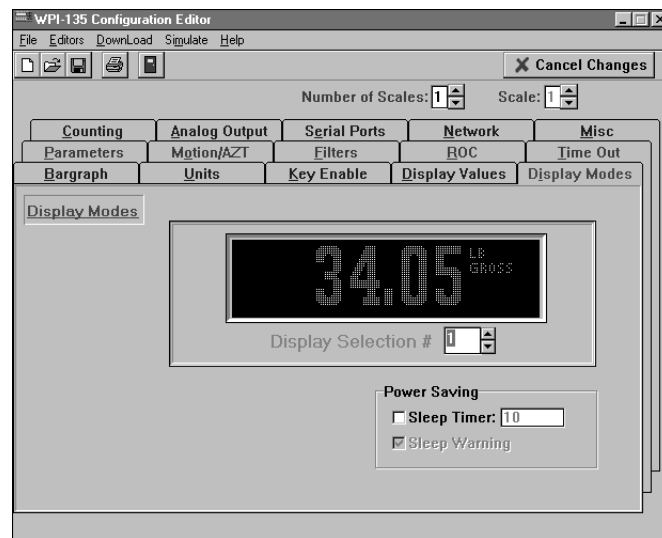


**Figure 16**  
Display Values dialog box

From this dialog box, enable the types of display values you wish to be active and available during normal weighing operations. The display values you choose will show up on your screen as you repeatedly push the **SELECT** key on the front panel of the WPI-135 during normal operation.

See Appendix 1 for samples of displays.

**Display Modes** is the next tab, shown in Figure 17.



**Figure 17**  
Display Modes dialog box

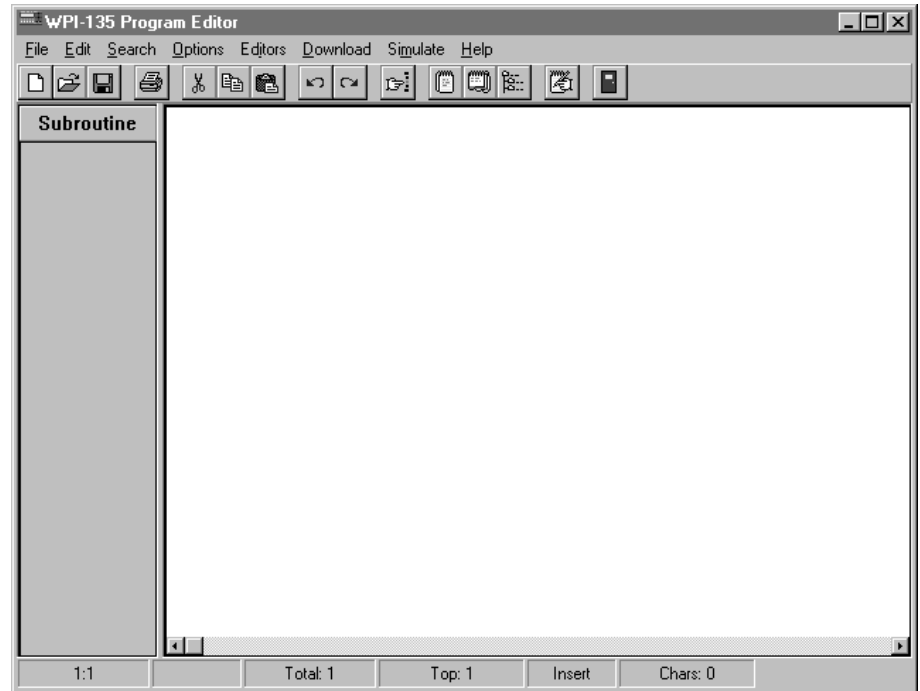
There are many display modes available. This dialog box lets you scroll through and select the style of display you want for your application. Displays vary in size of text, numbers, type of graphing, and soft key availability.

You can set the sleep timer and sleep warning in this dialog box. Enable the timer by clicking on the box, then type in the amount of idle time which will cause the unit to go into sleep mode. If you enable sleep mode you can enable or disable the sleep warning beeper. If enabled the beeper will sound several times before the indicator goes to sleep.

### Program Button



The next button on the WPI-135 SimPoser toolbar is **Program**. Figure 18 shows the screen which appears when you click on this button with your mouse.



**Figure 18**  
Program editor window

*If you forgot to save your file but have either downloaded to a WPI-135 indicator or have tested your file by using the simulator, a copy of your .cfg file exists in  
C:\wt\wpi135\sim\tempcfg.cfg*

You use the program editor window to enter the code for a WT-BASIC program. When you save the program you create in this window, all the setpoints, print formats, and configuration information you have selected are saved as a whole in a .cfg file. This file can be run in the simulator mode or downloaded to the WPI-135.

WT-BASIC programming is the key feature of the WPI-135's power and flexibility. Using the programming system in conjunction with the Setup configurations, print formats and the Setpoint configuration allow the WPI-135 to be adapted to a wide variety of user defined applications.

## Program Editor Window Commands

*The Program Editor window can be maximized by clicking the maximize button in the upper right corner of the window or be resized to suit your needs by clicking and dragging a corner of the window.*



The program editor window has several commands and buttons. The commands in this window are:

- File
- Edit
- Search
- Options
- Editors
- Download
- Simulate
- Help

### File

The file command operates the same way as the file command on the main toolbar but it also lets you access the Windows print setup dialog box and print the text currently in the Program Editor window.

### Edit

The edit command allows you to cut, copy, paste, and delete text in the editor window. You can also undo or redo actions and select all text in the window.

### Search

This command drops down a menu which helps you find and replace text. This can be very handy when the program is very long and you need to find a particular section for changes. Hot keys for these functions are F2 for Find, F3 for Find next, and F4 for Replace. When you access the Find function, the dialog box has a place to type in the text you want to find. You can cause the program to look for only exact matches to what you type in or words that contain the letters you type in.

This command also has a **Go to line** feature. This allows you to move to any line of the program simply by typing the line number in a popup dialog box.

The last feature in the search command is called **Program Errors**. This is used to retrieve a list of errors in your program. If you simulate a program and it contains an error or errors, WPI-135 SimPoser creates an error file. You can retrieve this file by clicking on **Program Errors** in this drop down menu or clicking the program errors button on the tool bar. See button at left. A dialog box pops up listing the error and the line it is on. This error file is automatically deleted when you load a new configuration file or test an updated version.

Use the **Set Marker** command to set place markers in the program. Use the **Go to Marker** command to find a previously set marker in the program. This is helpful when trying to find a routine you are currently creating.

## Options

This command lets you customize the editor window and its function. Each item is briefly described below.

Auto-indentation	If you enable this function with a checkmark, your lines of text will automatically indent the same as the previous line of text.
Font	Click on this option to bring up a Font popup box. Use this to choose what type font, font style and type size is used in the editor window.
Tabs	<p>This option lets you set three types of tabs and customize the tab size.</p> <p>Fixed Tabs - Pressing <b>Tab</b> keys moves the cursor to the next tab position. This inserts spaces not true tabs.</p> <p>Real Tabs - Pressing <b>Tab</b> keys moves the cursor to the next tab position. This inserts real tabs into the text, not spaces.</p> <p>Smart Tabs - Pressing <b>Tab</b> keys aligns the cursor on the current line to the position of the next closest text on the previous line. This inserts spaces not true tabs.</p> <p>Tab Size - Choose a tab size.</p>

## Editors

Click this command to access the other editing windows without returning to the main toolbar.

## Download

Click this command to download the program to your WPI-135. You are given a choice of which port to use. The F11 and F12 keys are hot keys for downloading to Com1 and Com2 respectively.

## Simulate

Choose this command to start the WPI-135 simulation using the program you are working on.

## Help

Choose this command to open help documentation on WPI-135 SimPoser operation.

## Program Editor Window Buttons

*New File Button*  
*Open File Button*  
*Save File Button*

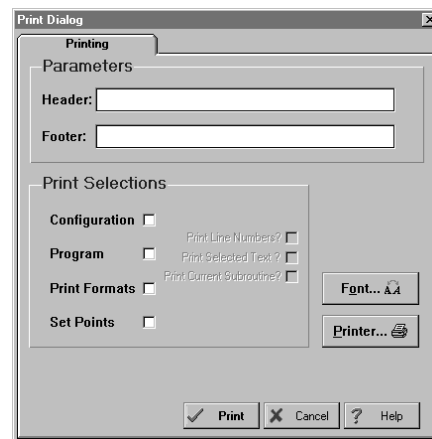


These three buttons are for opening a new file, opening an existing file or saving a file you are currently working on.

*Print Button*



This button is for printing the program. The following dialog box appears when you click on this button:



You can insert a header and footer into the printout. You can also choose what information to print: Configuration; Program; Print Formats; and Set Points.

*Cut Button*  
*Copy Button*  
*Paste Button*



These three buttons are for cutting, copying and pasting text.

*Undo Button*  
*Redo Button*



These two buttons are for undoing and redoing an action.

*Find Text Button*



This button brings up a dialog box for finding text.

*Toggle Event List Button*



This button is for toggling the event list on and off. Use the elevator button or the arrow keys to scroll through the entire list of events available to you.

*Teaching BASIC programming language is beyond the scope of this manual. There are many good reference books on the subject. All the WT-BASIC commands available to you in the WPI-135 SimPoser program are listed in **Appendix 2: The WT-BASIC Interpreter Command Set**. Use these commands to build your program.*

This is a list of predefined events and subroutines you can use in your program. Double click an event name to place it at the end of your program. If the event name already exists in your program, double clicking the event name will cause the event to be found and highlighted in the program. After the event name is placed in your program you need to fill in your particular commands.

### *Toggle Key Word List Button*



This button is for toggling the WT-Basic keyword list on and off. Below are some of the items in this list. The entire list is found in *Appendix 2* of this manual.

```
abs(NUM)
actvalue=NUM
actvalue(NUM)
and
asc(C$)
ask(PROMPT$,"f1","f5")
atn(NUM)
avgstart
avgstop
beep
bitmap(ADDR,"data1,data2,data3,data4")
```

When you double click a selection, it will appear in your program where your cursor is located. You will need to replace the syntax placeholders with your values or strings.

### *Toggle Sub Routine List*



This is a list containing currently used event names and self-created subroutine names that are currently in your program. The Update window on the editor screen will appear or disappear when this key is pressed. By double clicking on an event or subroutine name in this list, your cursor is relocated to the beginning of that subroutine.

### *Program Errors Button*



*If the line referenced in the error dialog box appears to be OK, click the line before and after it or check the Print Format for misspelled variable names.*

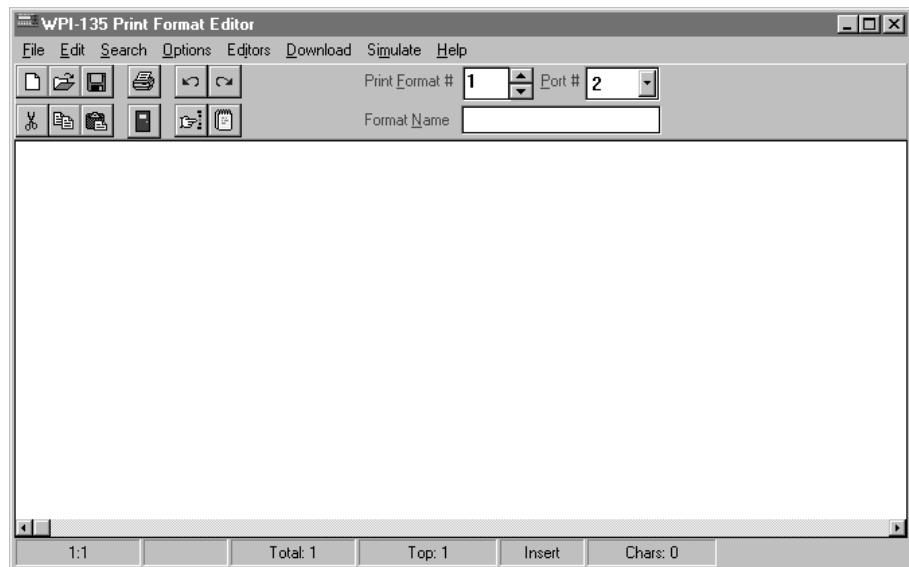
This button is for bringing up a program error dialog box. This box will only exist after an error is found in the program during simulation. When the simulation is canceled, this box will appear in the Program Editor window. If your program has multiple errors, you can move through them using the Previous Error and Next Error buttons. You can close this window by clicking the Close button. You can make it reappear by clicking the Program Error button again or clicking on Program Errors under the Search command at the top of the Program Editor window.

### Format Button



The next button on the toolbar is **Format**. Use this to control the way a particular printer connected to a WPI-135 will print a document or bar code label. The program is capable of 32 output formats.

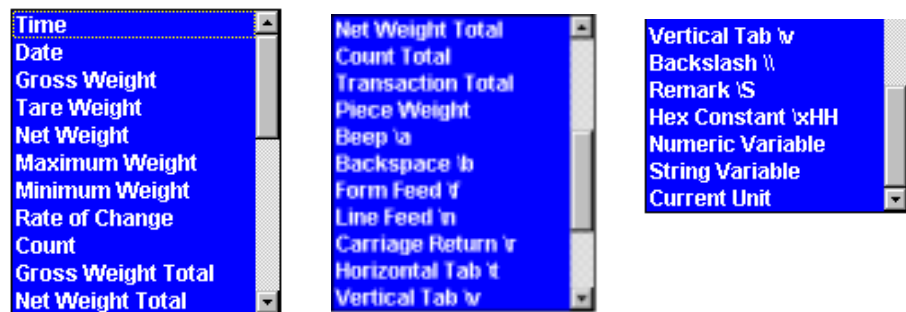
Click the format button and the format editing window shown in Figure 19 appears.



**Figure 19**  
Print format editing window

The commands along the top of the window are the same as those described in *Program Editor Window Commands*.

The buttons are also the same as those in the program editor window with one exception. The lower right button brings up a list of terms. The list is shown below.



Choose the number of the print format you want to create or work with. Choose which port you want it printed from and give the format a name by typing it in the Format Name box. You may use any number of print formats, from one to 32, with each configuration file. You do not have to use them in order: it is possible to use Format 1, Format 9 and Format 15, for example, or any other combination you desire.



In a WT-BASIC program, actual values that can change during the process are represented by "VARIABLE NAMES".

In WT-BASIC there are variables that are predefined. These are referred to as "system variables", such as "gross weight", "count", etc. These are available under TERMS. You may also create and use your own variables in WT-BASIC.

To get the current unit of measure label to print after a weight value, place the system variable {curunit\$} after the callout.

**\S** may be used in a print format to comment a section out or prevent a carriage return from being sent. Any character to the right of a \S will not be printed. This is a case sensitive command or term.

For printed tickets, you use the editor to lay out a format in logical order. The example in Figure 20 shows a typical truck scale ticket format, with item legends on the left side of the ticket and the actual values for the variables to be printed (in brackets) on the right side. As you can see in the adjacent sample ticket, the format looks very much like the actual ticket that is printed.

### Example

Format	Actual Ticket
{Cname\$}	HIGHLAND STONE
Operator: {opid\$}	Operator: JDB
Truck ID: {trID#,6.0}	TruckID : 69
Date : {date\$}	Date : 12-31-99
Time In : {trTIMEIN\$}	Time In : 09:33:23
Time Out: {trTIMEOUT\$}	Time Out: 12:59:59
Gross : {trGROSS#,6.0} {curunit\$}	Gross : 85280 lb
Tare : {trTARE#,6.0} {curunit\$}	Tare : 10500 lb
-----	-----
Net : {nET#,6.0} {curunit\$}	Net : 74780 lb
Signature:	Signature :
-----	-----
\r\r\r	

**Figure 20**  
Print format example

### Printing Titles and Legends


For fixed legends or titles, type the information in position in the edit area of the screen, exactly as you want to see it on the finished ticket. Use the space bar to move to the desired position and the **ENTER** key for additional lines. Lay out the ticket to match the design you desire. Once you have some of the information laid out you may move around the screen using the cursor arrow keys on your keyboard.

### Printing Actual Values

The actual values that will be printed on the ticket when run with the WPI-135 require a different method of placement. Values from the WPI-135 must be surrounded with brackets ( { } ) to tell the Print Format that this information will be coming from data stored, generated or collected by the WPI-135. Information such as scale weights, accumulated total, transaction counts, piece weights, part counts, and Variables must be enclosed in brackets in the Print Format. This is done automatically when using the TERMS list box shown on the previous page.

Depending on the type of printer you use, you may have to use the following commands to make it respond correctly:  
␣ "carriage return"  
or  
␣ "line feed"

### Selecting Actual Values

An alternative to typing in actual values is to use the terms list. Click the terms list button (  ) to make the list appear.

1. Place the cursor in the position on the screen where you want to place a new value.
2. Double click with the left mouse button on the value item you want to place in your format.
3. Repeat steps 1 and 2 until you have the terms you want in the window.
4. To remove the list box, click the right mouse button with the mouse pointer inside the list box or click on the terms button in the tool bar.

The term, in its correct syntax, is placed into the print format.

The list contains several "format codes" such as backspace, line feed, carriage return, beep and others designed for special functions when the ticket is printed.

Two selections, "Numeric Variable" and "String Variable" require that you replace the words 'Numeric Variable' or 'String Variable' with the actual variable name that you created in WT-BASIC.

Defining the way numeric variables are printed is accomplished by the following syntax:

SYNTAX:        **{VARIABLE,WIDTH.PRECISION}**

See the example to the left or the print format example on the previous page.

Example: {GROSS, 6.2}

This will print the gross weight as follows: 500.01

Width and Precision are optional expressions. You do not have to use them. By default, a numeric variable is right-justified. Use *width* to define a minimum width of the numeric variable. Use a negative width to left-justify the numeric variable. Use *Precision* to designate the number of positions to be printed to the right of the decimal point.

When defining a string variable, width is used to define a minimum width. If the string variable does not take the minimum width to print, spaces are printed to fill the gap. String variables are Left Justified by default and will print Right Justified if a negative width is used.

When **TIME\$** or **DATE\$** are used in a program's print statement, you may use the following syntax:

**TIME\$(n)** or **DATE\$(n)**

**TIME\$** and **DATE\$** are the two string system variables that have the following optional syntax in a print format:

**{TIME\$,0.n}**

When a **TIME\$** is printed, (n) is used to tell the system what format of **TIME\$** to print. A value of 0, 1, 2, or 3 is used in the (n) expression to print **TIME\$** in the following formats:

#### TIME FORMATS

<b>0 =</b>	<b>24 Hour format with seconds</b>	<b>18:00:00</b>
<b>1 =</b>	<b>AM/PM format with seconds</b>	<b>1:00:00 AM</b>
<b>2 =</b>	<b>24 Hour format without seconds</b>	<b>18:00</b>
<b>3 =</b>	<b>AM/PM format without seconds</b>	<b>1:00 AM</b>

**{DATE\$,0.n}**

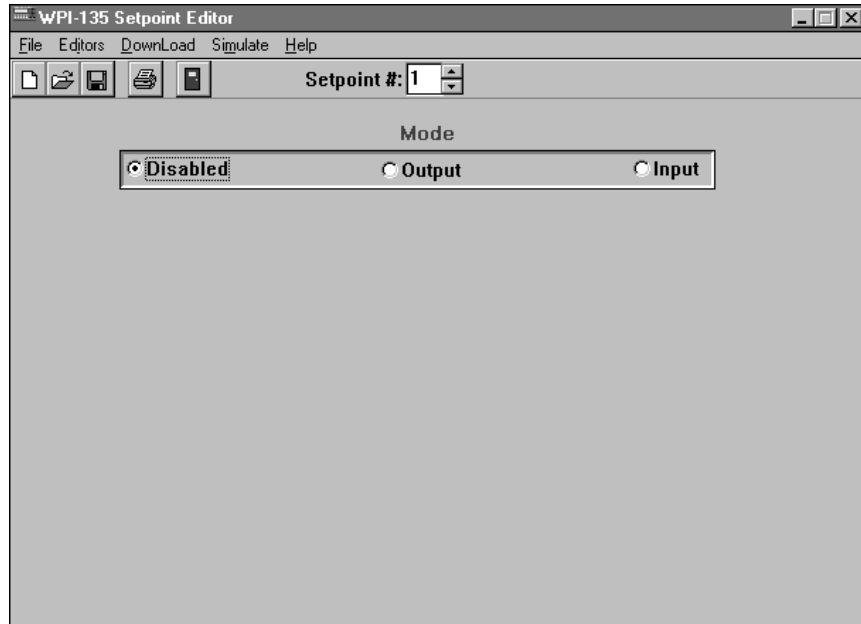
When a **DATE\$** is printed, (n) is used to tell the system what format of **DATE\$** to print. A value of 0, 1, 2, 3, or 4 is used in the (n) expression to print **DATE\$** in the following formats:

Date Formats	Description	MM/DD/YY	DD/MM/YY	YY/MM/DD
0	Numbers with 2-digit year	06-14-99	14-06-99	99-06-14
1)	Spelled Month	Jun 14, 1999	14 Jun, 1999	1999 Jun 14
2)	Numbers with day of week	Mon 06-14-99	Mon 14-06-99	Mon 99-06-14
3)	Spelled Month with day of week	Mon Jun 14, 1999	Mon 14 Jun, 1999	1999 Mon Jun 14
4)	Numbers with 4-digit year	06-14-1999	14-06-1999	1999-06-14

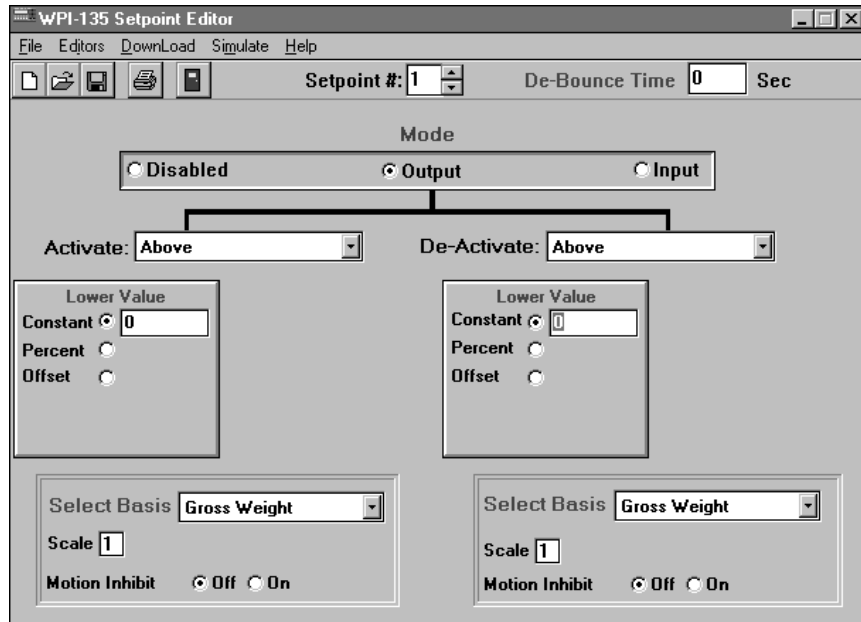
## Setpoint Button



The next button on the WPI-135 SimPoser toolbar is **Setpoint**. Press this button to configure setpoint operation. Depending on what things you choose, the window will display the parameters you need to set. The following three screens show the window as it first appears and as it may appear as you select different parameters.



*PLEASE NOTE: Applications using setpoints should be handled and tested with great care to assure that the system operates in conformance with the stated objectives and design parameters of the system. Always completely test all parameters to the fullest! Manual back up controls should always be installed on the most critical components to assure that the system can be adjusted and/or shut down manually, should the system malfunction or deviate from the proper operation sequence.*



*Setpoints are commonly used as outputs without I/O modules for program interactions like changing the display, continuous output, or diagnostic flags.*

The Setpoint window allows you to define the condition you wish to monitor or detect thus causing the WPI-135 program to trigger a new event (SetPoint Act or Deact).

The Setpoint capabilities of the WPI-135 range from very simple applications such as turning on an alarm when a weight value is reached, up to very sophisticated batching applications specifically tailored to a customer's requirements. The setpoint system is designed for almost limitless flexibility, to allow the application designer to use the system in conjunction with WT-BASIC programming to solve a wide variety of scale user requirements.

Not limited to batching, however, the WPI-135 Setpoint System is adept at solving all types of application needs, such as truck scale control, checkweighing systems, conveyor systems, PLC applications or any other situation where external feedback and control is required.

Use the setpoint window to configure up to 64 setpoints in the WPI-135 System. This dialog screen provides a visual representation of the configuration of a particular setpoint so that you can quickly glance at the form and see how the setpoint is configured.

The screen is separated into the following divisions:

- Setpoint #
- Mode
- De-Bounce Time In Seconds
- Activate/De-Activate Condition - Output Setpoint Only
- Lower/Upper Values - Output Setpoint Only
- Select Basis - Output Setpoint Only

Each of these areas of the screen is explained below.

#### **SetPoint #**

Determines which setpoint is currently active. Use the up/down arrow box to select the active setpoint or enter the number directly into the text box. You may not enter a number smaller than 1 or greater than 64.

### **Mode**

Mode determines what level of operation the setpoint is in. The possible choices are:

- Disabled - Setpoint not active
- Input - Setpoint receives input from external device
- Output - Setpoint sends signal to external device or setpoints are commonly used as outputs without I/O modules for program interactions like changing the display, continuous output, or diagnostic flags.

#### **Disabled**

This is the default state for a setpoint and means that the setpoint is not in use with the current configuration.

#### **Input**

Select Input to receive a signal from an external device, such as a switch. The state of an Input setpoint can be detected using commands in the WT-BASIC programming system. When this mode is selected, the only parameter to choose is De-Bounce Time. De-Bounce time is the time in seconds allotted to receive switch activations, to prevent receiving a double signal. For example, if a De-Bounce time of 1 is entered in the text box, the Input setpoint will only receive one activation input per second, no matter how many activations of the switch occur during the 1 second period.

#### **Output**

Select Output to send a signal to an external device to activate or deactivate the device. Output requires setting various parameters to achieve the proper control of external equipment.

### **De-Bounce**

When the setpoint is used as an output, the debounce time can be used as a timer interval for events to occur by selecting IMMEDIATE in the Activate and Deactivate drop down boxes.

For example, with Activate/Deactivate both set to immediate, and De-Bounce set to 1.0 seconds, the setpoint activates or turns on for one second, then deactivates or turns off for one second, then turns back on for one second. The time between activate events is two seconds.

*In addition to the conditions defined by the setpoint window, you may also force the setpoint on or off from your BASIC program by using the BASIC commands SETPT ON or SETPT OFF.*

### **Activate/Deactivate Condition**

Each Output setpoint must have a condition that activates and deactivates the setpoint. By clicking the combo box next to Activate and Deactivate, a list is displayed for you to select this condition. You must select both an Activate and Deactivate Condition in order for the setpoint to work properly. Make sure you fill in both sides of the form.

The following conditions are available:

- Above
- Below
- Inside
- Outside
- Motion
- No Motion
- Center of Zero
- Not Center of Zero
- BASIC Control
- Immediate
- Accum Operation
- Print Operation
- Zero Operation
- Tare Operation
- Tare Key
- Select Key
- Print Key
- Units Key
- Zero Key
- Equal To
- Logical AND
- Logical OR
- Logical XOR

Selection of the above conditions determine the additional parameters that will be required to make the setpoint functional. Each of these will be described in the following sections and each section will apply to both Activate and Deactivate.

#### **1. Above/Below**

Above/Below indicates that the setpoint will activate or deactivate when the scale reading is either Above or Below the selected value. When either of these selections is made, the "Lower Value" box on the screen will become active.

##### **Lower Value**

Indicates the value that the setpoint will activate on. Use the mouse to click on the selection in the Lower Value box.

**Constant** - Indicates that the value will be a fixed value, using the same standard as the calibration Unit of Measure selected for the system, such as 5000 lbs. When Constant is selected, the text box to the right (Value box) becomes active. Enter the constant value in this box.

**Percent** - Indicates that the value will be a percentage of a Variable amount. When selected, the Variable text box is enabled. Enter the percentage in the box to the right of the form (Value box) and the Variable Name in the box under "Variable". Variables can be used in the WT-BASIC program to control operations of the setpoint system. You may enter both positive and negative percentage amounts and amounts greater than 100% (i.e. 1000%)

**Offset** - Indicates that the value of a Variable from your BASIC program, plus this offset value in cal units, will control this setpoint.

For example, the value may be 100 lbs. more than the value currently contained in variable "TestAmount" (a variable defined in the WT-BASIC program). For this example, you would enter 100 in the Value text box and the name "TestAmount" in the Variable text box. You may use both positive and negative Offset amounts.

### **Select Basis**

The Basis defines the Weight or other value that the selected setpoint will activate on. By clicking the Combo box next to Select Basis, a list is displayed for you to select from. Depending on your selection, additional selections in the Select Basis box will be enabled for you to select from.

#### ***Gross Weight, Net Weight, Tare Weight, Minimum Weight, Maximum Weight, Rate of Change, Gross Weight Total, Net Weight Total, Count***

If one of the above Basis selections is made, you need to make the following additional selections from the Select Basis box:

Scale Number - Enter the scale number that this selection will apply to in the Text box.

Motion Inhibit - Click On to enable Motion Inhibit, or Off to disable Motion Inhibit.

#### ***Count Total, Transaction Total, Piece Weight***

No selections need to be made.

### **Variable**

If the above Basis selection is made, you need to make the following additional selection from the Select Basis box:

Variable - Enter the name of a Variable used in the WT-BASIC program for this application.

Motion Inhibit - Click On to enable Motion Inhibit, or Off to disable Motion Inhibit.

## **2. Inside/Outside**

Inside/Outside indicates that the setpoint will activate or deactivate when the scale reading is either Inside or Outside a range of selected values. When either of these selections is made, both the "Lower Value" and "Upper Value" boxes on the screen will become active allowing you to enter a range of values to meet this criteria. Make your selections in the same manner as described in 1. *Above/Below*, but make sure you fill in selections for both the Lower and Upper Value boxes. Both will become active when either of these selections is made.

## **3. Motion, No Motion, Center of Zero, Not Center of Zero**

The setpoint will activate or deactivate when any one of these chosen conditions is met. The only condition that has to be selected is the Scale Number. Enter the number in the adjacent text box.



#### 4. BASIC Control, Immediate, Accum Operation, Print Operation, Zero Operation, Tare Operation

The setpoint will activate or deactivate when any of the above operations are in control or are activated, as in the case of the **Zero** key being pressed and the zero operation successfully completed. There are no additional parameters to be set when one of these conditions is selected. The entire form below Activate/De-Activate will become disabled.

#### 5. EQUAL TO, LOGICAL AND, LOGICAL OR, LOGICAL XOR

The setpoint will activate or deactivate based upon the logical state of the other setpoints.

#### Example 1:

Following are several examples of how setpoints operate. There is much more you can do with setpoints once you familiarize yourself with all the variables and conditions you can use to trigger the setpoints.

**Figure 21**  
Example 1

In example 1, setpoint 1 is an output. The setpoint will activate below a constant value of 25 lbs gross weight. When the gross weight goes above 35 pounds, the setpoint will deactivate. Weight readings are coming from scale #1 and the motion inhibit is off.

## Example 2:

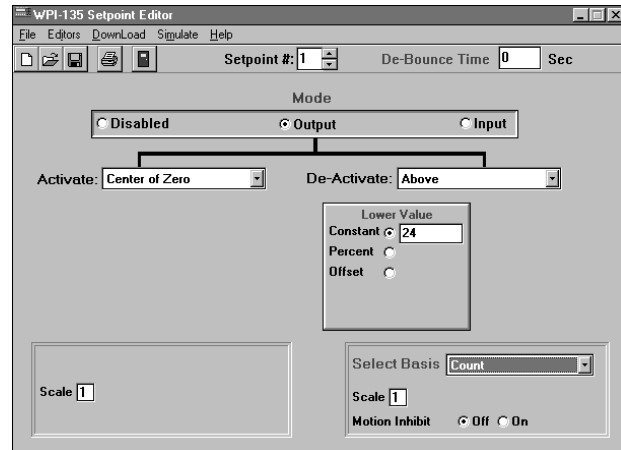
The screenshot shows the 'WPI-135 Setpoint Editor' window. At the top, there is a menu bar with 'File', 'Editors', 'Download', 'Simulate', and 'Help'. Below the menu bar is a toolbar with icons for file operations. The main configuration area is titled 'Setpoint #3' and 'De-Bounce Time 0 Sec'. Under the 'Mode' section, three radio buttons are present: 'Disabled', 'Output' (which is selected), and 'Input'. Below the mode selection, there are two dropdown menus: 'Activate: Inside' and 'De-Activate: Outside'. The central part of the window contains four panels, each with a 'Lower Value' and an 'Upper Value' section. Each section has three radio buttons: 'Constant' (selected), 'Percent', and 'Offset'. The 'Constant' values are set to 50 for the lower value and 150 for the upper value in all four panels. At the bottom, there are two identical blocks. Each block has a 'Select Basis' dropdown menu set to 'Net Weight', a 'Scale' input field set to 1, and a 'Motion Inhibit' section with 'Off' and 'On' radio buttons, where 'On' is selected.

**Figure 22**  
Example 2

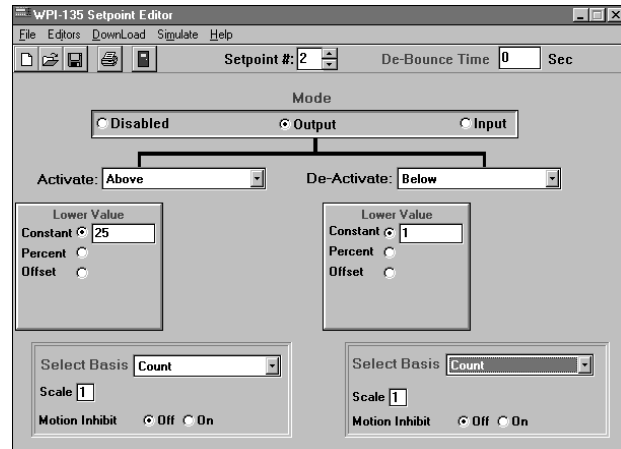
Figure 22 shows the setpoint window for example 2. In this example we have chosen Setpoint 3 as an output. The setpoint will activate when the net weight value is between 50 and 150 pounds and will deactivate outside of this range.

### Example 3

In this example we will use two setpoints #1 (see Figure 23) and #2 (see Figure 24).



**Figure 23**  
Example 3-Setpoint #1



**Figure 24**  
Example 3 - Setpoint #2

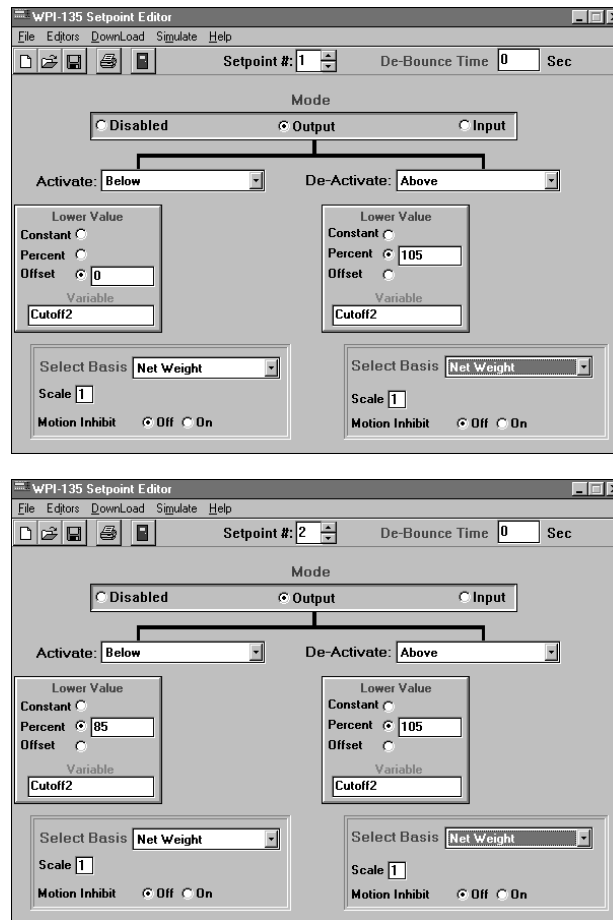
In this third example we are running a **very** simplified batching sequence. We are opening a valve or gate to drop gumballs into a hopper on a scale. We want to count out 25 gumballs and dump the hopper, then repeat the process. (Keep in mind that to run a batching procedure in real life it will probably be necessary to use a WT-BASIC program in conjunction with the setpoint configuration.)

Setpoint 1 controls the valve or gate to allow gumballs to roll in to the hopper. It is set up as an output and activates when the gross weight is at center of zero. The gumballs will roll into the hopper until the count is over 24. The valve will shut and the next setpoint (#2) which controls the dump gate on the scale hopper will activate since the count is now 25 or above. When the count on the scale hopper drops below 1, the gate closes and the fill valve reopens.

As was stated earlier, this is an extremely simplified batching program. With a WT-BASIC program you can design very sophisticated batching sequences.

#### Example 4

Figure 25 shows two screens using variables. The explanation for these is below.



**Figure 25**  
**Example 4**

In this fourth example we are controlling two setpoint outputs as cutoffs, similar to the way older indicators have worked. The first screen in Figure 23 refers to Cutoff1 as the variable name for control of setpoint #1. The second screen refers to Cutoff2 as the variable name for control of setpoint #2. Both variable names will have values assigned to them in the WT-BASIC program via the front panel.

Setpoint #1 activates when the net weight is below an offset of 0 of the variable Cutoff1. It deactivates when the net weight is above an offset of 24 in addition to the variable Cutoff1.

Setpoint #2 activates when the net weight is below 85% of the variable Cutoff2. It deactivates when the net weight is above 105% of the variable Cutoff2.

Example 5

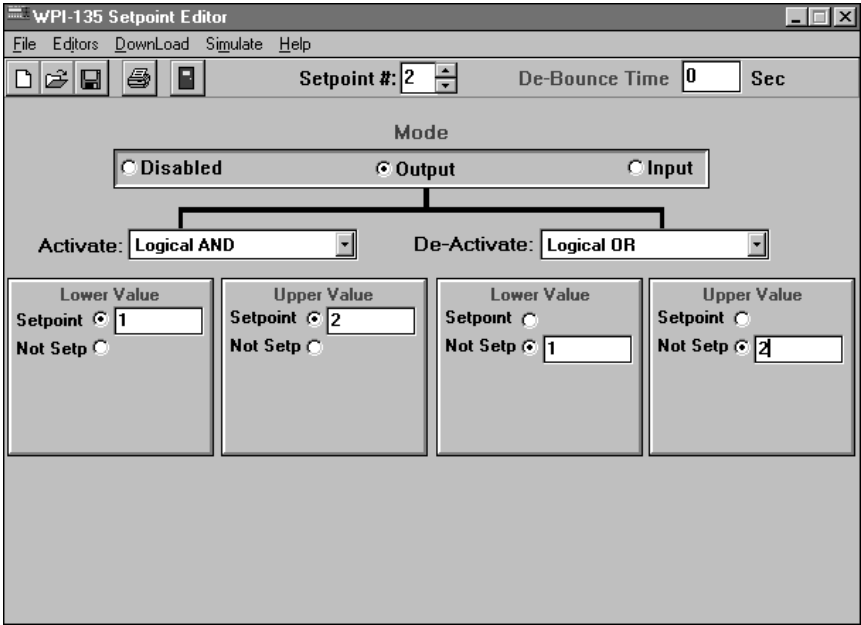
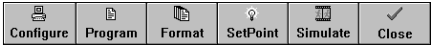


Figure 26  
Example 5

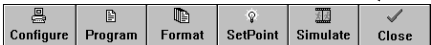
In this example, shown in Figure 26, we use the logical AND and OR conditions. Setpoint #6 will activate when Setpoints 1 AND 2 are on and deactivate when either Setpoint 1 OR 2 is off.

Simulate Button



The next toolbar button is **Simulate**. Click this button to start the WPI-135 simulation. This is the same as clicking the Simulate command on the command line. This was covered earlier in this manual.

Close Button



The last toolbar button is **Close**. Click this to close your WPI-135 SimPoser program. If you have made any changes in a program, a dialog box will pop up asking if you want to save the changes.

# Sample Application Program Summaries

## Batching

2spd1ing.135	Two speed single ingredient batching application.
4ingbatc.135	4 Ingredient single speed batching application.
Flow9.135	Demo using ROC to control flow rate(using setpoints) "LB/HR".
Jogbatch.135	Jog softkey example.
Wi-1106.135	Multi-scale (3-scales), dual cutoffs, single speed.

## Checkweighing

Chkweig2.135	Advanced checkweigher program with set points.
Setchk1.135	Simple checkweigher program.

## Counting

Count5.135	Simple counting scale using the Dribble sample method. Printing –sample size, piece weight, net weight, and count total.
Count6.135	Simple counting scale offering both Bulk & Dribble sample methods. Printing –Barcode labels out of Port number 2 using print Formats 4, 5, and 6 for a label printer.

## InMotion

Inmo5.135	Simple conveyor scale application.
Firminmo.135	Firmware level inmotion system application

## Miscellaneous

Sys_err.135	Example application for using SUB SYSTEM_ERROR and the ERR keyword.
Tare100.135	Application for tare channel store/recall based on an alphanumeric ID.
WI127.135	Application to simulate WI-125/WI-127 operation.
WI127R.135	Application to simulate the WI-125/WI-127 operation with 2 remote bases.

## Printers

Conout.135	Provides continuous output of GROSS weight out port 1.
Enq_cr	Remote request from a PC application
Orion.135	Eltron Orion sample label printer application.
Orion1.135	Eltron Orion sample label printer application. Examples from Orion printer price sheet.
Rs485.135	RS-485 multi-drop example application.
Tm_295a.135	Epson TM-295 ticket printer example.

## **Rail (Track Scales)**

Wt-line4.135

Standard Weigh-Line application.

## **RD (Remote Display)**

Rd4000\_6.135

RD-4/6000 with 6 digit display sample.

Rd4000\_8.135

RD-4/6000 with 8 digit display sample.

Rd-125.135

RD-125 sample application.

## **Sales Demos**

Chkweigh.135

Checkweigher sales demonstration.

Partct.135

Part Counter sales demonstration.

Slide.135

Slideshow for sales demonstration.

Specs.135

WPI-135 specification slideshow for sales demonstrations.

Tareiso.135

Multi-Channel tare database with ISO-9000 tracking information.

Wi110.135

WI-110 with 10 tare registers.

Wi110bat.135

WI-110 with dual cutoffs.

Wi120.135

WI-120 emulation sales demonstration.

## **Truck Scale**

10544f.135

Axle weigh truck scale.

9459d.135

GTN or 70 foot axle weigh scale.

Freetrk5.135

Inbound/Outbound truck scale application example.

Freetrk6.135

Dual indicator Inbound/Outbound truck scale application example.

Tare100.135

Multi-Channel tare database.

**Microsoft® Word documents explaining these programs are available in the  
C:\wt\WPI135\docs folder or the folder you designated during installation.**

Appendix 1: Display Samples



#1



#9



#2



#10

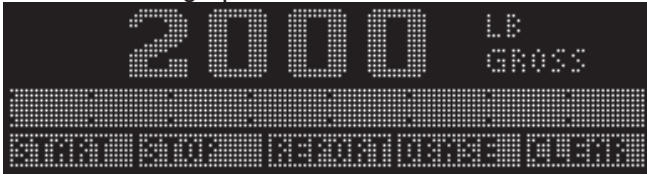


#3



#11

Bar graph below shown at 100%



#4

Bar graph below shown at 50%



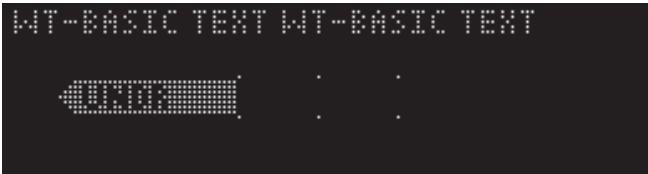
#12

Checkweigher shown in overload condition below.



#5

Checkweigher shown in underload condition below.



#13



#6



#14

Checkweigher shown in accept condition below.



#7



#15



#8



#16





#17



#26



#18



#27



#19



#28



#20



#29



#21



#30



#22



#31



#23



#32



#24



#33



#25

The following are multi-scale displays. If all the lines are not used for scales, they are available for Basic text. #34, 35, 40 and 41 are small basic text. #36, 37, 42 and 43 are large basic text.



#34

#39



#35

#40



#36

#41



#37

#42



#38

#43



## Appendix 2: WT-BASIC Interpreter Command Set

### WT-BASIC

*If you hold in the **CLEAR** key when powering up the WPI-135, the WT-BASIC program which is currently resident in the indicator will be temporarily disabled until the next power up.*

*This is used when troubleshooting to eliminate the BASIC program as the source of a problem or to disable continuous output so you can download a new program.*

The following pages contain the WT-BASIC interpreter command set you use to create programming for the WPI-135.

This command set goes beyond the normal BASIC language by adding many commands found only in this expanded WT-BASIC language. This makes programming the WPI-135 more flexible than was possible with the original BASIC language. It adheres closely to the QBASIC language included in current versions of MS-DOS. Syntax examples are included where necessary in the following pages.

There is a difference between WT-BASIC and QBASIC. In QBASIC a main program body runs and calls out subroutines and performs some task. In WT-BASIC, you design what are called Event Handlers. These are BASIC subroutines that activate only upon the occurrence of some event. What is an event? It can be any indicator or scale related activity such as

- pressing a soft key
- pressing a hard key
- on a time interval
- reaching a certain gross weight
- scale stability
- scale motion
- serial port data
- setpoint activation or deactivation
- input switch state change
- etc.

The idea is to handle the event then exit the subroutine so that other event handlers can be called. You can create your own names for subroutines that are not triggered by an event but can be called from other subroutines that are triggered (or handled) by an event.

The actual list is very long, but you should get the idea that you can tie a WT-BASIC program to many events. The program you write can cause an unlimited number of things to occur, whether prompting for data, opening valves, printing tickets or sending data to a computer. It all depends on your imagination and your particular application.

See the included examples for an idea of where to start. For those new to programming or those accustomed to other languages, a book on QBASIC is a good place to begin.

## Event Subroutines



### Attention

*When defining this subroutine the key buffer must be emptied using either INKEY\$ or an INPUT statement. If the key buffer is not cleared, the indicator may hang.*



Anytime one of the following events is triggered in the instrument, program execution is transferred to the corresponding sub procedure (if the subroutine procedure exists). If more than one event is triggered, they are executed in the order they were tripped. The syntax for a subroutine is identified by the word "SUB" followed by the name of the event subroutine. The subroutine is ended by the command "END SUB"

Some events have default actions that are executed when the event is triggered. The default action is not executed if a subroutine exists in the BASIC program. An example is ZERO\_KEY. If the subroutine ZERO\_KEY is in the BASIC program the code there is executed instead of the firmware level ZERO function. i.e.:

```
SUB ZERO_KEY
  x=4
  y=3
  PRINT X,Y
END SUB
```

### ACCUM\_ABORT

This event is activated if the DOACCUM command is not successfully processed.

### ACCUM\_OPER

This event is activated if the DOACCUM command is successfully processed.

### CLEAR\_KEY

This event is activated whenever the CLEAR key is pressed.

### COM1\_MESSAGE

This event is activated when a complete message has arrived in the COM1 serial port input buffer. A "Complete Message" is determined by the End of Message Character in the Serial Ports Configuration.

### COM2\_MESSAGE

Same as COM1\_MESSAGE only for Port #2 primary receive line.

### COM3\_MESSAGE

Same as COM1\_MESSAGE only for Port #3 primary receive line.

### COM4\_MESSAGE

Same as COM1\_MESSAGE only for Port #4 primary receive line.

### ENTER\_KEY

An event is activated whenever the ENTER key is pressed.

### ENTRY\_KEY

This event is activated whenever any key on the remote keyboard is pressed (except function keys).

### ESC\_KEY

An event is activated whenever the ESCAPE key is pressed.

### Fx\_KEY

For softkeys. These events queue when the front panel key is pressed or an external keyboard key is pressed. (x; 1-5)

### F6 to F10

Only available on an external keyboard or in the simulator. (x; 6-10)



## Attention

*When defining this subroutine the key buffer must be emptied using either INKEY\$ or an INPUT statement. If the key buffer is not cleared, the indicator may hang.*



## NUMERIC\_KEY

This event is activated whenever any numeric key (0-9) is pressed.

## PRINT\_ABORT

This event is activated if the DOPRINT command is not successfully processed.

## PRINT\_KEY

This event will be activated if the **PRINT** key is pressed on the WPI-135.

## PRINT\_OPER

This event is activated if the DOPRINT command is successfully processed.

Default: Prints the default print format.

## SELECT\_KEY

This event will be activated if the **SELECT** key is pressed on the WPI-135.

## SELECT\_OPER

This event is activated if the DOSELECT command is successfully processed.

## SETPTx\_ACT

These events queue when the SetPoint turns on. There is one subroutine for each setpoint. (x; 1-64)

## SETPTx\_DEACT

These events queue when the SetPoint turns off. There is one subroutine for each setpoint. (x; 1-64)

## SYSTEM\_ERROR

Whenever a BASIC error occurs, this event is queued and the error that triggered this event is held in ERR. See also ERR.

## SYSTEM\_SETUP

This event is queued by first issuing the setpwd keyword. The programmer should then hold in the **ESCAPE** key for five seconds and enter the password defined by the setpwd command. If the passwords match, SUB SYSTEM\_SETUP will be queued. If the passwords don't match the prompt will be aborted.

Example:

```
SUB SYSTEM_STARTUP
word$="135"
setpwd(1,word$)
END SUB
```

```
SUB SYSTEM_SETUP
dispmode=10
Cls
Print "HELLO WORLD!!!"
END SUB
```

## SYSTEM\_STARTUP

If a SYSTEM\_STARTUP procedure exists in the WT-BASIC program, it always gets executed at the time of startup or upon exiting CAL or CONFIG menus.

## SYSTEM\_TIMER

This event is activated on interval determined by Settimer command. This may generate multiple queues in the event buffer.

Defining this blocks the normal default operation from occurring when you press a key on the front panel of the WPI-135. Use the **doxxx** commands to make the regular function work. See **doxxx**.



## SYSTEM\_TIMER2

This event is activated on interval determined by Settimer command. This will generate one queue in the event buffer.

## SYSTEM\_TIMER3

This event is activated on interval determined by Settimer command. This will generate one queue in the event buffer.

## SYSTEM\_TIMER4

This event is activated on interval determined by Settimer command. This will generate one queue in the event buffer.

## TARE\_ABORT

This event is activated if the DOTARE command is not successfully processed.

## TARE\_KEY

This event is activated if the TARE key is pressed.

Default : Performs a firmware level tare function.

## TARE\_OPER

This event is activated if the DOTARE command is successfully processed.

Default: Changes the display value to NET if the display value is Gross and Tare does not equal 0 or changes the display value to GROSS if the display value is NET and Tare = 0.

## UNITS\_KEY

This event will be activated if the UNITS key is pressed on the WPI-135.

## UNITS\_OPER

This event is activated if the DOUNITS command is successfully processed.

## ZERO\_ABORT

This event is activated if the DOZERO command is not successfully processed.

## ZERO\_KEY

This event is activated if the ZERO key is pressed.

Default : Performs a firmware level zero.

## ZERO\_OPER

This event is activated if the DOZERO command is successfully processed.

## NETWORK\_OPER

If the WPI-135 initializes the network card properly, this event will be queued.

## NETWORK\_ABORT

If the WPI-135 fails to initialize the network card, this event will be queued.

## BASE\_OPER

If the dbase keyword was successful, this event is queued.

## NETWORK\_STATUS

If the network status changes, this event is queued.

-offline status change

Check the **neterr** keyword for the status change. **clearerr** can be used to reset the **neterr** keyword so multiple status changes can be viewed.



Defining this blocks the normal default operation from occurring when you press a key on the front panel of the WPI-135. Use the **doxxx** commands to make the regular function work. See **doxxx**.

---

## Variables

---

*Weigh-Tronix strongly recommends the use of the MUSTDIM command in the SYSTEM\_STARTUP event subroutine.*

### Numeric Variable

The user defined variables can be numeric variables or string variables. The variables can be defined by just referencing them. The type of the variable defined is dependant on the last character of the name of the variable. A variable can be defined also by the command "**DIM VARIABLE NAME**". For purposes of debugging, the command **MUSTDIM** can be placed in the WT-BASIC program forcing all variables to be defined exclusively through the dimension command.

A numeric variable represents a group of numbers only that mathematical functions may be performed on.  
A numeric variable is assigned using "=" i.e. **X = 42**.  
All variables without a special dimensioning character or with a "#" character on the end becomes a **15 digit, double precision, floating point**.  
By using a "%" character, the variable is dimensioned as a regular **integer**.  
An "!" character dimensions the variable as a **single precision floating point**.  
The "&" character dimensions the variable as **Long Integer**.

### String Variable

A string variable may represent a group of letters, numbers or a combination of the two that no math functions can be performed on.  
A string variable is assigned using "=" using quotes to surround the default string. i.e. **X\$ = "TARE"**.  
A standard **string** variable can store a string 16 characters or less and is dimensioned by the character "\$".  
To define a shorter or longer string, the string variable can be succeeded by an "\*" and an integer (1-64). The integer defines the length of the string. i.e. "**DIM X\$\*10**" defines a string X\$ with storage space for 10 characters.

### System Variable

These variables have already been predefined by Weigh-Tronix in the WPI-135 WT-BASIC command set.  
Items that bear an asterisk(\*) can be set through a WT-BASIC (TARE=100) statement, although they can't be part of an input statement. The rest of the values are set only through firmware level calls. Also, when these are set, they **must** be in calibration unit of measure. When they are recalled the system returns the variable in current unit of measure.  
All system variables have the ability to include a scale number in parenthesis after the keyword to select a scale number to access. If the parenthesis is not there the current scale is assumed. Scale#=0 returns the total of all scales.  
Example: CAPACITY(3) returns the current capacity of scale three.



*Items that bear an asterisk(\*) can be set through a WT-BASIC equivalent statement, although they can't be part of an input statement.*

*All system variables have the ability to include a scale number in parenthesis after the keyword to select a scale number to access. If the parenthesis is not there the current scale is assumed. Scale#=0 returns the total of all scales.*

*System flags (CZERO,, MOTION, OVERLD, UNDERLD) will return a true (-1) or a false (0) value depending on their current condition.*

## **ADCCNTS**

Returns filtered raw Counts (not scaled to current unit of measure) from the currently selected base. Counts for Quartzell and Analog Base have different ranges.

## **CAPACITY**

Returns the capacity of the currently selected scale.

## **COUNT**

Is the actual calculated count value based on the Net Weight.  
Count = Net/Pcwt.

## **COUNTTOT\***

Returns or sets count total used by the DoAccum function to total the count.

## **CURSCALE\***

Returns or sets the current active scale number selected. (0 through 8)

SYNTAX:

CURSCALE=3

## **CURUNIT\***

Returns or sets the value of the current units.

0=lbs	2=grams	4=lb oz	6=custom2	8=custom4
1=kg	3=oz	5=custom1	7=custom3	

## **CURUNIT\$**

Returns the label of the current unit of measure for the current scale, i.e. Kg, LB.

## **CZERO**

Center of Zero flag. Returns -1 if center of zero, 0 if not center of zero (on the current scale)

## **DATE\$(n)\***

Returns or sets the internal date. (n) is the format syntax. See also *Print Format* section of this manual.



#0	Gross
#1	Net
#2	Tare
#3	Minimum
#4	Maximum
#5	Rate of Change
#6	Gross Total
#7	Net Total
#8	Count Total
#9	Transaction Total
#10	Count
#11	Value
#12	Piece Weight
#13	A to D Counts

*System flags (CZERO,,  
MOTION, OVERLD,  
UNDERLD,) will return a true  
(-1) or a false (0) value de-  
pending on their current  
condition.*

## **DIVISION\***

Returns or sets the division size of the currently selected scale.

## **DISPLAY**

Returns the current system variable being displayed. (Not the actual number on the display. See list at left.)

## **ERR (use only in SUB SYSTEM\_ERROR event routine)**

This keyword returns the error number corresponding to a given error which occurred in a WT-BASIC application. The following error numbers and errors are available. Err returns 0 if no error has occurred since power-up. The last error remains until a new error has occurred or until power-down.

### **SYNTAX:**

X=ERR  
or  
PRINT ERR

0=	"No error"	12=	"NEXT without FOR"
1=	"syntax error"	13=	"too many nested GOSUBs"
2=	"unbalanced parenthesis"	14=	"RETURN without GOSUB"
3=	"no expression present"	15=	"double quotes needed"
4=	"equals sign expected"	16=	"String Expected"
5=	"not a variable"	17=	"Variable Name Too Long"
6=	"Label table full"	18=	"Var Not Defined (DIM)"
7=	"duplicate label"	19=	"too many nested WHILEs"
8=	"undefined label"	20=	"WEND without WHILE"
9=	"THEN expected"	21=	"Division by Zero"
10=	"TO expected"	22=	"EEPROM Sentinel"
11=	"too many nested FORs"	23=	"Ram Sentinel"

## **CLEARERR(errtype)**

Resets the err or neterr keywords to zero. Specify which one to clear.

errtype = 1: err Keyword  
          2: Neterr (1)  
          3: Neterr (2)

## **GROSS**

Returns the gross value of the current scale rounded off by division size.

## **GROSSTOT\***

Returns or sets the gross total used by the DoAccum function to total the gross weights. (Grosstot=Ø to reset)

## **MAXPEAK\***

Returns or sets the highest value on the current scale rounded by division size.

## **MINPEAK\***

Returns or sets the lowest value on the current scale rounded by division size.

## **MOTION**

Motion flag. Returns -1 if motion, 0 if no motion on the current scale.

*System flags (CZERO,,  
MOTION, OVERLD,  
UNDERLD) will return a true  
(-1) or a false (0) value de-  
pending on their current  
condition.*

### **NET**

Returns the net (net=gross - tare) value on the current scale rounded by division size.

### **NETTOT\***

Returns or sets net total used by the DoAccum function to total the net weights.

### **OVERLD**

Overload flag. Returns -1 if overload, 0 if not on the current scale. (gross up to 120% of capacity)

### **PCWT\***

Returns the pieceweight value of the current scale.  
(pcwt = 0.0017)

### **RAWGROSS**

Returns the unrounded gross weight of the current scale.

### **RAWNET**

Rawnet is calculated from Rawgross-tare.

### **RAWTARE**

Rawtare is calculated by converting the tare to A/D counts. Then the tare(in counts) is multiplied by the calibration span factor. The result is not rounded to the nearest division.

### **ROC**

Returns the rate of change of the current scale per configured parameters.

### **TARE\***

Returns or sets the tare value on the current scale. The display will not switch to net mode automatically. Use the ACTVALUE command. Rounded by division size.

### **TIME\$(n)\***

Returns or sets the internal time. (n) is the format syntax. See also *Print Format* section of this manual.

### **TRANSTOT\***

Returns or sets transaction total used by the DoAccum function to total the number of accumulations.

### **UNDERLD**

Underload flag. Returns -1 if underload, 0 if not on current scale. (gross up to -120% of capacity)

## Operators

*Logical operators return a true (-1) or false (0). Any nonzero value is true.*

*Example:*

*14 AND 2 results in a -1 or true.*

*14 AND 0 results in a 0 or false.*

### Logical Operators

For an expression using a logical operator to be considered true, the following conditions must be met:

#### AND

both parts of the expression must be true

#### OR

either part of the expression must be true

#### XOR

both parts of the expression must be the opposite

#### EQV

both parts of the expression must be the same

#### IMP

first part true, second part false

#### NOT

inverse of expression or opposite condition

### Arithmetic Operators

The arithmetic operators are listed in order of precedence. Multiplication, division, Integer Modulo, and Integer Divide have same precedence. Addition and subtraction have the same precedence. Items with the same precedence are evaluated left to right. Operations within parenthesis are performed first. Inside the parenthesis, the usual order of precedence is maintained.

() Parenthesis

- Negation

^ Exponent or Power of Operator

The result of the Power of operator(^) is the product of multiplying the first operand by itself, the second operand times.

Example:

```
SUB SYSTEM_STARTUP
```

```
dispmode=10
```

```
Cls
```

```
Print (2^8);"=";(2*2*2*2*2*2*2*2);"=256"
```

```
END SUB
```

\* Multiplication

/ Division

**m or MOD** Modulus Operator (Requires a space on each side of operator)

The result of the modulus operator (m) is the remainder when the first operand is divided by the second.

Example: 10 divided by 3 = 3 with a remainder of 1. The modulus is 1.

The "m" must always be lower case and must always have a space before and after it.

Example: Returns the value of 1

```
SUB SYSTEM_STARTUP
dispmode=10
cls
VALUE=(10 m 3)
PRINT "The remainder is"; VALUE
END SUB
REM The VALUE is one
```

**\** Integer Divide (Requires a space on each side of operator)

This operator divides 2 numbers and returns the Integer portion of the computation.

Example: 10 divided by 3 = 3 with a remainder of 1 (3 is the result)

```
SUB SYSTEM_STARTUP
dispmode=10
Cls
VALUE=(10\3)
PRINT "THE PRODUCT IS"; VALUE
END SUB
REM The VALUE is three
```

**+** Addition

**-** Subtraction

### **Relational Operators**

**=** Equal to

**>** Greater than

**<** Less than

**<=** Less than or equal to

**>=** Greater than or equal to

**< >** Not equal to

**> <** Not equal to

## Command Statements

### Input/Output

*Maximum of 40 characters/  
display line for small font.  
Maximum of 30 characters/  
display line for large font.  
Both fonts are fixed width.*

*Any keyword with a printable  
message in quotation marks  
must have a space separating  
the keyword from the message.*

Example: Print "Hello"  
                    ↑  
                    Space

*These input options remain  
active for all input statements  
until turned off by using  
INPUTOPT(0,0,0).*



#### PRINT

To output data to the screen. The area of the display dedicated to WT-BASIC messages is determined by the active display mode.

SYNTAX: PRINT "THIS WILL GO TO THE DISPLAY"

#### PRINT #x

To output data to a serial port.

SYNTAX: PRINT #1, "THIS WILL GO OUT OF PORT #1"

PRINT #2, "THIS WILL GO OUT OF PORT #2"

#### FMTPRINT(n)

Calls a predefined print format (n = 1 to 32) that will be sent out a serial port, as specified in destination port settings.

#### INPUT

To receive data input from the keypad or a remote keyboard. The specified prompt is shown on the display. The input softkey interface is selected by the variable type to be input.

SYNTAX: INPUT "Enter id", ID\$

OR

INPUT "Enter#", VALUE

#### INPUTOPT

This stands for input options. This keyword will allow the following features in an input statement:

SYNTAX:

inputopt(noecho, timeout, noclear)

noecho	<p>If noecho is set to "0" then the data entered during an input statement will be visible.</p> <p>If noecho is set to "1" or "-1" then the data entered during an input statement will appear as an asterisk "*". This will be useful for password protecting softkeys.</p>
timeout	<p>If timeout is set to "0", the input statement will wait for an <b>ENTER</b> or <b>ESC</b> key depression.</p> <p>If timeout is set to a number, then the input statement will timeout after that amount of time if a key is not hit. The timeout feature is reset each time a key is hit.</p>
noclear	<p>If noclear is set to "0" the variable used for the input statement will be cleared upon data entry.</p> <p>If noclear is set to "1" or "-1" the variable used for the input statement will NOT be cleared upon data entry, and the character/number entered will be concatenated to the existing data.</p>

*INKEY\$ is typically used in a program loop for detecting a specific key depression.*

*Also used to clear the value of "LASTKEY" or "KEYHIT"*

## INKEY\$

Returns one character read from the remote keyboard or from the WPI-135 front panel keypad as a string. Returns an empty string "", if no character is available.

SYNTAX: X\$ = INKEY\$

## KEYHIT

This command is used in program loops to detect a key hit on the front panel of the WPI-135. This command only recognizes numeric keys on the front panel of the WPI-135.

SYNTAX: x = KEYHIT

The variable **x** is returned as a true, negative one (-1), or a false, zero (0) depending on whether a key is hit or not.

## LASTKEY

LASTKEY returns the ASCII value of the last key pressed on the keyboard as a numeric value.

SYNTAX: x = LASTKEY

<i>Last Key</i>	<i>Values Returned</i>	<i>Last Key</i>	<i>Values Returned</i>
F1	15104	.	46
F2	15360	0	48
F3	15616	1	49
F4	15872	2	50
F5	16128	3	51
F6	16384	4	52
F7	16640	5	53
F8	16896	6	54
F9	17152	7	55
F10	17408	8	56
TARE	5120	9	57
ZERO	11264		
ESCAPE	27		
ENTER	13		
CLEAR	11776		

## LASTKEY1

lastkey1 returns the ASCII value for the last key pressed on the front panel keypad. The ASCII value of lastkey1 resets itself to 0 after it has been assigned or printed.

Example:  
SUB ENTER\_KEY  
dispmode=6  
x=lastkey1  
x1=lastkey1  
print x,x1  
END SUB



Displayed sample of above SUB ENTER\_KEY routine showing the enter key as the last key press with its character value 13 which is reset to a 0 value after printed to the display screen.

**Select X:**

1

2

### **ASK(MSG\$)**

ASK is a function that allows you to prompt the operator for a Yes/No type response. The ASK command accepts 2 optional arguments which allow the user to change the legends of the F1 key and the F5 key. The ASK(MSG\$) function returns a negative one (-1) or true for a YES response and a zero (0) or false for a NO response.

SYNTAX: ASK("MSG\$", "TRUE", "FALSE")

An example would be as follows:

```
SUB SYSTEM_STARTUP
dispmode=10
if ask("Select X:","1","2") then x=2 else x=1
cls
print "X=",x
END SUB
```

**Card Type:**

*Profibus* = 1

*Interbus* = 16

*DeviceNet* = 37

*ModBus+* = 64

*ControlNet* = 101

*ModBus/TCP* = 128

*EtherNet - Raw Sockets* = 128

### NETERR(cardnum)

This keyword returns an error number associated with a network card.

cardnum = 1 or 2

Return value:

0 = no error

1 = not enabled

2 = card type configured does not match card installed

3 = initialization failed

4 = offline

### GETNET

Returns the numeric value (x) that represents the current setting of the network card selected from the tables below.

SYNTAX: x=getnet(cardnum,y)

Pick 'y' from the table below (1 to 17)

1: Card type (see note at left)

2: Not used

3: Network card serial number

4: Not used

5: Input memory I/O size

6: Input memory RAM size

7: Input memory total size

8: Output memory I/O size

9: Output memory RAM size

10: Output memory total size

11: Software revision

12: Boot loader software revision

13: Not used

14: Card enable (0 = disable, 1 = enable)

15: Input memory reset state (0 = clear, 1 = freeze)

16: Output memory reset state (0 = clear, 1 = freeze)

17: WT standard output enable state (0 = disable, 1 = enable)

### SETNET

Sets the numeric value (x) which represents the current setting of the network card selected from the tables below.

SYNTAX: setnet(cardnum,y,x)

Pick 'y' from the table below:

5: Input memory I/O size

6: Input memory RAM size

7: Input memory total size

8: Output memory I/O size

9: Output memory RAM size

10: Output memory total size

13: Reinitialize network card

14: Card enable (0 = disable, 1 = enable)

15: Input memory reset state (0 = clear, 1 = freeze)

16: Output memory reset state (0 = clear, 1 = freeze)

17: WT standard output enable state (0 = disable, 1 = enable)

19: Set address bytes 0-3 Example: IP 10.5.0.7

x=10 a=5 b=0 c=7

20: Set address bytes 4-7 Example: SUBNET MASK 255.255.255.0

x=255 a=255 b=255 c=0

21: Set address bytes 8-11 Example Gateway 0.0.0.0

x=0 a=0 b=0 c=0



## MAP(card#,sclNum,exLoc,intLoc,cmd,I\_O,"var")

card#	Defines the card number for the system. 1 or 2 is used.
sclNum	Defines which scale should output its value. 1-8 is used.
exLoc	Defines the external memory location to store the information.
IntLoc	Defines the internal memory location for the network card.
Cmd	Defines the command/system variable to transmit/receive via the network. See table for definitions.
I_O	Defines if the command is inbound data or outbound data. 1=Inbound, 0=Outbound
"Var"	Optional argument to send a WT-BASIC variable via the network.

The map keyword allows the user to direct a given system variable to the network automatically. The system variables that are mapped to the network are updated at 10Hz.

### Map Command Definitions:

Inbound Data to WPI-135	Command#:	Outbound Data from WPI-135:	Command#:
Request Zero	0	Actvalue(Gross=0...ADC=13)	0-13
Request Tare	1	Motion	100
Request Print	2	Center of Zero	101
Request Accumulation	3	Actvalue#(Gross=0...ADC=13)	102
Set Tare	4	Over/Underload	103
WT-BASIC Variable	5	Current Unit(0=lb, 1=kg...)	104
Set Current Units	6	WT-BASIC Variable	105
Setpoint Bank	110-117	Setpoint Bank	110-117
Fmtprint(150=1...181=32)	150-181	Setpoint	200-263
Setpoint	200-263	Recall Memory	1000-9191
WT-BASIC Event#	400-591		
Store Memory	1000-9191		

**For more detailed information on network connections and programming please reference the WPI-135 NETWORK INSTALLATION GUIDE P/N 29762-0015.**

WT-BASIC example for 3 scale system transmitting gross weight only:

```
SUB SYSTEM_STARTUP
Map(1,1,0,0,0) 'map scale#1 gross weight
Map(1,2,2,1,0,0) 'map scale#2 gross weight
Map(1,3,4,2,0,0) 'map scale#3 gross weight
END SUB
```

### TIMER

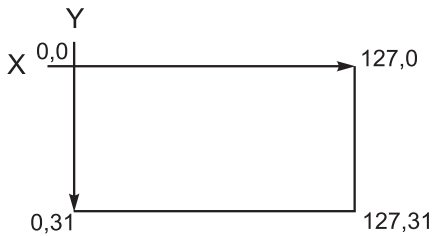
This command returns the number of seconds (in hundredths) since midnight of the current day.

### UNIXTIME

This command returns the number of seconds (in hundredths) since midnight January 1, 1970.

## Graphics

The WPI-135 has a display height of 32 dots and a width of 128 dots. Coordinate 0,0 is the upper leftmost dot.



## Structure Control

Use **REM** or **'** for describing operations, setup of the instrument, temporarily disabling a command, etc.

All forms of **IF.. commands** are used to make a decision regarding program flow based on the result of an expression.

Example of an expression:  
(A>B)

Examples of statements:  
PRINT "BIG NUMBER"  
or  
PRINT "LITTLE NUMBER"  
or  
C = (A\*B)+ d

Graphics can only be used when a display mode is chosen which does not contain a weight display. The upper left-hand corner of the display is (0,0). The lower right-hand corner is (127,31). The "X" references the horizontal coordinates (0-127). "Y" references the vertical coordinates (0-31).

### DOT(X,Y)

Draws a dot on the screen at coordinates X,Y.

### CIRCLE(X,Y,R)

Draws a circle on the screen starting at coordinates X,Y, using the value of R as the radius (in pixels).

### LINE(X1,Y1,X2,Y2)

Draws a line on the screen from coordinates X1, Y1 to X2, Y2.

### REM

Allows for comments to be placed in the body of a program.

### ' (apostrophe)

This also works in place of REM.

### IF. .THEN (singular)

SYNTAX: IF expression THEN statement

### IF. . THEN . .ELSE (singular)

SYNTAX: IF expression THEN statement ELSE statement

### IF. .THEN (block)

SYNTAX:

```
IF expression THEN
    statement
    statement
    statement
END IF
```

### IF. .THEN.. ELSE (block)

SYNTAX:

```
IF expression THEN
    statement
    statement
    statement
ELSE
    statement
    statement
    statement
END IF
```

### END IF

Used to stop a block of statements.

**WARNING**

*GOTOs and GOSUBs are not recommended*

*“Statement label” is any name or number used to identify the next “line of code” or statement to be executed.*

**ELSEIF**

This command allows absolute detection of a condition.

SYNTAX:

```
IF expression THEN
    statement
ELSEIF expression THEN
    statement
ELSEIF expression THEN
    statement
ELSE
    statement
END IF
```

**GOTO**

To branch out of the normal program sequence of instructions to a specified instruction identified by a “statement label”.

SYNTAX: GOTO *statement label*

**YOU ARE STRONGLY URGED NOT TO USE THIS COMMAND DUE TO COMPLICATIONS IT CAN CAUSE IF NOT USED CORRECTLY.**

*FOR. . NEXT loops only count up, not down.*

## **FOR. .TO. .NEXT**

To execute a series of instructions a specified number of times in a loop. Must also use "NEXT" command.

## **EXIT FOR**

Used to terminate a FOR. . NEXT loop upon detection of a certain condition prior to the loop expiring naturally.

SYNTAX:

```
FOR variable = X TO Y
    statement
IF expression THEN EXIT FOR
NEXT variable
```

## **WHILE. . WEND**

To execute a series of statements in a loop as long as a given condition is true.

## **EXIT WHILE**

Used to terminate a WHILE. . WEND loop upon detection of a certain condition prior to the loop expiring naturally.

SYNTAX:

```
WHILE expression
    statement
IF expression THEN EXIT WHILE
    statement
WEND
```

*It is better to use **CALL** instead of **GOSUB***

## **GOSUB . . . RETURN**

To branch to a subroutine and go back to the original subroutine. These work the same as CALL.

## **RETURN**

To return from a subroutine back to original subroutine.

## **SUB . . . END SUB**

Defines a subroutine procedure.

## **EXIT SUB**

Used to terminate a subroutine upon detection of a certain condition prior to the subroutine expiring naturally.

SYNTAX:

```
SUB subname
    statement(s)
IF expression THEN EXIT SUB
    statement(s)
END SUB
```

## **CALL**

Transfers control to a second subroutine procedure. After the second subroutine is done, control automatically transfers back to the next line after the CALL command in the previous subroutine.

SYNTAX: CALL PrintRoutine

## Display Control

*Pay close attention to the location of spaces used with keywords that have a message inside of quotation marks. A space usually precedes the first quotation mark.*

#0	Gross
#1	Net
#2	Tare
#3	Minimum
#4	Maximum
#5	Rate of Change
#6	Gross Total
#7	Net Total
#8	Count Total
#9	Transaction Total
#10	Count
#11	Value
#12	Piece Weight
#13	A to D Counts

### CLS

Clears the display of any WT-BASIC text. Not necessary before a DISPMODE command.

### REFRESH

Updates the displayed data while in a WT-BASIC program loop.

### DISPMODE(n) or DISPMODE=n

Sets the active display mode. The display modes may be viewed in accompanying software or in Appendix 1 of this manual. There are 43 displays to choose from. (n) is the number representing the display mode.

SYNTAX: DISPMODE(n)

### KEY x,"keyname"

Names a function key on the screen. (x; 1-5)

SYNTAX: KEY 1, "START"  
KEY 2, "STOP"  
KEY 3, "SET UP"  
KEY 4, "MORE"

### ACTVALUE

As a command ACTVALUE sets the displayed numeric data on the indicator to a different active value. See list at left.

SYNTAX: ACTVALUE(n) n=0 to 13

As a system variable ACTVALUE returns number (n) which represents the currently displayed active value per list at left.

SYNTAX: n=ACTVALUE n=0 to 13

### ANBASIS(CH#,SCL#,BASIS[,VAR\$])

**CH#** is the numeric value representing the analog output channel or board number in the system (CH#=1-8).

**SCL#** is the numeric value representing the system scale number this analog output channel is based on (SCL#=1-8).

**BASIS** is the active value in the system this analog output channel is based on (BASIS=0-13). If #11 is used then the optional parameter VAR\$ must be used

0=Gross	4=Max	8=Count Total	12=Piece Weight
1=Net	5=ROC	9=# Total Tans.	13=ADC
2=Tare Value	6=Gross Total	10=Count	
3=Min	7=Net Total	11=Variable	

**VAR\$** is a numeric value or numeric variable that the analog output is based on. This is implemented by one of the following methods in your WTbasic program.

```
ANBASIS(3,2,11,"numeric_variable_name")
```

Or

```
VAR$=" numeric_variable_name"  
ANBASIS(2,3,11,VAR$)
```

**SETANLOG(CH#,SCL#,MIN,MAX[,SPAN ADJ%,OFFSET ADJ%])**

**CH#** is the numeric value representing the analog output channel or board number in the system (CH#=1-8).

**SCL#** is the numeric value representing the system scale number this analog output channel is based on (SCL#=1-8).

**MIN** is the minimum value represented by the analog output channel (voltage or current). This normally represents empty scale on the control room display panel.

**MAX** is the maximum value represented by the analog output channel (voltage or current). This normally represents a full scale on the control room display panel.

**SPAN ADJ%,OFFSET ADJ%** are optional parameters to compensate for the loading characteristics of the device in the control room that will respond to the analog output signal (voltage or current). **OFFSET ADJ%** affects the minimum value sent to the control room. **SPAN ADJ%** affects the maximum value sent to the control room. These values are typically between 0% and 100%. It is recommended that these adjustments be made on site via the front panel configuration menu. However if done in your basic program remember to store the values and then recall them for use on power up.

*In the SHOWVAR syntax the precision value is the number of digits on the right side of the decimal point for your displayed numeric value. Remember to select a display mode that will allow you to see all the digits.*

*If you have multiple SHOWVARs, you must have an equal number of scales enabled. Display modes 34-43 must be used. Using Line# will replace a weight value line with the SHOWVAR value.*

## GRBASIS(*n*,[*varname*])

This command sets the basis for the bar graph representation on the display. (n = 0 to 13) Varname only applies when the graph basis is number 11, variable.

SYNTAX: GRBASIS(11,"VALUE")

OR

VAR\$="VALUE"

GRBASIS(11,VAR\$)

0 = Gross	4 = Max	8 = Count Total	12=Piece Weight
1 = Net	5 = Rate of Change	9 = # Total Trans.	13=ADC
2 = Tare	6 = Gross Total	10=Count	
3 = Min	7 = Net Total	11=Variable	

## SETBAR(*MinValue*,*MaxValue*)

Sets the values for the bar graph. The (MIN) is the minimum value where the graph starts. The (MAX) is the maximum value where the graph stops.

## SETCHECK(*Min*,*Under*,*Over*,*Max*)

Sets the values for the checkweigher graph. The (MIN) is the minimum value where the graph starts. The (MAX) is the maximum value where the graph stops. (UNDER) is the lowest acceptable value for the checkweigher graphic display. (OVER) is the highest acceptable value for the checkweigher graphic display.

## SHOWVAR

This lets you display a numeric value on the display where the weight value usually appears along with custom labels or legends. See note at left.

SYNTAX: SHOWVAR (varname\$,legend1\$,legend2\$,precision,[line#])

OR

SHOWVAR("VALUE", Legend1\$,Legend2\$,precision,[line#])

## SHOWSETP

Shows setpoints in the upper right of the display. Use only for troubleshooting. **This command should not be left in a customer's final program version.**

## MENU

SYNTAX:

MENU"key1","routine1", . . . . "key5","routine5"

This new keyword allows the user to create a menu system without creating it with a series of if/then statements. The keyword requires the softkey legend and the subroutine menu event which is queued for execution upon the keypress. Menu may be passed up to 10 softkey labels and sub routines. If a function is not defined for a given softkey, the default event handler, for that softkey will be queued, if it exists.

```

SUB SYSTEM_STARTUP
dispmode=16
menu "setup","setup_values","", "", "", "", "", "", "", "start","start"
END SUB

```

Brings up this display:



```

SUB start
cls
Print "  batching"
menu "stop","stop","", "", "", "", "", "", "", ""
END SUB

```

Brings up this display:



```

SUB stop
cls
Print " batch stopped"
menu "", "", "", "", "cont.", "continue","", "", "abort", "abort"
END SUB

```

Brings up this display:



```

SUB continue
call start
END SUB

SUB abort
call setup_values
END SUB

```



```

SUB setup_values
cls
print "enter new values"
print "or start batch"
menu "ingd1","ingd1","ingd2","ingd2","ingd3","ingd3","", "", "start","start"
END SUB

```

Brings up this display:



```

SUB ingd1
cls
print "enter ingred. value"
print "or start batch"
END SUB

```

```

SUB ingd2
cls
print "enter ingred. value"
print "or start batch"
END SUB

```

```

SUB ingd3
cls
print "enter ingred. value"
print "or start batch"
END SUB

```

All three of the previous subroutines will show this display:



## MENU1

Very similar to the standard MENU command used for nested menu structures using the soft keys. This command when used properly will avoid interaction of keys and allows you to provide alternate functionality of the labeled keys on the front panel.

### MENU1

"ESC","ENTER","ZERO","UNITS","TARE","PRINT","SELECT","NUMERIC","ENTRY","CLEAR"

#### SYNTAX:

MENU1 "sub routine name that works on the ESCAPE key press", "sub routine name that works on the ENTER key press",... "sub routine name that works on the NUMERIC key press - meaning the pressing of any numeric character", "sub routine name that works on the ENTRY key press - meaning the pressing of any non-numeric character", "sub routine name that works on the CLEAR key press"

Serial Port Hardware Control

EOM - End of Message character as set in the configuration menu.

Each serial port's transmit and receive buffer size is 512 characters. You may define the length of your string variable in a DIM statement. If the length is not defined, it defaults to 16 characters. Storage of strings in memory is limited to 16 characters per location. See Appendix 3 for a sample routine called GETCOM\$.

GETPORT

Returns the numeric value (x) that represents the current protocol of the port selected from the tables below.

SYNTAX: x=getport(port,y)

Pick 'y' from the list below (1 to 13)

- 1 baud rate
- 2 parity
- 3 databits
- 4 handshake
- 5 mode
- 6 EOM character
- 8 CTS (only used in getport)
- 9 transmit buffer free size (max 512)
- 10 receive buffer count (512 is full)
- 11 number of messages in the receive buffer
- 12 Serial Blocking
- 13 Download Blocking

X is the return value from the table below

Baud	Parity	Databits	Handshake	Mode	CTS
1=300	0=none	7=7	0=none	0=Basic Control	0=off
2=1200	1=odd	8=8	1=CTS	1=Keyboard	1=on
3=2400	2=even		2=XonXoff	2=Disable	
4=4800	3=set			3=Multidrop	
5=9600	4=clear			4=Computer mode	
6=19200					
7=38400					
8=56700					
9=115000					

SETPORT

SETPORT(port,y,z)

Pick y from the list below (1-15)  
Pick z from the corresponding table below.

- 1 baud rate
  - 2 parity
  - 3 databits
  - 4 handshake
  - 5 mode
  - 6 EOM character
  - 7 RTS (only used in setport)
  - 12 Serial Blocking
  - 13 Download Blocking
  - 15 Clears the receive buffer
- SYNTAX: setport(port,y,0)

Baud	Parity bits	Data	Handshake	Mode	CTS	RTS	Serial Blocking
1=300	0=none	7=7	0=none	0=Basic Control	0=off	0=off	0=on
2=1200	1=odd	8=8	1=CTS	1=Keyboard	1=on	1=on	1=off
3=2400	2=even		2=XonXoff	2=Disable			
4=4800	3=set			3=Multidrop			
5=9600	4=clear			4=Computer mode			
6=19200							
7=38400							
8=56700							
9=115000							

## System Functions

**WARNING:** Using **CONTOUT** may cause system performance problems. i.e.-execution speed may be slowed.

### GETCOM\$(portnum,<maxchars>)

GETCOM\$() reads one message from the specified Com-Port. The function reads from the first character in the input buffer to one of the following:

- a) End of Message Character (As specified in the Serial Port configuration Menu in the SimPoser)
- b) No more characters in the input buffer.
- c) Maximum characters reached (As specified in the optional GETCOM\$ parameter)
- d) Maximum string size reached

SYNTAX: X\$=GETCOM\$(portnum,<maxchars>)

portnum represents serial port 1, 2, 3 or 4

maxchars (optional) represents the maximum number of characters to read and assign to the string (X\$).

### CLEARCOM (Port)

Currently not available.

### CONTOUT(fmt,rate)

This command activates the continuous output of the chosen format. Destination port is selected from the format menu.

SYNTAX: CONTOUT(FMT,RATE)  
FMT = format print number  
RATE = update rate

Example: CONTOUT(1,2) This outputs format 1 twice per second.

### SETPTON(n)

This command will activate SETPOINT n.

### SETPTOFF(n)

This command will deactivate SETPOINT n.

### ISSETPT(x)

Tells you whether the setpoint is activated.

SYNTAX: Z = ISSETPT(x) x = 1 to 32

The variable Z is returned as a true, negative one (-1), or a false, zero (0) depending on the condition of the setpoint.

### TONE(x)

Turns the tone on.  $x = 1/\text{Frequency} \times \text{Offset}$

### TONEOFF

Turns the tone off.

### BEEP

Sounds the beeper in the instrument.

## GETITEM

Get item is used to recall configuration parameters to be used by BASIC. The GETITEM request has 3 parameters. The first parameter is a two letter code that is used by the downloader to configure the indicator. The second parameter is used to request a particular data item from that list. The third parameter is used to choose a particular scale used by only certain configuration parameters if they fetch multiscale data. The following shows the context for using GETITEM and the two letter configuration codes that are supported.

GETITEM("XX", n)

"XX" - Is the configuration parameter to get the value from (See list below for an explanation of the configuration parameters available).

Supports "PF", "SM", "LC", "LA", "PS", "UN", "KY", "VT", "MS", "GL", "DS"

n - Is the number of the configuration parameter to recall. The first configuration parameter is 1 and the second is 2 and so on. Any item that is numerical in nature is recalled, string parameters and undefined parameters return -2.

### PF

Print format port. PF, n returns the port number used for print format n.

### PS

Configures the power saving modes of the scale.

- 1) Power saving shutoff (-1 enable, 0 disable)
- 2) Power saving inactivity timer (integer entry 0 to 60 minutes)
- 3) Shutdown warning (-1 enable, 0 disable)

### UN

Configures the number of active scales and the unit of measures.

- 1) Number of active Scales (1-3 scales (Local only is 1))
- 2) Calibration units - Selectable 0-lbs to 6-Custom unit 2
- 3) Pounds enable (-1 enable, 0 disable)
- 4) Kilograms enable (-1 enable, 0 disable)
- 5) Grams Enable (-1 enable, 0 disable)
- 6) Ounces enable (-1 enable, 0 disable)
- 7) Pound-Ounces enable (-1 enable, 0 disable)
- 8) Custom unit 1 enable (-1 enable, 0 disable)
- 9) Custom unit 2 enable (-1 enable, 0 disable)
- 10) Custom unit 3 enable (-1 enable, 0 disable)
- 11) Custom unit 4 enable (-1 enable, 0 disable)

### KY

Key enable configuration.

- 1) SELECT key enable
- 2) UNITS key enable
- 3) PRINT key enable
- 4) TARE key enable
- 5) ZERO key enable

#0	Gross
#1	Net
#2	Tare
#3	Minimum
#4	Maximum
#5	Rate of Change
#6	Gross Total
#7	Net Total
#8	Count Total
#9	Transaction Total
#10	Count
#11	Value
#12	Piece Weight
#13	A to D Counts

## VT

Display value enable.

- 1) Gross enable (-1 enable, 0 disable)
- 2) Net enable (-1 enable, 0 disable)
- 3) Tare enable (-1 enable, 0 disable)
- 4) Min enable (-1 enable, 0 disable)
- 5) Max enable (-1 enable, 0 disable)
- 6) Rate of change enable (-1 enable, 0 disable)
- 7) Gross total enable (-1 enable, 0 disable)
- 8) Net total enable (-1 enable, 0 disable)
- 9) Count total enable (-1 enable, 0 disable)
- 10) Transaction count enable (-1 enable, 0 disable)
- 11) Count enable (-1 enable, 0 disable)
- 12) Variable enable (-1 enable, 0 disable)
- 13) Pc. Wt. Enable (-1 enable, 0 disable)
- 14) ADC Enable (-1 enable, 0 disable)

## MS

Miscellaneous configurations.

- 1) AC Excitation (0=DC, 1=360, 2=600, 3=1200)
- 2) Default print format (0-32)
- 3) Date format (0 MMDDYY, 1 DDMMYY, 2 YYMMDD)
- 4) Beeper volume ( 0 disabled, 1 low, 2 medium, 3 high)
- 5) Place holder
- 6) Place holder
- 7) Place holder
- 8) Place holder
- 9) Small font - enable lower case (0 disable, -1 enable)
- 10) Enable switching display mode with hidden key (0 disable, -1 enable)

## GL

Bar graph configuration.

- 1) Graph minimum (float - weight in cal. unit of measure)
- 2) Graph below (float - weight in cal. unit of measure)
- 3) Graph above (float - weight in cal. unit of measure)
- 4) Graph maximum (float - weight in cal. unit of measure)
- 5) Graph basis (0 Gross through 13 ADC value, see note at left)

## DS

Selected display mode.

- 1) Power-up display mode number (1-43 integer)

GETITEM("XX", n, sc)

"XX" - Is the configuration parameter to get the value from (See list below for an explanation of the configuration parameters available).

Supports "TS", "MC", "ZS", "FS", "RS", "SC", "RN", "TA"

n - Is the number of the configuration parameter to recall. The first parameter for these configurations are scale number. The parameter immediately following the scale number is 1 and the next is 2 and so on. Any item that is numerical in nature is recalled. String parameters and undefined parameters return -2.

sc - Is provided to choose a scale from which to recall the parameter from. If a scale number is not provided the Local scale (0) is assumed.

#### EXCEPTIONS:

GETITEM("PF",f) - Returns the port number to which the format number 'f' is configured to output to.

GETITEM("UN",10) Returns a true (-1) if more than one unit of measure is enabled, otherwise returns false (0).

#### TS

Key time out configuration for a **specific** scale.

- 1) Accumulator time out (Float in seconds)
- 2) Print time out (Float in seconds)
- 3) Zero time out (Float in seconds)
- 4) Tare time out (Float in seconds)

#### MC

Motion detection configuration for a **specific** scale.

- 1) Motion detection enable (-1 enable, 0 disable)  
( If motion detection is disabled Range and Delay are set to zero.  
Remainder of line is read, but not taken)
- 2) Range (float in divisions 0.1-20.0)
- 3) Delay (float in seconds 0.05-10.0)

#### ZS

Zero tracking configuration for a **specific** scale.

- 1) Zero tracking enable (-1 enable, 0 disable)  
( If zero tracking is disabled Range and Delay are set to zero.  
Remainder of line is read, but not taken)
- 2) Range (float in divisions 0.1-20.0)
- 3) Delay (float in seconds 0.05-10.0)

#### FS

Filtering configuration for a **specific** scale.

- 1) Unused
- 2) Samples to average (1 to 60)
- 3) Harmonizer enable (-1 enable, 0 disable)  
(Disabling the Harmonizer sets the constant value to 0)
- 4) Harmonizer constant (0 through 10)
- 5) Harmonize threshold (float weight in cal. unit of measure)

#### RS

Rate of change configuration for a **specific** scale.

- 1) Rate of change samples (0 through 60)
- 2) Rate of change multiplier (float)

## **SC**

Configures a single scale.

- 1) Scale Type (0 Analog, 1 Quartzzell)
- 2) Quartzzell port number
- 3) Capacity (Float- weight in cal. unit of measure)
- 4) Division (Float - weight in cal. unit of measure)
- 5) Update rate ( 0-0.5 Hz, 1-1 Hz, 2-2 Hz, 3-5 Hz,)
- 6) Zero range (0-100%)
- 7) Linearization setting
- 8) Accumulator return to zero (0 to 100%)
- 9) Print return to zero (0 to 100%)
- 10) Cal Wt (float - weight in cal. unit of measure)
- 11) Multi-interval setting

## **CT**

Configures counting settings.

- 1) Piece weight motion settings
- 2) Piece weight motion time

## **CU**

- 1=Custom unit 1 conversion factor
- 2=Custom unit 2 conversion factor
- 3=Custom unit 3 conversion factor
- 4=Custom unit 4 conversion factor

## **CF**

- 1=Custom Unit 1 Cal wt value
- 2=Custom Unit 1 Custom unit value
- 3=Custom Unit 2 Cal wt value
- 4=Custom Unit 2 Custom unit value
- 5=Custom Unit 3 Cal wt value
- 6=Custom Unit 3 Custom unit value
- 7=Custom Unit 4 Cal wt value
- 8=Custom Unit 4 Custom unit value

## **UNIT\$**

The command UNIT\$(0) returns 'lb' and UNIT\$(1) returns 'kg', etc. ( For LB-OZ unit of measure, OZ is returned due to all weight variables return ounces when the LB-OZ unit of measure is selected.)

## **CALDATA(x, [y])**

y = scalenum 1-8

Returns the following calibration information for the current scale, unless the scale is specified. This is for ISO purposes.

x=1 calfactor

x=2 calzero

x=3 Julian caltime

x=4 cal weight entered

x=5 cal weight on display after cal span

## **VERSION\$(x)**

Returns the version of the firmware and W-T Basic program.

x=1; Indicator type, SimPoser version, time, date of download, License #, customer name

x=2 ; not used

x=3; Firmware Part Number

x=4; Firmware Revision Letter

x=5; Serial number of the indicator

x=6; Serial number of SimPoser which downloaded current file

x=7; Company name of SimPoser license holder

x=8; Xilinx part number

x=9; Xilinx revision number

## **SETPWD(n,p\$)**

Sets the system password only.

The index n selects the password to set (1 - System password)

The string p\$ is the string containing the new password.

Syntax for setting the system password:

P\$ = "PASSWORD"

SETPWD(1,P\$)



## Timer/Event/Sleep Control

*There are four timer events available for use in the WPI-135. They are SYSTEM\_TIMER, SYSTEM\_TIMER2, SYSTEM\_TIMER3, and SYSTEM\_TIMER4. Use SETTIMER to regulate their frequency of occurrence.*

*EVENTCLR(0) will clear the entire event queue.*

### SHUTDOWN

This command causes the scale to disable power and shutdown. There are no parameters for this function.

### SETTIMER

This command causes the SYSTEM\_TIMER event to occur.

SYNTAX: SETTIMER(timernum,seconds)

Example:

SETTIMER(1,0.5) This will activate the event system\_timer every 0.5 second. With zero seconds, this disables the timer event specified. This is the default state of the Timer Events upon power up.

SETTIMER(1,0) This shuts the timer off.

### EVENTNUM

This command returns the variable (x) which identifies the event name. The variable (x) can then be used in the EVENTRDY and EVENTCLR commands.

SYNTAX: x = EVENTNUM(NAME\$)  
The variable (x) is the EVENTNUM.  
Name\$ is the event name.

### EVENTRDY(x)

This command detects the existence of an event name in the event queue.

SYNTAX: Z = EVENTRDY(x)

The variable Z returns true, negative one (-1), or false, zero (0).

The variable x represents the value of the event name from the command EVENTNUM.

### EVENTCLR(x)

This clears the oldest instance of EVENTNUM from the event queue.

SYNTAX: EVENTCLR(x)

The variable x represents the value of the event name from the command EVENTNUM.

### SLEEP(n)

Causes the program to halt processing for n number of seconds.

SYNTAX: SLEEP(10) (Causes processing to halt for 10 seconds)

### CALCSTAT

Calcstat is a function that calculates a number of statistical values based on a series of numbers.

SYNTAX: CALCSTAT(START,COUNT,DEST)

#### Parameters

**Start-** Start is the beginning memory location of a series of values to calculate the statistical analysis on.

**Count-** Count is the number of sequential memory locations from the previous stated START parameter that have values stored to calculate the statistical analysis on.

**Destination-** Destination is the starting memory location to record the results of the statistical analysis. There must be eight sequential memory locations available for the results of CALCSTAT to be recorded in. The results will be in the following order starting at the Destination memory location:

Average  
Minimum  
Maximum  
Average Deviation  
Standard Deviation  
Standard Variance  
Skewness  
Curtosis

### ABS(n)

Returns the absolute value of the expression.

### ATN(n)

Returns the arctangent of the expression.

### SQR(n)

Returns the square root of the expression.

### INT(n)

Returns the integer value of the expression.

### FIX(n)

Truncates the expression to a whole number.

### CINT(n)

Rounds numbers with fractional portions to the next whole number or integer.

### SGN(n)

Returns the sign of the expression. 1, 0, or -1

### SIN(n)

Calculates the trigonometric sine of the expression, in radians.

### COS(n)

Calculates the cosine of the range of the expression.

### TAN(n)

Calculates the trigonometric tangent of the expression, in radians.

**LOG(n)**

Calculates the natural logarithm of the expression.

**EXP(n)**

Returns e to the power of x.

**ROUND(x,y)**

Rounds X to the nearest y.

SYNTAX: Z=Round(X,Y)  
 With X=10.04 and Y=0.05  
 The value of Z is now set to 10.05

**RANDOM**

Generates a random decimal number between 0 and (n).

SYNTAX: x = RANDOM(n)

The variable x represents the generated decimal number.

The variable n represents the largest decimal number you want to use.

---

**String Functions**


---

**STR\$(n)**

Converts the value of an expression to a string.

SYNTAX: Z\$=STR\$(X)  
 With X= 5000

The string Z\$ is now set to "5000" not the numeric value of 5000

**MID\$(C\$,n,[x])**

This command copies part of an existing string (C\$) and makes a new string. The new string starts at the nth character of the original string (C\$) and continues for x number of characters.

SYNTAX: Z\$=MID\$(C\$,n,[x]) ( [ ]=optional )  
 With C\$ ="The WPI-135 Indicator"  
 and n = 5 and x = 6  
 MID\$ returns the new string Z\$ or "WPI-135".

**LEFT\$(C\$,x)**

This command copies the leftmost characters of an existing string (C\$) and makes a new string. The new string starts at the leftmost character of the original string and continues for x number of characters to the right.

SYNTAX: Z\$=LEFT\$(C\$,x)  
 With C\$ ="The WPI-135 Indicator"  
 and x = 3  
 LEFT\$ returns the new string Z\$ or "The".

**RIGHT\$(C\$,n)**

This command copies the rightmost characters of an existing string (C\$) and makes a new string. The new string starts at the rightmost character of the original string and continues for x number of characters to the left.

SYNTAX: Z\$=RIGHT\$(C\$,x)  
 With C\$ ="The WPI-135 Indicator"  
 and x = 9  
 RIGHT\$ returns the new string Z\$ or "Indicator".

*Converts the numeric value of X to a string as defined by width and precision parameters.*



### **CHR\$(n)**

Converts an ASCII code to its equivalent character. See Appendix 5.

SYNTAX: Z\$ = CHR\$(n)

### **FORMAT\$**

Format\$(x,width.prec) Returns x as string with a width and precision the same as in the print formats. See Format section in this manual for more information on width and precision.

SYNTAX: Z\$ = FORMAT\$(x,w.p)

### **LCASE\$(C\$)**

Converts the C\$ to lower case characters.

SYNTAX: Z\$ = LCASE\$(C\$)

### **UCASE\$(C\$)**

Converts the C\$ to upper case characters.

SYNTAX: Z\$ = UCASE\$(C\$)

### **HEX\$(n)**

Creates a new string that represents the hexadecimal value of the numeric argument.

SYNTAX: Z\$ = HEX\$(n)

### **LTRIM\$(C\$)**

Strips C\$ of any leading spaces.

SYNTAX: Z\$ = LTRIM\$(C\$)

### **RTRIM\$(C\$)**

Strips C\$ of any trailing spaces.

SYNTAX: Z\$ = RTRIM\$(C\$)

### **JULDATE\$**

x\$=JULDATE\$(y) Where y is a Julian date, this will return the date as the string x\$.

### **JULIAN**

y=JULIAN(x\$) Where x\$ is the date to convert, this will return the value y as the Julian date.

### **VAL(C\$)**

Returns the numerical value of string C\$.

SYNTAX: Y = VAL(C\$)

### **LEN(C\$)**

Returns the number of characters in C\$.

SYNTAX: Y = LEN(C\$)

### **ASC(C\$)**

Returns the numeric value that is the ASCII code for the first character of C\$.

SYNTAX: Y = ASC(C\$)

*Julian date is the number of days since some day in the ancient past. One reference point is Julian day 2,440,000 is May 23, 1968. This allows any date to be represented or stored as a value rather than a string.*

*Julian dates start at noon.*

*All 4 digits of the year must be used to ensure correct Julian values beyond the year 2000.*

**INSTR(C\$,D\$)**

Searches for the first occurrence of string D\$ in C\$, and returns the position.

SYNTAX: Z=INSTR(C\$,D\$)  
 C\$="WPI-135 INDICATOR"  
 D\$="N"  
 Z=INSTR(C\$,D\$)  
 The value of Z is now set to 9

**SPACE\$(n)**

SPACE\$(n) will return a string with the specified number of spaces.

**PLEN(C\$)**

PLEN returns a numeric value (X) which represents the width of a string (C\$) in dots for proportional fonts.

SYNTAX: X = PLEN(C\$)

**CHECKSUM(C\$,n)**

Calculates a checksum value for the given string (C\$). The checksum calculation type (n) is an integer value representing the type of checksum to calculate. The possible values for n are: 1 = 32 Bit Sum; 2 = 8 bit XOR; 3 = CCITT method CRC16; 4 = XMODEM method CRC16; and 5 = CRC-16; The value returned is 32 bits, but not all calculation methods use the whole resolution.

---

## Weighing Functions

---

**DOZERO**

This command zeroes the scale if no motion is detected.

**DOUNITS**

This command switches the display to the next valid unit of measure for all scales. (If other scales are not on the same unit of measure the current scale is incremented to the next unit of measure and the rest of the scales are forced to that same unit of measure)

**DOTARE**

This command tares the scale if no motion is detected.

**DOPRINT**

This command prints the selected default print format as previously configured if no motion is detected. The default print format is not printed if PRINT\_KEY subroutine appears in the program.

**DOACCUM**

This command requests an Accumulate event. The weight must be stable within the motion window parameters for the accumulation to occur.

**DOSELECT**

This command switches the display to the next valid active displayable value (gross, net, tare . . .).

**DOBASE**

This command switches to the next configured and enabled scale.

*The average weight used in these calculations is based upon the system variable GROSS weight and is affected by filtering settings from the configuration menu.*



## RESETMIN

RESETMIN command resets the value of the system variable MINIMUM to the current value on the scale platform.

## RESETMAX

RESETMAX command resets the value of the system variable MAXIMUM to the current value on the scale platform.

## AVGSTART

Starts averaging for in-motion systems.

## AVGSTOP

This returns the average weight (X) on the scale since the last AVGSTART command or for the last 256 weight conversions, whichever is shorter.

SYNTAX: X=AVGSTOP

## INMOTION(startIO,stopIO,startLocation,quantity,min,max)

This keyword configures the firmware level inmotion system. This is different from the AVGSTART and AVGSTOP keywords.

Explanation of parameters:

startIO - IO location (setpoint) for the input to start the weighment. Positive value indicates start on activate, negative indicates start on deactivate.

stopIO - IO location (setpoint) for the input to stop the weighment. Positive value indicates stop on activate, negative indicates stop on deactivate.

startLocation - Beginning Store/Recall location to store values from in motion weighing.

quantity - Number of values to store beyond startLocation before wrapping around back to startLocation.

startLocation + quantity - New insertion pointer for store/recalls

min - Minimum (inclusive) acceptable gross weight.

max - Maximum (inclusive) acceptable gross weight.

The Store/Recall value at recall(startLocation+quantity) holds the value

stopIO+1 = Accept Setpoint

stopIO+2 = Reject Setpoint

stopIO+3 = Over Setpoint

stopIO+4 = Under Setpoint

Note: These setpoints must be configured for Basic control Activate and Deactivate.

Example:

```
SUB SYSTEM_STARTUP
  dispmode=1
  store(0,0,1023)
  minval=0
  maxval=50
  inmotion(3,-4,0,1000,minval,maxval)
END SUB
```

```

SUB SETPT4_DEACT
  wt=RECALL(x)
  fmtprint(1)
  x=x+1
  if x>1000 then x=0
END SUB

SUB F1_KEY
  inmotion(3,-4,0,1000,minval,maxval)
END SUB

SUB F2_KEY
  input "Enter Min: ",minval
END SUB

SUB F3_KEY
  input "Enter Max: ",maxval
END SUB

```

## GETCONV

Gets the conversion factor of the currently displayed unit measure.

SYNTAX: X = GETCONV

## PCWTSAMP(SAMPLEsize)

This keyword takes the sample size and tries to calculate a piece weight based on the weight on the scale.

## PCWTZERO

This keyword performs a finer zero operation based on the same algorithm the sampling routine uses.

## CAPTURE

The capture command allows storage of weight readings into RAM for analysis at varying rates. There are multiple commands built into the capture keyword as outlined below.

0. Pauses the capture.
1. Starts the data capture.
2. Configures the Data capture.
3. Returns the number of data points the capture command has stored.

SYNTAX:

```

CAPTURE(0) - stop capture
CAPTURE(1) - start capture
CAPTURE(2, xstart, xdatapt, xrate, xch, xs) - configure capture

```

xstart - starting memory location (store/recall number)

xdatapt - number of data points to take

xrate - sampling rate (max 60Hz)

xch - scale number to take readings from

(On the 830 only the current scale "curscale" can be captured from)

xs - setpoint number that can control the activation or deactivation of the capture feature.

Enter "0" to disable the need for setpoint activation.

CAPTURE(3, xstatus) - store current status.

The status stores the store/recall address of the next data point. To calculate the number of samples taken, subtract the status character from the start address (xstart).

xstatus - the store/recall address to place the status.

The following application outlines how to use the capture command with a single scale.

```
SUB SYSTEM_STARTUP
dispmode=17
key 1,"START"
key 2,"SETUP"
key 3,"OFF"
key 4,"STOP"
key 5,"RE-STRT"
xstart=0          'default starting memory location
xdapt=1600        'take 1600 data points
xrate=60          'take samples at 60Hz
xch=1             'take readings from scale#1
xs=32             'setpoint 32 may control the activation or 'deactivation of the capture feature, but isn't required

END SUB
```

'start the data capture

```
SUB F1_KEY
capture(2,xstart,xdapt,xrate,xch,xs)
setpton(32)
capture(1)
settimer(2,1)
END SUB
```

'change the default capture settings

```
SUB F2_KEY
input "Start:",xstart
if lastkey=27 then exit sub
input "DataPT:",xdapt
if lastkey=27 then exit sub
input "RATE: ",xrate
if lastkey=27 then exit sub
input "Channel:",xch
if lastkey=27 then exit sub
input "SETPT:",xs
if lastkey=27 then exit sub
capture(2,xstart,xdapt,xrate,xch,xs)
'capture(2,0,1600,60,1,32)
END SUB
```

'turn the data capture setpoint off, essentially you paused the data

```
'capture
SUB F3_KEY
setptoff(32)
settimer(2,0)
END SUB
```



'turn the data capture off this resets the data capture

SUB F4\_KEY

setptoff(32)

capture(0)

settimer(2,0)

END SUB

'clear ram to start over again

SUB SELECT\_KEY

store(0,0,8000) 'store 8000 zeros starting at address 0

END SUB

'show the data capture status to the display

sub system\_TIMER2

capture(3,8100) 'tell capture where to store the index

print RECALL(8100) 'get the index

end sub

'restart the data capture again

sub F5\_key

setpton(32) 're-enable the capture

settimer(2,1)

end sub

---

## Memory

---

There are 8192 numeric storage locations and 4096 sixteen character string storage locations available in nonvolatile memory. These memory locations are reusable and will remain through a power loss. Four commands are available for accessing these memory locations:

Locations start at 0      numerics go from 0 to 8191  
                                 strings go from 0 to 4095

### **STORE(loc, #value, hiloc)**

Stores #value in the loc numeric memory location. If a number for hiloc is included, a fill is performed thereby storing the value in all locations from loc to loc+hiloc.

### **STORE\$(loc, "string value", hiloc)**

Stores "string value" in the loc string memory location, 16 characters maximum per location. If a number for hiloc is included, a fill is performed, thereby storing the string value in all locations from loc to loc+hiloc.

### **RECALL(loc)**

Recalls the loc numeric memory location and assigns the value stored there to your variable (X).

Syntax:      X = RECALL(loc)

### **RECALL\$(loc)**

Recalls the loc string memory location and assigns the value stored there to your variable (C\$).

SYNTAX:    C\$=RECALL\$(loc)

### **FIND(VAR,x,y)**

Searches numeric memory slots between x and y for a variable and returns the memory location. Returns a (-1) if not found.

SYNTAX:    Z = FIND (VAR,x,y)

Optional Argument 'type':

SYNTAX:      Z = FIND (VAR,x,y,type)

type - provides optional search patterns. If type is not provided, the default is to find an equal value (type = 0) The search options are as follows; type =

- 0 - Find an equivalent to the given value (=)
- 1 - Find the first value that is less than the given value(<)
- 2 - Find the first value that is less than or equal to the given value (<=)
- 3 - Find the first value that is greater than the given value (>)
- 4 - Find the first value that is greater than or equal to the given value (>=)
- 5 - Find the first largest value within the given range (VAR is ignored) (max)
- 6 - Find the first smallest value within the given range (VAR is ignored) (min)
- 7 - Find the pass through value within the given range (pass through)

Pass through is defined as the index of the first value from (x+1) to (y) that is greater than the given value where VAR > value(x), or the index of the first value from (x+1) to (y) that is less than the given value where VAR < value(x) If VAR = value(x) then x = x + 1.

*If x > y the keyword will look backwards through the specified memory locations.*

*If x > y the keyword will look backwards through the specified memory locations.*

## **FINDSTR(C\$,x,y)**

Searches numeric memory slots between x and y for an equivalent string and returns the memory location. Returns a (-1) if not found.

SYNTAX: Z = FINDSTR (C\$,x,y)

Optional Arguments 'type' and 'nocase':

SYNTAX: Z = FINDSTR (VAR,x,y,nocase,type)

nocase - if set to true (nonzero) the strings are compared after being converted to all upper case letters. The default is False (0).

type - provides optional search patterns. If type is not provided, the default is to find an equivalent string (type = 0) The search options are as follows;  
type =

0 - Find an equivalent string to the given string (=)

1 - Find the first string that is less than the given string(<)

2 - Find the first string that is less than or equal to the given string (<=)

3 - Find the first string that is greater than the given string (>)

4 - Find the first string that is greater than or equal to the given string (>=)

5 - Find the first largest string within the given range (VAR is ignored) (max)

6 - Find the first smallest string within the given range (VAR is ignored) (min)

7 - Find the pass through string within the given range (pass through)

Pass through is defined as the index of the first string from (x+1) to (y) that is greater than the given string where VAR > value(x), or the index of the first string from (x+1) to (y) that is less than the given string where VAR < value(x) If C\$ = string(x) then x = x + 1.

8 - Find the string that contains the given string (Contains)

9 - Find the string that is contained by the given string (is contained by)

10 - Find the first string that is less than the given string(<). This selection differs from #1 in that the search criteria is determined by the length of the string, then by the normal string compare conventions.

11 - Find the first string that is less than or equal to the given string (<=). This selection differs from #2 in that the search criteria is determined by the length of the string, then by the normal string compare conventions.

12 -Find the first string that is greater than the given string (>). This selection differs from #3 in that the search criteria is determined by the length of the string, then by the normal string compare conventions.

13 -Find the first string that is greater than or equal to the given string (>=).This selection differs from #4 in that the search criteria is determined by the length of the string, then by the normal string compare conventions.

## Appendix 3: Subroutine Examples

### Sample : GETCOM\$

This is an example of using GETCOM2 for data storage:

```
SUB SYSTEM
dim X$*32
dim SPACE$*32
ROOM$="<32-Blanks>"
DISPMODE(17)
KEY 1,"RECALL"
END SUB
```

This section reserves room in memory by dimensioning the variables X\$ and SPACE\$ each to 32 characters. Display mode is set to 17 (See *Appendix 1*). KEY labels softkey 1 as RECALL.

```
SUB COM2_MESSAGE
X$ = GETCOM$(2)
X$ = X$+LEFT$(SPACE$(32-LEN(X$))
A$=LEFT$(X$,16)
B$=RIGHT$(X$,16)
STORE$(1,A$)
STORE$(2,B$)
END SUB
```

This section retrieves a message from serial port #2 and then makes it 32 characters long and splits this into two 16-character string variables (A\$, B\$), and stores them in locations 1 and 2.

```
SUB F1_KEY
C$=RECALL$(1)
D$=RECALL$(2)
X$=C$+D$
PRINT X$
END SUB
```

This section recalls the two 16-character strings from memory, combines them and prints them to the display.

### Sample : SHOWVAR

```
SUB SYSTEM_STARTUP
dim WT
dim VAR$
dim LEGEND1$
dim LEGEND2$
dim PREC
VAR$="WT"
LEGEND1$="kg"
LEGEND2$="brutos"
PREC=2
SHOWVAR(VAR$,LEGEND1$,LEGEND2$,PREC)
SETTIMER(2,0.5)
ACTVALUE(11)
END SUB
```

This subroutine reserves memory by dimensioning the variables of the program. It assigns strings to the variables LEGEND1\$ and LEGEND2\$ which are used in the SHOWVAR command.

SETTIMER causes the SUB SYSTEM\_TIMER2 event to occur every ½ second. ACTVALUE allows the variable WT to be displayed where the weight normally is displayed on the WPI-135 display.

```
SUB SYSTEM_TIMER2
WT=GROSS
END SUB
```

The SUB SYSTEM\_TIMER2 subroutine assigns the system variable GROSS into the variable WT.

## Sample: STORE and RECALL

```
SUB SYSTEM_STARTUP
MUSTDIM
dim WORD$
dim LOC
dim PASSWORD
dim GUESS
dim Y
dim VNUM
dim X
WORD$=""
PASSWORD=111
DISPMODE(17)
KEY 1, "NEXT"
KEY 2, "STORE"
KEY 4, "CLRMEM"
KEY 5, "PREV"
LOC=RECALL(1000)
END SUB

SUB F2_KEY
LOC=LOC+1
input "NAME",WORD$
STORE$(LOC,WORD$)
INPUT "PHONE#",VNUM
STORE(LOC,VNUM)
STORE(1000,LOC)
END SUB

SUB F1_KEY
X=X+1
IF X>LOC THEN X=LOC
WORD$ = recall$ (X)
VNUM = RECALL (X)
PRINT WORD$," ";VNUM
END SUB

SUB F4_KEY
GUESS=0
INPUT "PASSWORD?",GUESS
IF PASSWORD=GUESS THEN
FOR Y=0 TO 1024
STORE(Y,0)
STORE$(Y,"")
NEXT
LOC=0
END IF
END SUB

SUB F5_KEY
X=X-1
IF X<0 THEN X=0
WORD$ = recall$ (X)
VNUM = RECALL (X)
PRINT WORD$," ";VNUM
END SUB
```

This program gives you an example of storage and retrieval of names and numbers from memory.

## Sample Setpoint (Cutoff) Application

```
SUB SYSTEM_STARTUP
mustdim
dim cutoff1
dim cutoff2
dim disp
cutoff1=RECALL(1023)
cutoff2=RECALL(1022)
END SUB
```

```
SUB F1_KEY
disp=dispmode
dispmode=6
input "Enter Cutoff#1",cutoff1
if lastkey=27 then
    dispmode=disp
    exit sub
end if
input "Enter Cutoff#2",cutoff2
if lastkey=27 then
    dispmode=disp
    exit sub
end if
dispmode=disp
store(1023,cutoff1)
store(1022,cutoff2)
END SUB
```

```
'print screen of setpoints
'remote zero and remote tare examples
SUB SETPT1_ACT
dozero
END SUB
```

```
SUB SETPT2_ACT
dotare
END SUB
```

## Sample Continuous Output

```
'old way for continuous output
format #1
\x02 G {SEND$} LB\r\n

SUB SYSTEM_STARTUP
DISPMODE=10
ZERO$="000000"
SETTIMER(1,0.5)
END SUB

SUB SYSTEM_TIMER
GROSS$=STR$(ABS(GROSS))
X=6-LEN(GROSS$)
TEMP$=LEFT$(ZERO$,X)+GROSS$
IF GROSS<0 THEN
    SEND$="-"+TEMP$
ELSE
    SEND$=""+TEMP$
END IF
FMTPRINT(1)
END SUB

'new way for continuous output
'max is 10 times a second
SUB SYSTEM_STARTUP
contout(1,2)          'format #1 output twice a second
END SUB
```

## Sample Enquire via Serial Port

```
'do a print screen of format 1 and serial port??
'when a enquire EOM is received
'print format#1 will be sent
SUB COM2_MESSAGE
fmtprint(1)
END SUB
```

## Appendix 4: Error Messages

	Message	Meaning
<b>When Saving</b>	Drive not Ready	No disk in drive
	Disk Write Protected	Write protect tab on diskette is in place.
<b>Edit Menu</b>	Invalid Line Number	A line number that doesn't exist. (Goto Line# in Edit Menu)
	Search Item Not in this Program!	You tried to find something that wasn't there.
<b>Downloading</b>	Begin Transfer.....Bad Block	Cabling problem has occurred, intermittent connection.
<b>Other</b>	Subscript Out of Range	Restart WPI-135 SimPoser.
	Out of Memory	1. You have too many applications running in Windows®. 2. Your WPI-135 SimPoser configuration file is too large.
<b>Non-Mouse Users:</b>	Permission Denied	Do a FILE, SAVE AS c:\WPI-135 SimPoser\dat then a FILE, SAVE AS A:
	Disk not ready	Make sure disk is inserted properly.
	ALT-F6	Gets you back to the main screen if you were in either SETPOINTS or CONFIGURE.
<b>WT-BASIC Errors</b>	syntax error	A keyword or command was spelled wrong or used improperly.
	unbalanced paren	A parenthesis is missing.
	no expression present	Missing part of an expression.
	equals sign expected	Missing a space before a quotation mark , a system variable is mistakenly used as a regular variable, or variable name is too long.
	not a variable	Trying to assign a value to a command statement.
	Label table full	May have too many subroutines or not enough memory. Try adding the extended memory option.



duplicate label	You've duplicated an event name.
undefined label	Statement label in a GOTO is needs to be defined.
THEN expected	Part of IF THEN statement is missing.
<b>Message</b>	<b>Meaning</b>
TO expected	TO missing in a FOR. . .NEXT loop.
too many nested FORs	To many levels of FOR. . .NEXT loops.
NEXT without FOR	FOR. . .NEXT loop missing the FOR.
too many nested GOSUBs	To many levels of GOSUB.
RETURN without GOSUB	GOSUB. . .RETURN loop missing the GOSUB.
double quotes needed	Quotation marks missing.
String Expected	A string variable has no string assigned to it or a \$ is missing.
Variable Name Too Long	Variable name has too many characters.
Var Not Defined (DIM)	Variable name is probably misspelled or not dimensioned.
too many nested WHILEs	To many levels of WHILE statements.
WEND without WHILE	WHILE. . .WEND without a WHILE.
Division by Zero	You cannot divide by a value of zero.

# Appendix 5: ASCII Chart

Code #	Cont. Char.	Print Char.	Hex	Code #	Cont. Char.	Print Char.	Hex	Code #	Cont. Char.	Print Char.	Hex	Code #	Cont. Char.	Print Char.	Hex	Code #	Cont. Char.	Print Char.	Hex
0	NUL		00	045	-	-	2D	090	Z	Z	5A	0128	NA	Ç		0173	NA	ı	
01	SOH	☺	01	046	.	.	2E	091	[	[	5B	0129	NA	ü		0174	NA	«	
02	STX	☹	02	047	/	/	2F	092	\	\	5C	0130	NA	é		0175	NA	»	
03	ETX	♥	03	048	0	0	30	093	]	]	5D	0131	NA	â		0176	NA	☐	
04	EOT	♦	04	049	1	1	31	094	^	^	5E	0132	NA	ä		0177	NA	☐	
05	ENG	♣	05	050	2	2	32	095	_	_	5F	0133	NA	à		0178	NA	☐	
06	ACK	♠	06	051	3	3	33	096	`	`	60	0134	NA	á		0179	NA		
07	BEL		07	052	4	4	34	097	a	a	61	0135	NA	ç		0180	NA	†	
08	BS		08	053	5	5	35	098	b	b	62	0136	NA	ê		0181	NA	‡	
09	HT		09	054	6	6	36	099	c	c	63	0137	NA	ë		0182	NA	‖	
010	LF	LF	0A	055	7	7	37	0100	d	d	64	0138	NA	è		0183	NA	⌈	
011	VT	♂	0B	056	8	8	38	0101	e	e	65	0139	NA	í		0184	NA	⌋	
012	FF	FF	0C	057	9	9	39	0102	f	f	66	0140	NA	î		0185	NA	‖	
013	CR	CR	0D	058	:	:	3A	0103	g	g	67	0141	NA	ï		0186	NA		
014	S0	🎵	0E	059	;	;	3B	0104	h	h	68	0142	NA	Ä		0187	NA	⌋	
015	S1	⚙	0F	060	<	<	3C	0105	i	i	69	0143	NA	Å		0188	NA	⌋	
016	DLE	4	10	061	=	=	3D	0106	j	j	6A	0144	NA	É		0189	NA	⌋	
017	DC1	3	11	062	>	>	3E	0107	k	k	6B	0145	NA	æ		0190	NA	⌋	
018	DC2	ø	12	063	?	?	3F	0108	l	l	6C	0146	NA	Æ		0191	NA	⌋	
019	DC3	Ø	13	064	@	@	40	0109	m	m	6D	0147	NA	ô		0192	NA	⌋	
020	DC4	ß	14	065	A	A	41	0110	n	n	6E	0148	NA	ö		0193	NA	⌋	
021	NAK	§	15	066	B	B	42	0111	o	o	6F	0149	NA	ò		0194	NA	⌋	
022	SYN		16	067	C	C	43	0112	p	p	70	0150	NA	û		0195	NA	⌋	
023	ETB	—	17	068	D	D	44	0113	q	q	71	0151	NA	ù		0196	NA	—	
024	CAN	↑	18	069	E	E	45	0114	r	r	72	0152	NA	ÿ		0197	NA	†	
025	EM	↓	19	070	F	F	46	0115	s	s	73	0153	NA	ÿ		0198	NA	‡	
026	SUB	→	1A	071	G	G	47	0116	t	t	74	0154	NA	ÿ		0199	NA	‖	
027	ESC	←	1B	072	H	H	48	0117	u	u	75	0155	NA	ç		0200	NA	ℓ	
028	FS	—	1C	073	I	I	49	0118	v	v	76	0156	NA	£		0201	NA	℥	
029	GS	—	1D	074	J	J	4A	0119	w	w	77	0157	NA	¥		0202	NA	⌚	
030	RS	5	1E	075	K	K	4B	0120	x	x	78	0158	NA	ℙ		0203	NA	⌚	
031	US	6	1F	076	L	L	4C	0121	y	y	79	0159	NA	ƒ		0204	NA	⌚	
032	SP		20	077	M	M	4D	0122	z	z	7A	0160	NA	℥		0205	NA	=	
033	!	!	21	078	N	N	4E	0123	{	{	7B	0161	NA	ı		0206	NA	⌚	
034	"	"	22	079	O	O	4F	0124			7C	0162	NA	ó		0207	NA	⌚	
035	#	#	23	080	P	P	50	0125	}	}	7D	0163	NA	ú		0208	NA	⌚	
036	\$	\$	24	081	Q	Q	51	0126	~	~	7E	0164	NA	ñ		0209	NA	⌚	
037	%	%	25	082	R	R	52	0127	DEL	☐	7F	0165	NA	Ñ		0210	NA	⌚	
038	&	&	26	083	S	S	53					0166	NA	ª		0211	NA	ℓ	
039	'	'	27	084	T	T	54					0167	NA	º		0212	NA	ℓ	
040	(	(	28	085	U	U	55					0168	NA	¿		0213	NA	℥	
041	)	)	29	086	V	V	56					0169	NA	┐		0214	NA	℥	
042	*	*	2A	087	W	W	57					0170	NA	┐		0215	NA	⌚	
043	+	+	2B	088	X	X	58					0171	NA	½		0216	NA	⌚	
044	,	,	2C	089	Y	Y	59					0172	NA	¼		0217	NA	┐	

## Appendix 6: Alphabetical Listing of WT-BASIC Commands

' .....	66	CALDATA .....	80	DOPRINT .....	85
- .....	59	CALL .....	68	DOSELECT .....	85
* .....	59	CAPACITY .....	56	DOT .....	66
/ .....	59	CAPTURE .....	87	DOTARE .....	85
( ) .....	59	CHECKSUM .....	85	DOUNITS .....	85
\ .....	60	CHR\$ .....	84	DOZERO .....	85
+ .....	60	CINT .....	82	ELSE .....	66
- .....	60	CIRCLE .....	66	ELSEIF .....	67
= .....	60	CLEARCOM .....	75	END IF .....	66
> .....	60	CLEARERR .....	57	END SUB .....	67
< .....	60	CLEAR_KEY .....	52	ENTER_KEY .....	52
<= .....	60	CLS .....	69	ENTRY_KEY .....	52
>= .....	60	COM1_MESSAGE .....	52	EQV .....	59
< > .....	60	COM2_MESSAGE .....	52	ERR .....	57
> < .....	60	COM3_MESSAGE .....	52	ESC_KEY .....	52
^ .....	59	COM4_MESSAGE .....	52	EVENTCLR .....	81
ABS .....	82	CONTOUT .....	75	EVENTNUM .....	81
ACCUM_ABORT .....	52	COS .....	82	EVENTRDY .....	81
ACCUM_OPER .....	52	COUNT .....	56	EXIT FOR .....	68
ACTVALUE .....	69	COUNTTOT .....	56	EXIT SUB .....	68
ADCCNTS .....	56	CURSCALE .....	56	EXIT WHILE .....	68
ANBASIS .....	69	CURUNIT .....	56	EXP .....	83
AND .....	59	CURUNIT\$ .....	56	Fx_KEY .....	52
ASC .....	84	CZERO .....	56	F6- F10 .....	52
ASK(MSG\$) .....	63	DATE\$ .....	56	FIND .....	90
ATN .....	82	DIM .....	55	FINDSTR .....	91
AVGSTART .....	86	DISPLAY .....	57	FIX .....	82
AVGSTOP .....	86	DISPMODE .....	69	FMTPRINT .....	61
BASE_OPER .....	54	DIVISION .....	58	FOR . . TO . . NEXT .....	68
BEEP .....	75	DOACCUM .....	85	FORMAT\$ .....	84
CALCSTAT .....	82	DOBASE .....	85	GETCOM\$ .....	75

GETCONV ..... 87  
 GETNET ..... 64  
 GETITEM ..... 76  
 GETPORT ..... 74  
 GOSUB . . RETURN ..... 68  
 GOTO ..... 67  
 GRBASIS ..... 71  
 GROSS ..... 57  
 GROSSTOT ..... 57  
 HEX\$ ..... 84  
 IF . . THEN. . ELSE ..... 66  
 IMP ..... 59  
 INKEY\$ ..... 62  
 INMOTION ..... 86  
 INPUT ..... 61  
 INPUTOPT ..... 61  
 INSTR ..... 85  
 INT ..... 82  
 ISSETPT ..... 75  
 JULDATE\$ ..... 84  
 JULIAN ..... 84  
 KEY X, "keyname" ..... 69  
 KEYHIT ..... 62  
 LASTKEY ..... 62  
 LASTKEY1 ..... 62  
 LCASE\$ ..... 84  
 LEFT\$ ..... 83  
 LEN ..... 84  
 LINE ..... 66  
 LOG ..... 83  
 LTRIM\$ ..... 84  
 m ..... 60  
 MAP ..... 65

MAXPEAK ..... 57  
 MENU ..... 71  
 MENU ..... 73  
 MINPEAK ..... 57  
 MID\$ ..... 83  
 MOD ..... 60  
 MOTION ..... 57  
 MUSTDIM ..... 55  
 NET ..... 58  
 NETERR ..... 64  
 NETTOT ..... 58  
 NETWORK\_ABORT ..... 54  
 NETWORK\_OPER ..... 54  
 NETWORK\_STATUS ... 54  
 NEXT ..... 68  
 NUMERIC\_KEY ..... 53  
 NOT ..... 59  
 OR ..... 59  
 OVERLD ..... 58  
 PCWT ..... 58  
 PCWTSAMP ..... 87  
 PCWTZERO ..... 87  
 PLEN ..... 85  
 PRINT ..... 61  
 PRINT #x ..... 61  
 PRINT\_ABORT ..... 53  
 PRINT\_KEY ..... 53  
 PRINT\_OPER ..... 53  
 RANDOM ..... 83  
 RAWGROSS ..... 58  
 RAWNET ..... 58  
 RAWTARE ..... 58  
 RECALL ..... 90

RECALL\$ ..... 90  
 REFRESH ..... 69  
 REM ..... 66  
 RESETMAX ..... 86  
 RESETMIN ..... 86  
 RETURN ..... 68  
 RIGHT\$ ..... 83  
 ROC ..... 58  
 ROUND ..... 83  
 RTRIM\$ ..... 84  
 SELECT\_OPER ..... 53  
 SELECT\_KEY ..... 53  
 SETANLOG ..... 69  
 SETBAR ..... 71  
 SETCHECK ..... 71  
 SETNET ..... 64  
 SETPORT ..... 74  
 SETPTOFF ..... 75  
 SETPTON ..... 75  
 SETPT\_ACT ..... 53  
 SETPT\_DEACT ..... 53  
 SETPWD ..... 80  
 SETTIMER ..... 81  
 SGN ..... 82  
 SHOWSETP ..... 71  
 SHOWVAR ..... 71  
 SHUTDOWN ..... 81  
 SIN ..... 82  
 SLEEP ..... 81  
 SPACE\$ ..... 85  
 SQR ..... 82  
 STORE ..... 90  
 STORE\$ ..... 90

STR\$ .....	83
SUB . . END SUB .....	68
SYSTEM_ERROR .....	53
SYSTEM_SETUP .....	53
SYSTEM_STARTUP ....	53
SYSTEM_TIMER .....	53
SYSTEM_TIMER2 .....	54
SYSTEM_TIMER3 .....	54
SYSTEM_TIMER4 .....	54
TAN .....	82
TARE .....	58
TARE_ABORT .....	54

TARE_KEY .....	54
TARE_OPER .....	54
THEN .....	66
TIME\$ .....	58
TIMER .....	65
TO .....	68
TONE .....	75
TONEOFF .....	75
TRANSTOT .....	58
UCASE\$ .....	84
UNDERLD .....	58
UNIT\$ .....	80

UNITS_KEY .....	54
UNITS_OPER .....	54
UNIXTIME .....	65
VAL .....	84
VERSION\$ .....	80
WEND .....	68
WHILE . . WEND .....	68
XOR .....	60
ZERO_ABORT .....	54
ZERO_KEY .....	54
ZERO_OPER .....	55

**Weigh-Tronix**

1000 Armstrong Dr.  
Fairmont, MN 56031 USA  
Telephone: 507-238-4461  
Facsimile: 507-238-4195  
e-mail: [industrial@weigh-tronix.com](mailto:industrial@weigh-tronix.com)  
[www.wtxweb.com](http://www.wtxweb.com)

**Weigh-Tronix Canada, ULC**

217 Brunswick Blvd.  
Pointe Claire, QC H9R 4R7 Canada  
Telephone: 514-695-0380  
Facsimile: 514-695-6820

**WEIGH-TRONIX**

Weighing Products & Systems