EAN

04

AVERY WEIGH-TRONIX

Engineering Application Note

WTCommScl ActiveX Control

v3.0

ActiveX Control Documentation

This document is a reformat and update of the *WTCommScl.OCX ActiveX Control Documentation and Features* document (P/N: 8421-15950-04) Rev. B, dated 20-MAR-98. The update occurs as the result of significant software enhancements made to the OCX control beginning with version **(v3.0).** New functionality added will be highlighted in **blue.**

This new version is recommended for new designs only. Binary-compatibility with the previous (v2.0) has been intentionally broken. If you do not need the capabilities added in (v3.0) or above, you may continue to use the version (v2.0) control and this document for your legacy applications. If you choose to replace the legacy (v2.0) ActiveX control currently in your application with the new (v3.0) control in a .NET environment, please see Appendix-A for further information.

A summary of new features added are listed in the **New Features** section of **Appendix-A** at the end of this document.

DOC REV: 15-OCT-12

Introduction

This document describes the **WTCommScl** ActiveX control from Avery Weigh-Tronix. This software component is a true 32-bit in-process OCX control (.dll) which can be loaded into any ActiveX compliant development environment.

WTCommScl is a special purpose control which is used to handle all serial communications between a host computer and any Avery Weigh-Tronix scale that supports the NCI 7010 Serial Protocol as defined in the Weigh-Tronix Serial Communications Protocol (SCP-11).

Some scales supported include: the NCI Model 7010 Bench Scale, the Salter Brecknell Model 335 Postal Scale and the Salter Brecknell Model PS25 Postal/Shipping Scale.

By setting just a few properties in the control, a developer can have instant access to weight and status information from the scale.

Hardware/Software Requirements

- Personal computer running Windows XP (32-bit or 64-bit) or Windows 7 (32-bit or 64-bit). See installation document special instructions when installing on 64-bit systems.
- One available RS-232 Serial COM Port, *or*, one available USB Port if using a virtual communications port driver (VCP).
- A serial cable (may be proprietary for some scales), for legacy installations using traditional serial RS-232 ports, *or* a USB cable for scales that support communications using a virtual communications port/driver.
- Microsoft Visual Basic 5.0 or (6.0) Programming System or other equivalent ActiveX compliant development environment in order to use the OCX control in a visual 'drag-and-drop') design mode *or* any ActiveX compliant development environment to use the .dll as a referenced component in a non-visual design mode.

Scale Communications

This control provides an event-driven method of handling serial communications from the scale. Each control you use corresponds to one serial port and is used to receive data from one scale. If you need to access more than one scale in your application, you must use more than one **WTCommScl** scale communications control. Each serial port address can be set as a property in the control.

With the event-driven method, your application will be notified the moment an event takes place, such as when a complete weight message has been received from the scale. In such a case, you would use the **OnScaleComm** event to trap and handle these conditions in your application program.

Since the **WTCommScl** control uses the Microsoft MSComm as a constituent control, a lot of the

details usually necessary to handle scale communications are hidden and taken care of for you. This includes such items as synchronizing received messages, parsing message strings, extracting and converting weight and status information, handling communication errors and detecting scale disconnect.

This document will describe the control properties, methods and events that are available to the developer.

Properties, Methods and Events

All properties, methods and events for this control are listed in the following table. Scale related properties and events are listed below and are documented in the following sections. Standard properties and events that are inherited from the MSComm control are marked with an asterisk (*) and may not be document here. Please refer to the Microsoft Visual Basic 5.0 (or 6.0) documentation for details.

PROPERTIES:	*CommID	*CommPort	ConnectStatus	FormattedWt
	*Index	*Left	*Name	NetWt
	NetWtUnits	RawWt	RawWtUnits	ScaleStatus
	*Tag	*Top	WtNotification	
	ScaleCommEvent		ControlVersion	
EVENTS:	OnScaleComm			
METHODS:	ScaleOpen()			

CommPort Property

Description: Sets and returns the communications port number.

Synopsis: *[form.]WTCommScl.***CommPort***[=portNumber]*

Remarks: You can set *portNumber* to any number between 1 and 99 at design time. However, the scale communications control generates error 68 (Device Unavailable) if the port does not exist when you attempt to open it with the ScaleOpen method.

Note: This is the property used by the MSCOMM constituent control.

Data Type: Integer
Default: 1

Warning: *You must set the CommPort property before opening the port using the ScaleOpen method.*

ConnectStatus Property

Description: Returns the current scale connection status.
This property is not available at design-time and is read-only at run time.

Synopsis: *[form.]WTCommScl.***ConnectStatus**

Remarks: This is the value returned which indicates whether or not the scale is communicating with the computer. The following table lists the ConnectStatus property settings for the scale communications control.

Setting	Description
wtSCALE_OFFLINE	The scale is not communicating
wtSCALE_ONLINE	The scale is communicating properly.

Data Type: Integer
Default: n/a

FormattedWt Property

Description: Returns the current scale weight in a more readable form.
This property is not available at design-time and is read-only at run time.

Synopsis: *[form.]WTCommScl.***FormattedWt**

Remarks: The raw weight string returned from the scale is encoded in such a way which makes it inconvenient to

display and read directly. This property uses the raw weight string and units-of-measure status code to format a weight string that is more easily read. The following table lists the weight string formats for the vari ous scale units-of-measure.

Scale Units	Weight String Format	
wtUOM_G	XXXX	
wtUOM_KG	XXX.XX	
wtUOM_LB_OZ_DEC	XX:XX.X	
wtUOM_LB_OZ_FRAC	XX: XX	(no fractional ounce)
"	XX : XX-1/4	(one quarter ounce)
"	XX: XX-1/2	(one half ounce)
"	XX : XX-3/4	(three quarters ounce)

Data Type: String
Default: n/a

NetWt Property

Description: Returns the current scale weight as a variant type value which can be used directly in calculations by the application program.
This property is not available at design-time and is read-only at run time.

Synopsis: *[form.]WTCommScl.***NetWt**

Remarks: You can use this property when you want the actual weight value (ie not a formatted string) for use in calculations that are based on weight. The weight value will be in pounds, kilograms, grams or ounces as determined by the current **NetWtUnits** property setting regardless of what units-of-measure the scale raw weight is in. All values are returned in single precision.

Data Type: Single
Default: n/a

NetWtUnits Property

Description: Sets and returns the desired units-of-measure for the **NetWt** property value during run time and may be set at design time.

Synopsis: *[form.]WTCommScl.***NetWtUnits =** *enumUnits*

Remarks: You can choose the units-of-measure from the following enumerated list.

Setting *(enumUnits)*	**Description**
wtPounds	**NetWt** will be in pounds (single precision float)
wtKilograms	**NetWt** will be in kilograms (single precision float)
wtGrams	**NetWt** will be in grams (single precision float)
wtOunces	**NetWt** will be in ounces (single precision float)

Data Type: Integer (enumerated constants)
Default: 1

OnScaleComm Event

Description: The OnScaleComm event is generated whenever the value of the ScaleCommEvent property changes, indicating that either a weight has been received from the scale, the weight value has changed or the scale status has changed.

Synopsis: **Sub** *[form.]WTCommScl.***OnScaleComm()**

Remarks: The ScaleCommEvent property contains the numeric code of the actual cause of the event that generated the OnScaleComm event. See ScaleCommEvent for a list of event codes.

Data Type: n/a
Default: n/a

RawWt Property

Description: Returns the raw weight string value received from the scale.
This value is read only at run time and is not available at design time.

Synopsis: *[form.]WTCommScl.***RawWt**

Remarks: The five characters of raw weight data only are extracted from the ten character data string received from the scale. The leading <STX>, three characters of status and the terminating <CR> are stripped from the received string.

```
<STX> X X X # # # # # <CR>      (original string received from scale)
          # # # # #             (weight data only placed in RawWt property)
```

Note: If the RawWtUnits property is read as **wtUOM_LBOZ_FRAC,** it means that the scale transmits its weight in pounds:ounces and the ounces are fractional (i.e. by 1/4 oz). Under this condition the least significant digit of the RawWt string is encoded to specify the current fractional ounce as described in the following table. For further details please refer to the Serial Communications Protocol document SCP-11 (p/n: 8408-14788-11).

Character	Fractional ounce
0	none
1	1 /4 ounce
2	1 /2 ounce
3	3/4 ounce

Data Type: String
Default: n/a

RawWtUnits Property

Description: Returns the units-of-measure of the raw weight received from the scale. This value is read only at run time and is not available at design time.

Synopsis: *[form.] WTCommScl***.RawWtUnits**

Remarks: Use this property to determine the specific units-of-measure that the scale is transmitting weight data in. The returned value will be one of the units as specified in the following table.

Scale Units	Description	
wtUOM_G	Grams	
wtUOM_KG	Kilograms	
wtUOM_LB_OZ_DEC	Pounds:Ounce	(LSD by 0.1 oz)
wtUOM_LB_OZ_FRAC	Pounds:Ounce	(LSD by 1/4oz)

Data Type: Integer
Default: n/a

ScaleStatus Property

Description: Returns the current scale status code.
This value is read only at run time and is not available at design time.

Synopsis: *[form.] WTCommScl.* **ScaleStatus**

Remarks: Extracts the 'U' status bits from the first character of the three-character status string and converts it to an integer value representing the current scale status as described in the following table. For further details please refer to the Serial Communications Protocol document SCP-11 (p/n: 8408-14788-11).

Setting	Status Description
wtNORMAL_MODE	Normal mode: positive weight
wtTEST_MODE	Test mode: adjust zero counts
wtCALIB_MODE	Calibration mode: adjust span
wtSHOWING_TARE	displaying: **tArE**
wtSHOWING_LO	displaying: **Lo** (low battery)
wtSHOWING_ERR	displaying: **Err** (overload)
wtSHOWING_ERRL	displaying **ErrL** (zero counts too low)
wtSHOWING_DASHES	displaying: - - - - (negative weight, e < 0.00)
wtNOT_USED1	n/a
wtNOT_USED2	n/a
wtNOT_USED3	n/a
wtNOT_USED4	n/a
wtSHOWING_8888	displaying: 8 8 8 8
wtSHOWING_TARE_ERR	displaying: Err (tare error)
wtCALIB_MODE_TARE	Calibration mode: Tare
wtSHOWING_CAL	displaying: CAL (in calibration mode)

Data Type: Integer
Default: n/a

ScaleCommEvent Property

Description: Returns the most scale event.
This property is not available at design time and is read-only at run time.

Synopsis: *[form.] WTCommScl.* **ScaleCommEvent**

Remarks: The ScaleCommEvent property holds the numeric code for the event that caused the OnScaleComm event to occur. Although the MSCOMM constituent control is generating many more serial communications events, the ScaleCommEvent is generated only under certain conditions. The events are described in the following table.

Setting	Description
wtWTCOMM_EV_WEIGHT	weight updated (or changed)
wtWTCOMM_EV_STATUS	status changed
wtWTCOMM_EV_DISCONNECT	scale has disconnected (offline)

Data Type: Integer
Default: n/a

ScaleOpen Method

Description: Sets and returns the state of the scale communications port (open or closed).
This property is not available at design time.

Synopsis: **Function** *[form.]WTCommScl.***ScaleOpen({True** | **False})**

Remarks: The following table lists the ScaleOpen parameter settings for the scale communications control.

Setting	Description
True	Open the port and establish communications with scale.
False	Close the port.

Calling the ScaleOpen method with the parameter set to **True** opens the serial port and establishes the communications link with the scale. Setting it to **False** closes the port.

Note: This is similar to the PortOpen property used by the MSCOMM constituent control with the additional requirement that a communications link with the scale also be established.

Warning: *You must set the CommPort property before opening the port using the ScaleOpen method. It is recommended that you call ScaleOpen(False) before terminating your application.*

Data Type: Integer
Default: n/a

WtNotification Property

Description: Sets and returns the condition under which the OnScaleComm event will be generated.

Synopsis: **[***form.***]***WTCommScl***.WtNotification [** = **{wtEverytime** | **wtChanged }]**

Remarks: You can set this property to cause the OnScaleComm event to occur everytime a weight is read from the scale or only when the weight value (or status) has changed.

Note: If the property is set to wtEverytime, the event will be generated approximately four times per second which is the rate of continuous transmission from the NCI Model 7010 scale.

Data Type: Integer
Default: wtEverytime

ControlVersion Property [Added in v3.0]

Description: Returns the current version of this ActiveX control.
This property is not available at design-time and is read-only at run time.

Synopsis: *[form.]WTCommScl.***ControlVersion**

Remarks: This property returns a string value of the major version and minor version numbers.
Ex.: **(2.1)**

Data Type: String
Default: n/a

Enumerated Constants

Enumerated Constant Definitions

Description: This is the public constant declarations list for enumerated constants defined in the **WtCommScl** ActiveX control (NCI p/n: 1150-16067) developed by Weigh-Tronix/NCI. The developer should use these constants when accessing various properties, events and methods in the scale communications control.

Note: Since these constants are defined and made public in the WTCommScl control, no additional file needs to be added to the project in order to use these constants.

NetWtUnits

wtPounds (=0)	(converted/presented in decimal pounds)
wtKilograms (=1)	(converted/presented in decimal kilograms)
wtGrams (=2)	(converted/presented in grams)
wtOunces (=3)	(converted/presented in decimal ounces)

WtNotification

wtEverytime (=0)	(every time a weight is received)
wtChanged (=1)	(only when the weight value has changed)

WTCONST.BAS

Constant Definitions

Description: This is the public constant declarations file for use with the **WTCommScl** ActiveX control. The developer must add this file to the application project to use these constants when accessing various properties, events and methods in the scale communications control.

ConnectStatus

wtSCALE_OFFLINE = 0

wtSCALE_ONLINE = 1

ScaleStatus

wtNORMAL_MODE = 0
wtTEST_MODE = 1
wtCALIB_MODE = 2
wtSHOWING_TARE = 3
wtSHOWING_LO = 4
wtSHOWING_ERR = 5
wtSHOWING_ERRL = 6
wtSHOWING_DASHES = 7
wtNOT_USED1 = 8
wtNOT_USED2 = 9
wtNOT_USED3 = 10 wtNOT_USED4 =
11 wtSHOWING_8888 = 12
wtSHOWING_TARE_ERR = 13
wtCALIB_MODE_TARE = 14
wtSHOWING_CAL = 15

ScaleCommEvent

wtWTCOMM_EV_WEIGHT = 1	(weight updated or changed)	
wtWTCOMM_EV_STATUS = 2	(status changed)	
wtWTCOMM_EV_DISCONNECT = 3	(scale disconnected , went offline)	
wtWTCOMM_EV_RECONNECT = 4	(scale reconnected , back online)	**[Added in v3.0]**

RawWtUnits

wtUOM_NONE = 0	(no units of measure recv'd with weight)
wtUOM_KG = 1	(weight is in kilograms)
wtUOM_LBOZ_DEC = 2	(weight is in pounds:ounces, LSD by 0.1oz)
wtUOM_G = 3	(weight is in grams)
wtUOM_LBOZ_FRAC = 4	(weight is in pounds:ounces, LSD by 1/4 oz)

APPENDIX-A

New Features [Added in v3.0]

- A new **`ControlVersion()`** method to allow applications to determine the software version of the ActiveX Control.

- A new **`wtWTCOMM_EV_RECONNECT`** event to signal the application that the scale communications has been reestablished.

- Faster 'scale disconnected' detection time.

- An 'auto-reconnect' mechanism to periodically check to see if scale communications has been re-established. Please see the **Scale Connection Notes** topic below for further details.

- A new **WTCommScl Tester-3** application used to validate operation of the (v3.0) ActiveX Control and also allow convenient demonstration of the scale disconnect and reconnect mechanism.

  The WTCommScl Tester 3 application is available separately (free-of-charge) on the Salter Brecknell software download website at:

  http://www.brecknellscales.com/index.php?option=com_content&task=view&id=9&Itemid=36

Scale Connection Notes [Added in v3.0]

With the release of version 3.0 of this control, several features have been added in an attempt to provide a more robust connection (and reconnection) mechanism between the scale and a PC application. The following narrative describes the connection scenario. References to 'test status' are messages that would be indicated on the WTCommScl Tester-3 demo/test program at various points in the connection example.

- The scale may be connected to a PC using either an RS-232 or USB cable. A scale connected via a USB cable is likely to receive its' power over the same cable and so may result in a slightly different disconnection scenario described later.

- Initial Connection for a given session requires that the controlling application first must set the **`CommPort`** property (of the RS-232 or USB Virtual Comm Port) and then call the **`ScaleOpen(True)`** method. The scale may be powered on or off when the ScaleOpen() method is called. Test status: **MANUAL SCALE OPEN**

  Note: If the **`ScaleClosed()`** method is subsequently called, the demo program indicates status: **MANUAL SCALE CLOSED**

- IMPORTANT: it is recommended that after calling **`ScaleOpen()`**, the application next read the **`ConnectStatus`** property from the control. If the **`ConnectStatus`** value is **`wtSCALE_ONLINE`**, then the initial startup is complete and weight data will be available to the application. If the value is **`wtSCALE_OFFLINE`** then the scale is either not connected or not powered on. In that case a warning message might be displayed to a user. For the demo program the status displayed will be: **NO DATA FROM SCALE. PLEASE TURN SCALE ON**.

- Once communications has been established, the control will monitor the receipt of the continuous data from the scale. If any condition occurs which prevents valid scale data from being received for one (1) second, then the control will raise the **`wtWTCOMM_EV_DISCONNECT`** event to the controlling application.

- The control will continuously attempt to automatically reconnect with the scale every three (3) seconds. If the condition causing the loss of data from the scale is corrected, the control will resume reception and raise the **`wtWTCOMM_EV_RECONNECT`** event. It is not necessary for the calling application to call the **`ScaleOpen()`** method again unless (during the loss of data period) it explicitly closed the communications port by calling the **`ScaleOpen(False)`** method or the application had been terminated.

- Scale communications with the controlling application can be lost if the interconnect cable (RS-232 or USB) is disconnected or if scale power is turned off.

- When using an RS-232 cable, it is likely that the scale will not lose power if the cable is disconnected. Therefore, when the cable is reconnected, the reception of scale data will be detected by the control, the **wtWTCOMM_EV_RECONNECT** event will be raised, and the application can resume.

- When using a USB cable, (with USB powered scales), the scale will in fact lose power if the cable is disconnected. In this case, Windows will also automatically unload the virtual communications port (VCP) driver. Subsequently, when the USB cable is reconnected it is likely that the user will first have to turn scale power on manually at which point Windows will load the VCP driver, the reception of scale data will be detected by the control, the **wtWTCOMM_EV_RECONNECT** event raised, and the resumption of the application can occur.

Upgrading from (v2.x) to (3.x) [Added in v3.0]

This new version of the ActiveX control is recommended for new designs only. Binary-compatibility with the previous (v2.0) control has been intentionally broken. If you do not need the capabilities added in (v3.0) or above, you may continue to use the version (v2.0) control and this document for your legacy applications.

If you choose to replace the legacy (v2.0) ActiveX control currently in your application with the new (v3.0) ActiveX control within a .NET environment, please be aware of the following considerations:

- If you are in a visual development environment you should remove all references to the WTCommScl (v2.0) ActiveX control, remove the control from the toolbox, and then uninstall the control from your computer. Next, install the new (v3.0) ActiveX control and add it to your toolbox. This should automatically add references to the control class (**WTCommScl.OCX**) and the two new interop wrappers (**AxInterop.WTCommScl_OCX.DLL** and **Interop.WTCommScl_OCX.DLL**) that are needed. If not, you will have to manually handle this requirement as needed in your environment.

- In some environments you will now need to fully qualify names to access an enumeration. For example:

```
axWTCommScl1.WtNotification = WT_NOTIFY_ENUM.wtEverytime;

must now become:

axWTCommScl1.WtNotification = WTCommScl_OCX.WT_NOTIFY_ENUM.wtEverytime;
```

- The base WTCommScl control is a 32-bit ActiveX. If the control will be used on a 64-bit PC, then you must install the **WTCommScl ActiveX Control -- Upgrade (v2/v3)** after installing the base control. The upgrade may also be installed on 32-bit PC's.