SHUMATECH

OpenDRO User's Guide

Release 1.0.0

Revision History

Revision	Description
1	OpenDRO release 0.3.0
2	OpenDRO release 0.3.1
3	OpenDRO release 0.4.0
4	Revised iGaging scale description
5	OpenDRO release 1.0.0

Table of Contents

1 Introduction

OpenDRO is the open source software that runs on the ShumaTech DRO-550 and DPU-550 digital read-out (DRO) systems. It is designed to run on Atmel's SAM series of ARM7 processors and is developed in C with some ARM assembly in critical spots. OpenDRO is architected to be generic and easily ported to new hardware platforms.

OpenDRO supports all known protocols of the ubiquitous Chinese scales that can be purchased at a number of retailers like Harbor Freight. It also supports quadrature signals so you can use most glass scales and rotary encoders as well. It features advanced adaptive moving average filtering to eliminate display jitter often associated with Chinese scales.

Positioning operations in OpenDRO are accomplished by "working to zero" where a number is entered into an axis and you move that axis to show zero in order to arrive at the destination. OpenDRO supports both absolute and incremental zeros as well as workspaces to save working sets of zeros. You can enter zeros with the keypad in incremental or absolute positions as well as using an electronic edge finder.

OpenDRO on the DRO/DPU-550 supports a three axis main display using 7-segment LED modules and also a two axis, external LCD display for the DRO-550 called the LCD-200. You can see axis readings in either metric or imperial units and can also configure the number of decimal places displayed. In addition to floating point, OpenDRO can also show tachometer readings, fractional readings, feed rates, and several other types of readings.

You can perform a variety of complex functions with OpenDRO. It supports tool offsets, bolthole circles, diameters, centerlines, compound vectoring, and a number of other functions. For multi-step functions, OpenDRO steps the user through each operation and you can move back and forth at will through each operation.

The most advanced part of OpenDRO is its automation interface. OpenDRO uses the Lua scripting language to enable the user to enhance the functionality of OpenDRO as they see fit. You can store and run Lua scripts on an SD memory card and run these scripts on demand from the keypad. OpenDRO has an advanced shell with line editing, command completion, and history so you can easily experiment and test automation tasks from the USB or RS-232 interfaces.

OpenDRO is easily installed over the USB or RS-232 interfaces using either Microsoft Windows or Linux. More information on the exact software upgrade procedures is found on the product pages at the ShumaTech web site.

2 Features

OpenDRO incorporates a very large feature set and the major features are summarized below. You can find more detailed information in the sections that follow.

Scales

- Support for Chinese (24BIT, 21BIT, BCD7, and BIN6 protocols) and quadrature scales
- Chinese scale protocol auto-detection
- Fast scale reading for Chinese scales with a display update rate of up to 50 times a second
- Scale summing
- Adaptive moving average scale filtering to minimize jitter
- Linear error correction

Zeroing

- Absolute and incremental zeros
- Enter zero values for both coordinate systems via keypad
- Enter incremental presets via keypad
- Enter fractional numbers via keypad
- Zero warning with axis blink and piezo buzzer
- Electronic edge finder zeroing with audible alert

Display

- Display units in inches or millimeters (mm)
- Configurable display precision up 4 decimal places in inch mode (3 for mm mode)
- Auto-precision to automatically reduce precision to fit the display
- Fractional display in 64ths
- Raw scale count display
- Feed rate display in inches per minute or millimeters per minute
- Tachometer display in RPM or surface feet per minute (SFM)
- Display brightness configurable in 20% increments

Functions

- Tool offsets
- Bolthole circles

- Sub-datum workspaces
- Hole grids
- Radius cutting (concave and convex)
- Compound vectoring
- Centerline calculation
- Diameter mode
- Auto power off and on

Automation

- USB and RS-232 serial interfaces
- Software upgradeable
- Lua command interface
- Run Lua scripts from keypad
- SD memory card support (SD and SDHC)
- Lua shell with advanced editing (edit keys, history, and completion)
- Windows virtual DRO application to control DRO remotely

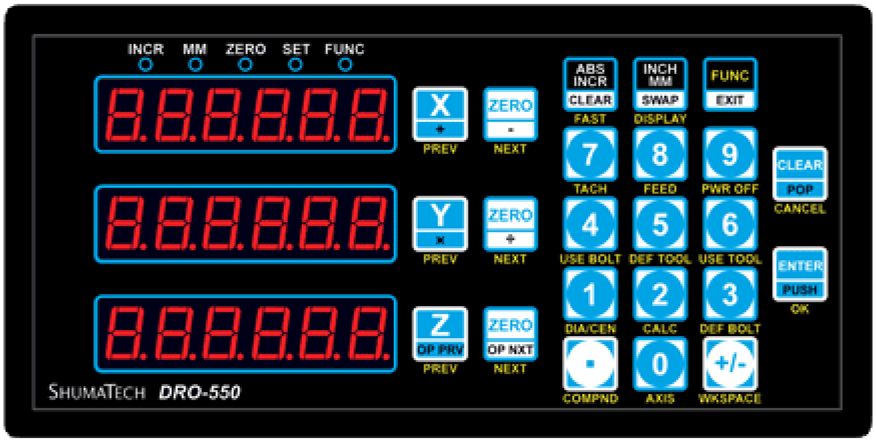
Utilities

- Reverse polish notation (RPN) calculator with a 4 level stack
- RPN calculator operations: addition, subtraction, multiplication, division, cosine, sine, tangent, square root, power, memory, scale transfer, fractions, and more
- Audible key presses

.. and more!

3 Front Panel

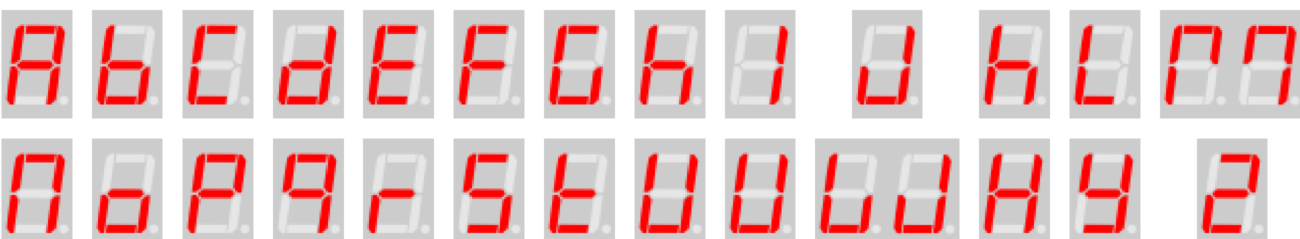
The DRO/DPU-550 front panel is shown below. On the left side is the main display that consists of three axis displays for the X, Y/Z1, and Z/Z2 axes. Above the three axis displays are five LED indicators that show current state and status. The right half of panel contains the keypad. The keypad is used to set presets, set zeros, change modes, begin functions, enter values, etc. The keypad includes text integrated into each key that gives its function when using the built-in RPN calculator. There is also yellow text below each key that shows the keys function when the FUNC key is pressed.



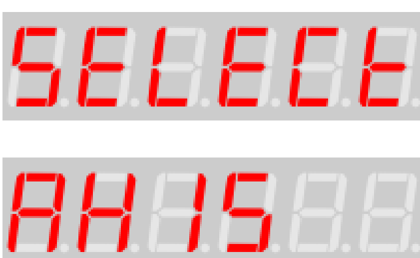
DRO/DPU-550 Front Panel

3.1 Axis Display

The main axis display consists of three rows of six columns each using seven segment LED displays. This is where the main axis readings are displayed when the DRO is operating and it is also where textual messages and prompts are displayed. Unfortunately, seven segment displays do not excel at displaying text so we must use a bit of imagination for some of the letters. Shown below is the full OpenDRO alphabet.



Note that the M and W take two digits to display. Using the alphabet, the message "select axis" would look like the following:



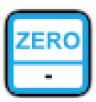
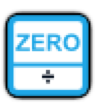
When displaying axis readings, the left-most decimal point is used to represent that the axis reading is not its true value due to either a tool offset or due to diameter mode. This serves as a reminder that one of these functions is active on the axis. The right-most decimal point is used as a half-digit indicator when the axis precision is set to a half digit granularity. For example, "1.231." represents 1.2315 with the final decimal point representing the 0.0005 value.

3.2 Indicators

INCR	**Incremental Mode** This indicator is on when the DRO is in incremental mode. All axes will display their positions relative to their incremental zeros. When the indicator is off, the DRO is in absolute mode and each axis will display its position relative to its absolute zero. The ABS/INCR key toggles between incremental and absolute modes.
MM	**Metric Mode** This indicator shows when the DRO is in metric mode. When in metric mode, all positions either displayed by the DRO or entered via the keypad are in millimeters. When the indicator is off, the DRO is in imperial mode and all positions are in inches. The INCH/MM key toggles between metric and imperial modes.
ZERO	**Absolute Zero Set** The absolute zero set indicator is on when the DRO is setting an absolute zero. The ENTER key or the axis ZERO set key must be pressed to complete the absolute zero set. Alternatively, an electronic edge finder can be used to trigger the absolute zero set at the position where an edge is contacted.
SET	**Set Value** The set value indicator is a prompt that the DRO expects the user to enter a number on the numeric keypad. When it flashes rapidly, then it indicates that you are in number offset mode.
FUNC	**Function** The function indicator turns on when the FUNC key is pressed to indicate that a function should be selected through the function menu or via a speed key. The indicator is also kept on if a function such as bolthole or grid is currently active. When it turns

	off, the DRO is back in its normal mode.

3.3 Keypad

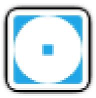
	**X Preset** This key begins a preset operation on the X axis. See the operation section for more detail on how to set a preset.
	**Y Preset** This key begins a preset operation on the Y axis for a mill or the Z1 axis for a lathe. See the operation section for more detail on how to set a preset.
	**Z Preset** This key begins a preset operation on the Z axis for a mill or the Z2 axis for a lathe. See the operation section for more detail on how to set a preset.
	**X Zero** This key is used to set either an incremental or absolute zero on the X axis. For detailed information on setting axis zeros, see the operation section.
	**Y Zero** This key is used to set either an incremental or absolute zero on the Y axis for a mill or the Z1 axis for a lathe. For detailed information on setting axis zeros, see the operation section.
	**Z Zero** This key is used to set either an incremental or absolute zero on the Z axis for a mill or the Z2 axis for a lathe. For detailed information on setting axis zeros, see the operation section.

Numeric Keypad

Numbers are entered into the DRO via the numeric keypad. It is used during preset and zero operations to enter offsets and is also used to enter information for function operations. The set indicator is lit when the DRO expects the user to enter a number on the numeric keypad.

The ENTER key is used to finish number entry.

The CLEAR key is used to delete the right-most digit or decimal point or to exit numeric entry when all digits are cleared.



Decimal Point

This key enters a decimal point at the current position when entering a number on the numeric keypad.

The CLEAR key can be used to delete the decimal point if it is at the right-most position. A second decimal point is used to indicate that the number entered is a factional number instead of a floating point number as described below.

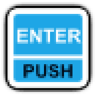
Sign

The sign key toggles between a positive or negative value while entering a number on the numeric keypad. It can be pressed at any time during number entry.

Clear

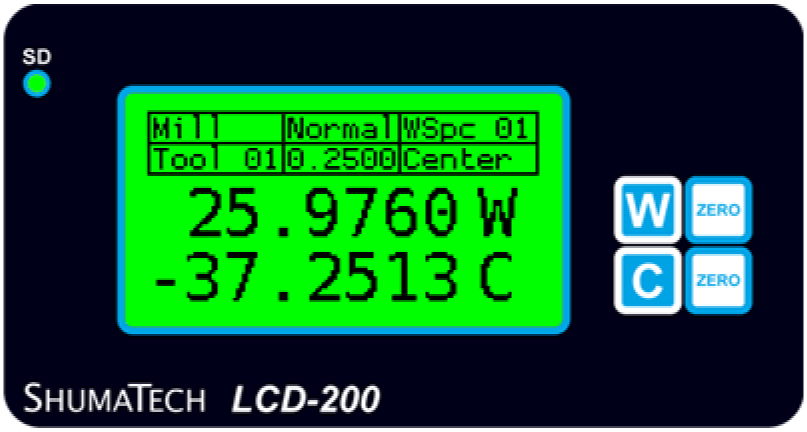
The clear key serves a variety of uses:

- During number entry, it deletes the right-most digit or decimal point. If the number is a single zero, it will end number entry without entering a number.
- During setup, it serves as an abort key to not save the changed configuration.
- During functions, it serves as a backward key while moving through function steps.

	**Enter** The enter key serves a variety of uses: • During number entry, it is used to finish entering the number. • During setup, it serves as an accept key to save the changed configuration. • During functions, it serves as the forward key while moving through function steps.
	**Absolute/Incremental** This key toggles between absolute and incremental modes for all axes.
	**Imperial/Metric** This key toggles between imperial (inches) and metric (millimeters) modes for all axes.
	**Function Key** The function key is pressed to display the function menu. You can also press another key on the keypad as a speed key to quickly access a function rather than selecting it through the function menu.

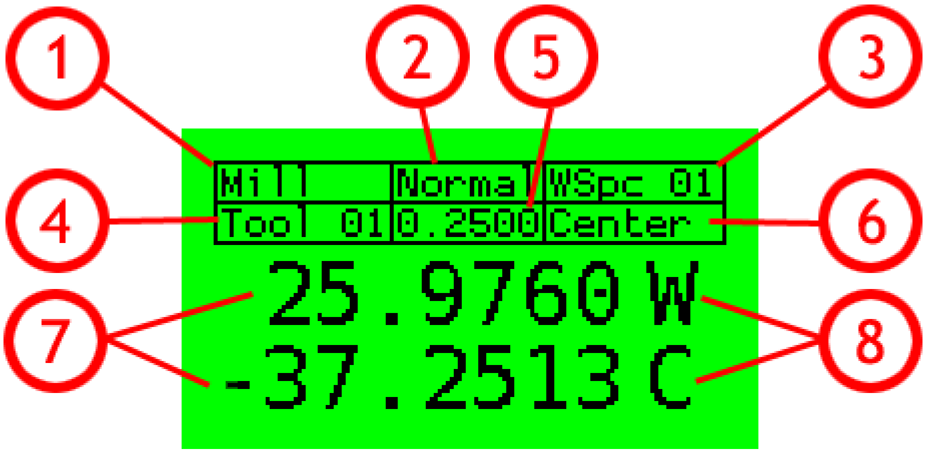
3.4 LCD-200

OpenDRO supports the LCD-200 secondary LCD display that shows two additional axes as well as current status information. Like all OpenDRO axes, you can configure which axes are displayed on the LCD. You can also display other information such as the tachometer, SFM, or axis feed rate. The LCD-200 also has preset and zero keys for these axes as well. Finally, the LCD-200 includes an SD memory card reader in the side so that data can be easily exchanged with a PC. Note that the LCD-200 is only supported on the DRO-550. The front panel is shown below for reference.



LCD-200 Front Panel

The LCD screen dominates the LCD-200 front panel. The figure below shows an enlargement of the LCD screen with callouts listing each of its visual components.



LCD Display

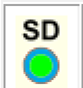
1. Machine Type - Displays the selected machine type either "Mill" or "Lathe".
2. Function State - Displays the active function - "Bolt" for a bolthole operation, "Grd" for a grid operation, "Calc" for RPN calculator, 'Rad' for a radius operation, or "Normal" if no function is active. The selected bolthole, grid, or radius number is also shown after the text.
3. Workspace/Step - During normal operation, this displays the active workspace. During a function operation, this displays the function step number.
4. Tool Number - Displays the active tool number or "No Tool" if none is selected.
5. Tool Offset - Displays the diameter of the active tool.

6. Edge/Compound - In mill mode, this displays the edge compensation side, either "Left", "Right", "Back", "Front", or "Center". In lathe mode, this displays the angle of the compound if compound vectoring is selected.
7. W/C Axis - These are the fourth and fifth display lines. They show the W and C axes by default though you can configure them to display any axis. You can also display tachometer, SFM, or feed rate data on either or both lines.
8. Axis Labels - Displays the labels for what the fourth and fifth display lines are showing. For a tachometer, a "T" is shown. For feed rate, an "F" is shown with a subscript of the axis letter.

The LCD-200 also has additional keys that control its operation. These additional keys are summarized below.

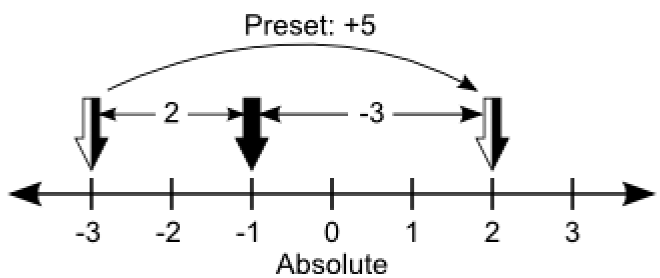
	**W Preset** This key begins a preset operation on the W axis. By convention, the W axis is a secondary Z axis. See the operation section for more detail on how to set a preset.
	**C Preset** This key begins a preset operation on the C axis. By convention, the C axis is a rotational axis in the Z plane. See the operation section for more detail on how to set a preset.
	**Zero Axis** This key is used to set either an incremental or absolute zero on an axis. For detailed information on setting axis zeros, see the operation section.

The LCD-200 also includes one additional indicator LED as described below.

	**SD Activity** This indicator is lit whenever the SD memory card in the LCD-200 is accessed. You should not remove the card if this indicator is lit or flashing.

4 Operation

The main operation to master on the DRO is setting zeros. OpenDRO uses a "work to zero" philosophy where you set a zero and then machine towards that zero. There are two kinds of zeros - absolute and incremental. Absolute zeros are usually set once when starting on a work piece and correspond to the origin for the work. Incremental zeros are set for each movement you make. For example, assume you are at absolute position -1 and you set an incremental zero at absolute position -3. The display will show 2 in incremental mode indicating that you are 2 units to the positive side of the incremental zero. If you switch back to absolute mode, you are still at position -1. If you then set an incremental zero at an incremental preset of 5, then the incremental zero moves 5 to the right to absolute position 2 and the display shows -3 in incremental mode and still -1 in absolute mode. The picture below shows this example visually with the incremental zeros depicted as half black and half white arrows and the current position as a solid black arrow.



The sections below will detail how to set each type of zero and how to enter and modify numeric data.

4.1 Absolute Zero

Start an absolute zero set on an axis by pressing the ZERO key for that axis when the DRO is in absolute mode. The zero set indicator will turn on indicating that an absolute zero set is in progress. After an absolute zero set is started, there are three ways to set the absolute zero, at the current position, by an offset from the current absolute zero, or at the position of an electronic edge finder.

- Set an absolute zero at the current position by pressing either the ENTER key or the ZERO key.
- Set an absolute zero at an offset from the current absolute zero by pressing the preset key and entering the offset on the numeric keypad. Use the ENTER key to complete the position and the CLEAR key to make corrections or cancel the absolute zero.
- Set an absolute zero with an electronic edge finder by contacting the work piece with the electronic edge finder. An absolute zero is set at the current position on the instant of contact. Before using the electronic edge finder, make sure it is properly connected to the DRO and that the machine setting in the setup is set to use it.

4.2 Incremental Zero

Set an incremental zero at the current position by pressing the ZERO key when the DRO is in incremental mode. The incremental zero is set immediately without any additional key presses.

4.3 Absolute Preset

An absolute preset is an incremental zero set at an absolute position. With the DRO in absolute mode, press the preset key for the desired axis. The DRO will switch to set mode and you can enter an absolute position via the numeric keypad. Use the ENTER key to complete the position and the CLEAR key to make corrections or cancel the absolute preset.

4.4 Incremental Preset

An incremental preset is an incremental zero set at an offset from the current incremental zero. With the DRO in incremental mode, press the preset key for the desired axis. The DRO will switch to set mode and you can enter an offset via the numeric keypad. Use the ENTER key to complete the offset and the CLEAR key to make corrections or cancel the incremental preset.

4.5 Number Review

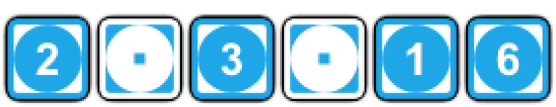
Any time the DRO expects you to enter a floating point number, as evidenced by the SET indicator LED at the top of the display being on, the previous value is displayed for review until you start entering a new number to replace it. If you press the ENTER key without entering a new number, then the previous value is left unmodified.

4.6 Number Offsets

When you are entering a floating point number, you can enter an offset from the previous value by pressing the ABS/INCR key. The SET indicator LED will flash quickly to indicate that you are in offset mode. You can press the ABS/INCR key again to exit offset mode and go back to normal number entry. When in offset mode, any number you enter will be taken as an offset from the previous value. For example, if the previous value is 1.5 and you enter 0.25 in offset mode, then the new value is 1.75.

4.7 Fractional Numbers

You can enter a fractional number any time the DRO is expecting you to enter a floating point number. For example, to enter the fractional number 2 3/16, press the following keys:



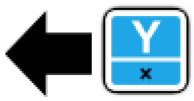
The display will also show "2.3.16" and when you press ENTER, the display will change to the fractional number's decimal equivalent of 2.1875. You can use the CLEAR key just like when entering floating point numbers and it will back up past numbers and decimal points.

5 Function Menu

The function menu is displayed by pressing the FUNC key. The FUNC indicator will turn on and the function menu is shown on the display. The top line will display FUNC as a reminder. The second line will display the menu and the third line will display the menu item. You can move through the menus by pressing the Y ZERO key to move forward and the Y PRESET key to move backward. The menu item is correspondingly changed by pressing the Z ZERO key to move forward and the Z PRESET key to move backward. When you have the desired menu item representing the function you want to perform, then press the ENTER key to activate it. Press the CLEAR key to exit the function menu without activating a function.

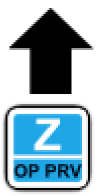
The following table summarizes the function menu. Differences between mill and lathe operation are noted.

Function Menu



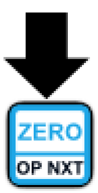
PREVIOUS

NEXT



PREVIOUS

NEXT



Position	Status	Define	Setup	System
Tool Offset	Tachometer	Tool Offset	Machine	Power Off
Diameter (Lathe)	Feed Rate	Bolthole (Mill)	Axis	Machine
Centerline (Mill)	Calculator	Grid Pattern (Mill)	Scale	Send Fast
Compound (Lathe)		Radius (Mill)	Display	Version
Bolthole (Mill)			Keypad	Update (DRO-550)
Grid Pattern (Mill)				Save (DRO-550)
Radius (Mill)				Load (DRO-550)
Workspace				Erase

5.1 Speed Keys

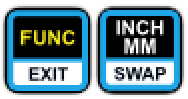
Speed keys are numeric shortcuts that jump straight to a menu item without having to navigate through the menus. To execute a speed key, first press the FUNC key and then press a number key 0 through 9. The FUNC menu is displayed until the speed key is pressed. The speed keys are shown in the table below with mill and lathe differences noted.

Speed Keys

Speed Key	Menu	Item
FUNC/EXIT .	Position	Compound (Lathe)
FUNC/EXIT 0	Setup	Axis
FUNC/EXIT +/-	Position	Workspace
FUNC/EXIT 1	Position	Centerline (Mill) Diameter (Lathe)
FUNC/EXIT 2	Status	Calculator
FUNC/EXIT 3	Define	Bolthole (Mill)
FUNC/EXIT 4	Position	Bolthole (Mill)
FUNC/EXIT 5	Define	Tool
FUNC/EXIT 6	Position	Tool
FUNC/EXIT 7	Status	Tachometer
FUNC/EXIT 8	Status	Feed Rate
FUNC/EXIT 9	System	Power Off

Special Function Speed Keys

Item	Description

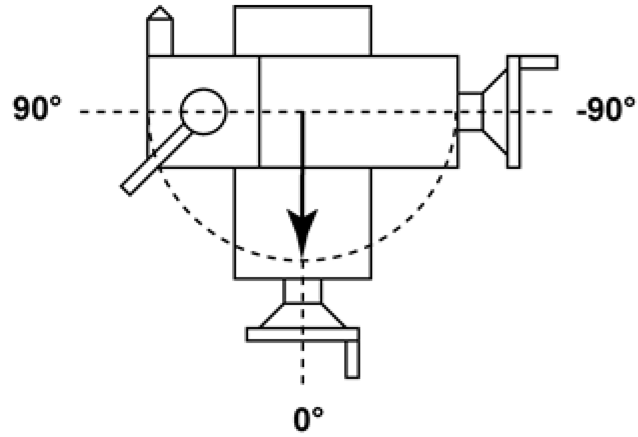
	**Fast Mode Toggle** This function will program the fast mode sequence on any 24BIT or BCD7 protocol scales connected to the DRO that are not already in fast mode. This function can be used to put a newly connected scale into fast mode without powering the DRO off and back on.
	**Display Toggle** This function is used to toggle between displaying the axis reading, the raw scale counts, and the scale jitter. This function is mainly used for troubleshooting and helping to determine how much a scale reading is jittering. Each activation of this function will toggle between the three possible displays.
	**User Script** This function is used to run a user-defined, Lua automation script. After entering this speed key, you are prompted to enter a number from 0 to 9 representing one of the 10 user functions named user0 through user9, respectively. See the automation section for more information on the automation interface.

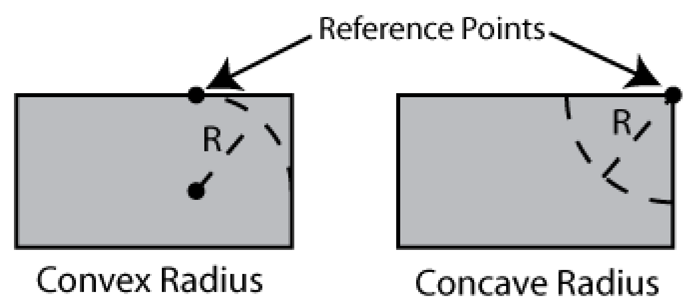
5.2 Position Menu

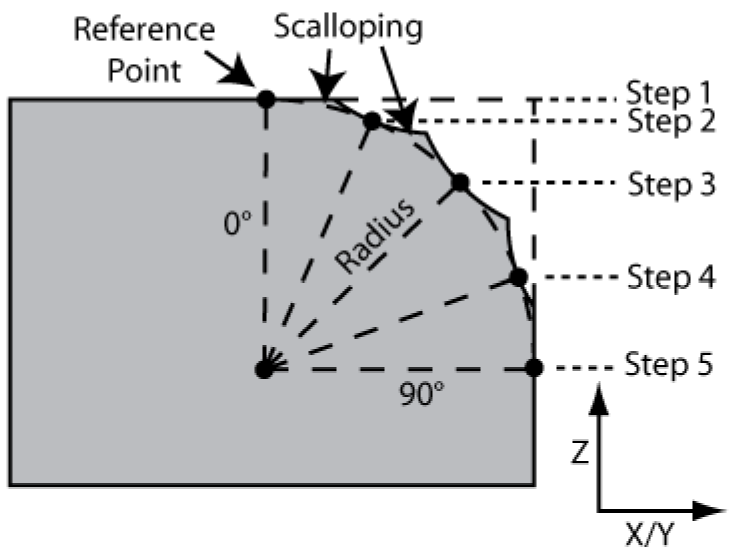
Note that the position menu is displayed on screen as "PoSitN". The on-screen display for each menu item is shown underneath the menu name.

Item	Machine	Description
Tool Offset "tooL"	Mill	This function selects one of the defined tool offsets. The DRO prompts you to select a tool offset number with the numeric keypad. Tool offset 0 is a special number that applies a tool with no offsets and is effectively used to turn off tool offsets. After selecting the tool offset, use the numeric keypad to select the cutting edge to compensate for. The edge positions are selected according to the following keypad pattern.

		 When an axis position is compensated due to an edge position selection, the left-most decimal point in the display for that axis is turned on.
Tool Offset "tooL"	Lathe	This function selects one of the defined tool offsets. The DRO prompts you to select a tool offset number with the numeric keypad. Tool offset 0 is a special number that applies a tool with no offsets and is effectively used to turn off tool offsets.
Centerline "C-LINE"	Mill	Centerlines are meant to quickly find center points in lines and circles. The centerline function sets a zero at the half way point between the current zero and current position for an axis. If the scale is in incremental mode, the zero will be an incremental zero. Vice versa, if in absolute mode, the zero will be an absolute zero. Set a centerline zero by first moving to the desired position. Switch to incremental or absolute mode with the ABS/INCR key for the desired zero type. Select the centerline menu item and the DRO will display a prompt to press the preset key for the axis the zero is desired on.
Diameter "dIA"	Lathe	Diameter mode doubles an axis dimension to show the diameter of a work piece instead of its radius. The left-most decimal point in the axis display is turned on to indicate that the DRO is in diameter mode. When this operation is selected, the DRO will prompt you to select the axis to apply the diameter to. Selecting this operation when diameter mode is already active will clear it.
Compound Angle "CoMPN"	Lathe	This function sets the angle of the compound for use when vectoring the compound (Z2 axis) movement into the cross-slide (X axis) and carriage (Z1 axis). Positive angles indicate that the compound is positioned clockwise from the center position and negative angles indicate that the compound is positioned counter-clockwise.

		 Enter the angle from positive to negative 99.9 degrees with the numeric keypad and press the ENTER key to accept the angle. Press the CLEAR to make corrections or exit without setting the angle.
Bolthole "boLtho"	Mill	This function uses the defined bolt-hole pattern to set X and Y axis incremental zeros at each bolt-hole position in the pattern. The center of the bolt-hole pattern is the incremental zeros of the X and Y axes when the function is started so before starting this function, set X and Y incremental zeros at the center of the bolt-hole pattern. A "hole 1" prompt is displayed when this function is initiated. This indicates that an incremental zero is about to be set for the first bolt-hole. Press the ENTER key and the DRO switches to displaying the axis positions so that you can move the X and Y axes to zero for the first bolt-hole. Press the ENTER key again and continue through the bolt-hole pattern. After the incremental zero is set for the last bolt-hole, press ENTER to exit out of the bolt-hole pattern use function. The incremental zeros for the X and Y axis are restored to the values that were defined for the center of the bolt-hole pattern. If you make a mistake, you can go backward through the screens at any time with the CLEAR key.
Grid "GrId"	Mill	This function uses the defined grid pattern to set X and Y axis incremental zeros at each grid hole in the pattern. The center of the first hole in the grid pattern is the incremental zero of the X and Y axes when the function is started. Before starting this function, set X and Y incremental zeros at the center of the first grid hole. A "hole 1" prompt is displayed when this function is initiated. This indicates that an incremental zero is about to be set for the first grid hole. Press the ENTER key and the DRO switches to

		displaying the axis positions so that you can move the X and Y axes to zero for the first hole. Press the ENTER key again and continue through the grid hole pattern. After the incremental zero is set for the last grid hole, press ENTER to exit out of the grid use function. The incremental zeros for the X and Y axis are restored to the values that were defined for the first hole in the grid pattern. If you make a mistake, you can go backward through the screens at any time with the CLEAR key.
Radius "rAdIUS"	Mill	The radius function is used to cut a convex or concave radius into a work piece by progressing through the cut in a series of small steps. See the description of the radius function in the define menu section below for more detail on the parameters that define the radius. The reference point for the radius is taken from the incremental zeros for the two axes in the plane of the radius, either X and Z or Y and Z. The reference point is defined as the center point of the radius for a concave radius or the point on the radius directly above the center point for a convex radius. You must define the incremental zeros that define the reference point before starting the function. The figure below shows the location of the reference points for both convex and concave radii.  Before starting the function, select the tool that you want to use for the cut. The tool must be a [ball nose cutter](#) and the diameter and Z offset must be correctly defined for the tool. If you initiate the radius function without a tool selected, it will display a message and cancel the function. A "step 1" prompt is displayed when this function is initiated. This indicates that incremental zeros are about to be set for the first step cut. Press the ENTER key and the DRO switches to displaying the axis positions so that you can move the Z and X/Y axes to their zeros for the first step. After reaching the zero points, you can lock those axes and begin the cut. The cut is made by moving the third axis, either X or Y, that is

		perpendicular to the plane of the radius. When finished with the current cut, press the ENTER key again and proceed through the rest of the cuts. After incremental zeros are set for the last step, press ENTER to exit out of the radius function. The previous incremental zeros for the X, Y, and Z axes are restored to the values that were defined before starting the radius function. Also, the cutting edge compensation is reset to the previous setting. If you make a mistake, you can go backward through the screens at any time with the CLEAR key. An example of a finished radius cut is shown below. In this example, the radius is convex, has a start angle of 0 degrees, has an end angle of 90 degrees, and is defined for 5 steps. Notice the circular cuts from the ball nose cutter are tangent to the radius at each step point but arc out from that point leaving excess material. This is called scalloping and can give the radius a non-smooth finish. To reduce scalloping, define the radius with more steps and/or use a larger diameter ball nose cutter. 
Workspace "WSPAC"	Both	Workspaces, sometimes known as sub-datums, are incremental and absolute zero points defined in addition to the primary zero points. When you switch to a new workspace, the current zeros are saved and the zeros for the new workspace are used instead. You can switch around between all of the workspaces and the zero points for each workspace are permanently saved. Zeros for the other workspaces are established just like they are with the primary zeros using the ZERO and PRESET keys. Workspaces are useful when switching between multiple parts or when working on a drawing that establishes a new datum and using an

		incremental zero is not convenient or possible.

5.3 Status Menu

Note that the status menu is displayed on screen as "StAtUS". The on-screen display for each menu item is shown underneath the menu name.

<table>
<tr><th>Item</th><th>Machine</th><th>Description</th></tr>
<tr><td>Tachometer
"tACh"</td><td>Both</td><td>Selecting this function will toggle the tachometer display off and on. You are first asked to enter the desired units for the tachometer display either in RPM or in surface rate. You then select the axis line to replace with the tachometer reading.

The surface rate is calculated with the RPM reading from the tachometer and the currently selected tool diameter and calculates the surface rate with the equation Surface Rate = (RPM * 2 * PI * radius).</td></tr>
<tr><td>Feed Rate
"FEEd"</td><td>Both</td><td>Display the instantaneous rate of movement per second for an axis. After initiating this function, the DRO will prompt you to select an axis to display the rate of. It will then ask for the line to display the feed rate on. The axis position display for the selected line is replaced by the feed rate display for the previously selected axis. This allows for simultaneously displaying both the position reading and rate for an axis.

When you move the axis, the rate is displayed for the axis relative to the current units. For example, if the DRO is in metric mode and you move the axis at the rate of 5mm per second, then the display will show 5.0. When you stop moving the axis, it will display 0.0.

To turn off the feed rate display, initiate this function again and the line displaying the feed rate will return to displaying the position reading for its axis.</td></tr>
<tr><td>Calculator
"CALC"</td><td>Both</td><td>Enter the reverse polish notation (RPN) calculator function. The RPN calculator has a 4-level stack and supports basic mathematical functions and a number of special operations.

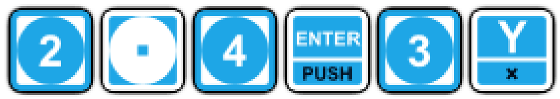
<table><tr><th>Key</th><th>Function</th></tr></table></td></tr>
</table>

FUNC EXIT	**Exit** Exit the calculator and return to normal DRO function. When the operation menu is displayed, this exits operation selection and returns to normal calculator function.
X +	**Addition (+)** Add the current and penultimate stack values together and push the result back on the stack.
ZERO -	**Subtraction (-)** Subtract the current stack value from the penultimate stack value and push the result back on the stack.
Y ×	**Multiply (*)** Multiply the current and penultimate stack values together and push the result back on the stack.
ZERO ÷	**Divide (/)** Divide the penultimate stack value (dividend) from the current stack value (divisor) and push the result back on the stack.
Z OP PRV	**Operation Previous** Move to the previous operation in the list (see below).
ZERO OP NXT	**Operation Next** Move to the next operation in the list (see below).
+/-	**Sign (+/-)** Toggle the sign for the current stack value making a positive number negative and vice versa.
FUNC EXIT	**Exit** Leave calculator mode and return to DRO processing.
INCH MM SWAP	**Swap** Exchange the values for the current and penultimate positions in the stack.
ABS INCR CLEAR	**Clear** Erase all stack values and set them to zero.
CLEAR POP	**Pop** Remove the current stack value and shift the stack down.
ENTER PUSH	**Push** Push the current value onto the stack. The next number entered will shift the stack up and

	start a new current value. When the operation menu is active, this serves as a confirmation to execute the displayed operation

Example:

To divide 2.4 by 3, enter the following keys:



The result of 0.8 is now the current value on the stack.

Operations

The calculator supports a number of mathematical operations in addition to the basics of addition, subtraction, multiplication, and division. The operations are accessed with the operation previous (OP PRV) and operation next (OP NXT) keys. The first press of either key will recall the last used operation. The next press will either move backwards or forwards through the operations list summarized below:

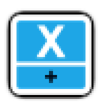
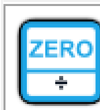
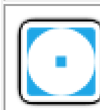
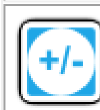
Operation	Display	Description
Memory In	M IN	Store into a memory location 0-9. The operation prompts you to enter a location 0-9 and will store the value on the bottom of the stack to that memory location replacing any previous value.
Memory Out	M OUT	Read out from a memory location 0-9. The operation prompts you to enter a location 0-9 and will read the value at that location and insert it into the bottom of the stack.
Sine	SIN	This operation calculates the geometric sine of the angle on the bottom of the stack expressed in degrees and replaces it with the result.

		Cosine	COS	This operation calculates the geometric cosine of the angle on the bottom of the stack expressed in degrees and replaces it with the result.
		Tangent	TAN	This operation calculates the geometric tangent of the angle on the bottom of the stack expressed in degrees and replaces it with the result.
		Square Root	SQRT	Calculates the square root of the value on the bottom of the stack and replaces it with the result.
		Power Of	POW	Takes the penultimate value on the stack and raises it to the power of the value on the bottom of the stack. Both values are replaced with the result.
		Axis Preset	PRESET	Perform an incremental preset on an axis with the value on the bottom of the stack. The operation will ask for an axis which you must select by pressing the preset key corresponding to the axis you want to set. The value on the bottom of the stack is NOT removed.
		Axis Reading	AXIS	Get the current incremental position for an axis and insert it onto the bottom of the stack. The operation will ask for an axis which you must select by pressing the preset key corresponding to the axis you want to get the position for.
		Pi	PI	Inserts the constant Pi (3.14159...) onto the bottom of the stack.
		Length	LENGTH	Calculate the hypotenuse of a right-angled triangle with the Pythagorean theorem { c = sqrt(a^2 + b^2) } given the two other sides. The two sides are taken from the two values on the

		bottom of the stack and they are replaced by the result.
Fraction	FRAC	Display the fractional power-of-2 equivalent of a decimal value along with the amount of error. The top line shows the whole value on the left and the numerator on the right. The middle line shows the denominator. The bottom line shows the decimal amount of error in the fraction. Press ENTER to clear the display.

Speed Keys

The operations also have speed keys to allow you to access them more quickly rather than scrolling through the list. After the operation menu is displayed, press one of the following keys to immediately jump to and execute the function.

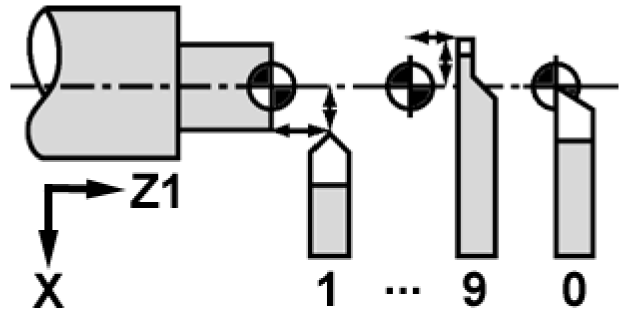
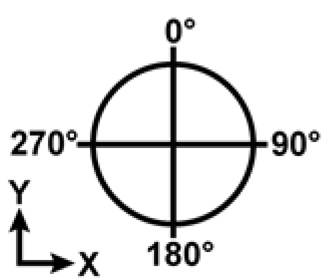
Speed Key	Operation
	Memory In
	Memory Out
	Axis Preset
	Axis Reading
	Fraction
	Length

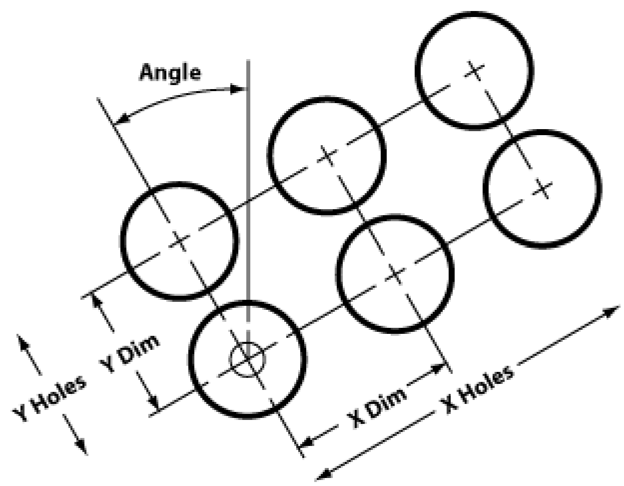
		3	Pi
		4	Square Root
		5	Power Of
		7	Sin
		8	Cosine
		9	Tangent

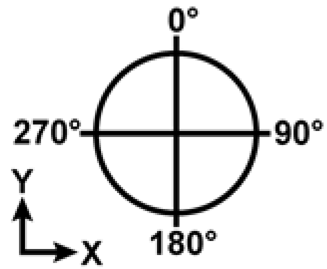
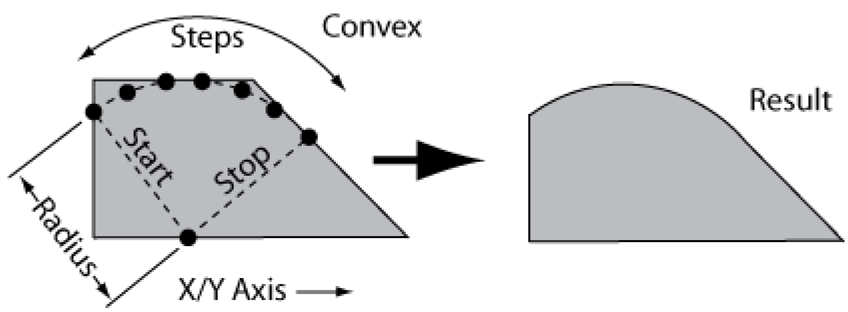
5.4 Define Menu

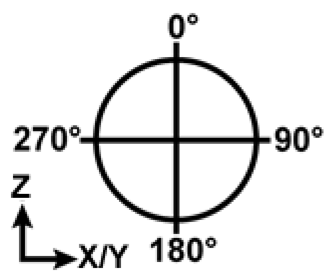
Note that the define menu is displayed on screen as "dEFINE". The on-screen display for each menu item is shown underneath the menu name.

Item	Machine	Description
Tool Offset "tooL"	Mill	This function defines a tool offset for cutting edge compensation. The tool offsets are defined in terms of the tool diameter and the Z axis offset. The first screen will prompt you to enter the tool offset number with the numeric keypad. After pressing the number, you are prompted to enter the tool diameter. When entering the offsets, press the ENTER key to complete the offset or the CLEAR to make corrections or exit. Next, you will be prompted to enter the Z offset. After entering the Z offset, the tool offset is stored in the DRO.
Tool Offset "tooL"	Lathe	This function defines a tool offsets for lathe operations. The tool offsets are defined as offsets of the X and Z1 axes. The following figure shows a visual representation.

		 The first screen will prompt you to enter the tool offset number with the numeric keypad. After pressing the number, you will be prompted to enter the X offset. When entering the offsets, press the ENTER key to complete the offset or the CLEAR key to make corrections or exit. Next, you will be prompted to enter the Z1 offset. After entering the Z1 offset, the tool offset is stored in the DRO.
Bolthole "boLtho"	Mill	This function defines a new bolthole pattern for use with a mill. When this function is selected, the first screen will prompt you for the number of the bolthole memory position. Memory positions allow the specification of multiple sets of bolthole patterns and are recalled when the bolthole function is used. Enter the memory position number and press the ENTER key. Next, you are prompted to enter the total number of holes in the bolthole pattern. Enter the number and press the ENTER key. After entering the number of holes, the next prompt will be for the radius of the bolthole circle. Enter the radius and press the ENTER key. Next, the starting and ending angles of the bolthole circle are entered. The angles are entered in degrees and with respect to the following diagram when looking at the bolt-hole circle from the top:  The valid angle ranges are from 0.0 to 359.9 degrees. If the same

		angle is entered for both the starting and ending angles, a full circle bolt-hole pattern is created. Press the ENTER key to enter each angle. After entering the end angle, the bolt-hole pattern is stored in the DRO at the given memory position.
Grid "GrId"	Mill	The grid function is used to define a series of equally spaced points along the X and Y axes that are oriented on an angle. When this function is selected, the first screen will prompt you for the number of the grid memory position. Memory positions allow the specification of multiple sets of grid points and are recalled when the grid function is used. Enter the memory position number and press the ENTER key. The diagram below shows the parameters that define a grid. Upon executing this function, you are prompted for each parameter. Enter the desired value via the numeric keypad and press the ENTER key to proceed on to the next parameter until all of them are defined.  The angle of the grid is centered on the lower left hole and is expressed in terms of the angular directions in the diagram below.

		
Radius "rAdIUS"	Mill	The radius function is used to define a convex or concave radius that is cut step by step. When this function is selected, the first screen will prompt you for the number of the radius memory position. Memory positions allow the specification of multiple sets of radii and are recalled when the radius function is used. Enter the memory position number and press the ENTER key. There are a number of parameters that define how the radius is cut. Each of the parameters are summarized in the figure below and described in more detail in the text that follows.  The first parameter to enter is the radius length. This defines the size of the arc for the cut. Enter the radius via the numeric keypad and press the ENTER key. For all parameters, press the ENTER key to confirm and move on to the next parameter. The second parameter is the radius shape, either concave or convex. A convex radius bulges out from its center point while a concave radius bulges in. Press the Z PRESET key to toggle between the two shapes. The next parameter is the second axis of the radius and defines the plane of the radius. The first axis is implicitly always the Z axis. If the X axis is selected, then stepping for the radius will proceed on the X and Z axes with the direction of the cuts occurring on the Y axis direction (front and back). If the Y axis is selected, then stepping for the radius will proceed on the Y and Z axes with the direction of the cuts occurring on the X axis

		direction (left and right). Press the Z PRESET key to toggle between the two axes. The steps parameter is next and defines the number of steps to cut along the radius. The more steps the smoother the radius but also the more cutting operations that you must perform. Enter the desired number between 2 and 99. The start parameter defines the starting angle for the radius. Look at the figure below for a definition of the angles. For a convex radius, the angle must be on the top angular hemisphere between 270 and 90 degrees. For a concave radius, the angle must be on the bottom angular hemisphere between 90 and 270 degrees.  The ending angle is the next parameter and it defines the stopping point on the radius. The radius cut always proceeds in a clockwise direction beginning at the starting angle so the ending angle must be in a clockwise direction from the starting angle and also be in the correct angular hemisphere. For example, for a convex radius with a starting angle of 0 degrees, 270 degrees is an invalid ending angle since it is counter-clockwise from the starting angle in the top angular hemisphere.

5.5 Setup Menu

The setup menu allows you to change the configurable parameters for each DRO object. When you select an object type from the menu, OpenDRO switches the display to show the object instance in the top line, the parameter name for that object on the middle line, and the parameter's current setting on the bottom line. For objects with multiple instances, like scales and axes, press the X PRESET key to cycle forward the available instance numbers and press the X ZERO key to cycle back. Press the Y PRESET key to move to the next parameter and press the Y ZERO key to move to the previous parameter. To change a parameter, press the Z PRESET or Z ZERO key which will, depending on the parameter, either rotate through the allowable settings or will prompt you to enter a numeric value. After changing a parameter, you can cycle through additional parameters to change for that object instance. You can also cycle through additional object instances.

To permanently store your parameter changes in the DRO's non-volatile memory, press the ENTER key. To continue without permanently storing your changes, press the CLEAR key. If you want to return to the setting stored in non-volatile memory, then reset the DRO.

Note that the setup menu is displayed on screen as "SEtUP". The on-screen display for each object and parameter is shown underneath their names.

<table>
<tr><th>Object</th><th>Parameters</th></tr>
<tr><td>Machine
"MACh"</td><td>The following setup parameters apply to the current machine. You can change the current machine in the system menu.
<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>Type
"tyPE"</td><td>Set the machine type - lathe or mill</td></tr>
<tr><td>Tach
"tACh"</td><td>Set the machine's tachometer - tach 1 or tach 2.

DPU-550: The tach 2 header is dual purpose and is also used to output the scale clock for 21 bit protocol scales. If select tach 2 as a tachometer on any machine, then the scale clock output is disabled and you you cannot use 21 bit protocol scales with the DPU-550 unless an external circuit is used to provide the clock.</td></tr>
<tr><td>Tach
Divisor
"tACdIV"</td><td>Divide the tachometer signal measurement by this divisor before display. This feature is useful if the tachometer sensor sends more than one pulse per revolution such as when detecting the movement of a gear. Set this to the number of pulses sent per revolution. In most cases, this can be left at the default of one.</td></tr>
<tr><td>Edge
"EdGE"</td><td>Set the machine's edge finder - none, edge 1, or edge 2

DPU-550: You can also select tach 1 or tach 2 to mimic the DRO-350 auxiliary port where the edge and tachometer functions are shared on the same port.</td></tr>
<tr><td>Display
"dISPLy"</td><td>Set what is shown by default on the axis displays:
Reading - The normal scale reading in either inches or mm.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>Counts - The raw counts from the scale before conversion. This is useful for troubleshooting the scales.
Jitter - For Chinese scales, this displays the standard deviation of the samples in the filter window. You can use this to display a quantitative measurement of the scale jitter for troubleshooting noise problems.</td></tr>
<tr><td>Warning
"WArN"</td><td>Set the near-zero warning threshold. When an axis approaches a zero within a positive or negative distance from this value, then the display will flash and the beeper, if configured, will sound. Once a near-zero warning is triggered, then you must travel 10% farther away than this value to rearm the near-zero warning. This prevents repeated triggering when working around the near-zero threshold.</td></tr>
<tr><td>Beeper
"bEEPEr"</td><td>Turn the beeper on or off. You must turn the beeper on before any other function that uses the beeper will work.

DPU-550: Sets the tachometer port the beeper is connected to since it must be shared.</td></tr>
<tr><td>Auto Off
"AUtoFF"</td><td>The number of minutes to wait before automatically turning the DRO off when it is idle. Setting this to zero disables the auto off feature and keeps the DRO on at all times.</td></tr>
<tr><td>Auto On
"AutooN"</td><td>The amount of axis movement for the DRO to automatically power on. When an axis is moved by this amount, the DRO will turn on or will reset the auto off timer if already on.</td></tr>
<tr><td>Edge Beep
"EdGbEP"</td><td>Turn on to enable a beep when the electronic edge finder makes contact during a zero operation. Note that the beeper must be turned on in the machine setup before this feature will function.</td></tr>
<tr><td>Line
"LINE"</td><td>For each display line on the DRO, set the axis that is displayed on it or turn off the line entirely.

<table><tr><td>LINE1</td><td>The top X display line on the DRO.</td></tr><tr><td>LINE2</td><td>The middle Y display line on the DRO.</td></tr><tr><td>LINE3</td><td>The bottom Z display line on the DRO.</td></tr><tr><td>LINE4</td><td>The top W display line on the LCD-200.</td></tr></table></td></tr>
</table>

LINE5	The bottom C display line on the LCD-200.

Axis "AXIS"

The axis numbers are identified as follows:

	Mill	Lathe
AXIS1	X	X
AXIS2	Y	Z1
AXIS3	Z	Z2
AXIS4	W	W
AXIS5	C	C

For each axis on the DRO, you can set the following parameters:

Parameter	Description
Scale "SCALE"	Set the scale that is the primary input for this axis.
Summing "SUMNG"	Sum a second scale into this axis.
Reverse "rEVErS"	Reverse the polarity of the axis. When enabled, this makes all positive values negative and all negative values positive and effectively moves the directional conventions of the origin. The default with all reverse settings turned off is positive X to the right, positive Y to the rear, and positive Z upwards. The DRO will not function properly if you do not set the scale reverse setting properly when installing the scales (see below in scale setup).
Precision "PrECSN"	Set the precision value to high, medium, low, or fractional. The high, medium, and low settings control the number of displayed digits after the decimal point and depend on the current measurement units. For a half digit, the DRO lights the right-most decimal point to represent the half value. Also, if a configured precision cannot fit on the display, then the DRO automatically changes the precision to the next lower value until it does fit.

	Imperial	Metric

<table>
<tr><td></td><td>
<table>
<tr><td></td><td>
<table>
<tr><td>High</td><td>4</td><td>3</td></tr>
<tr><td>Medium</td><td>3 1/2</td><td>2 1/2</td></tr>
<tr><td>Low</td><td>3</td><td>2</td></tr>
</table>

When the precision is set to fractional, the DRO will display the reading in 64ths fractions. On the main DRO display, this is represented as "A.B.C" where A is the integral number, B is the numerator, and C is the denominator. On the LCD-200 display, the reading is shown as "A B/C".
</td></tr>
<tr><td>LEC
"LEC"</td><td>Linear error correction. Enter the LEC factor in parts per million (PPM) and the reading for the axis is automatically adjusted by that amount. 1 PPM = 0.0000001.</td></tr>
</table>
</td></tr>
<tr><td>Scale
"SCALE"</td><td>
Scale numbers 1 through 5 correspond to the scale numbers printed on the DRO printed circuit board. For each scale input on the DRO, you can set the following parameters:
<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>Type
"tyPE"</td><td>Set the protocol type for the scale input to one of:

Auto - Automatically detect the Chinese scale type and set the CPI accordingly
24BIT - The original Chinese scales that are a knock-off of the Sylvac scales.
21BIT - Scales from iGaging that use the same Sylvac technology.
BCD7 - A less common format sometimes seen in calipers.
BIN6 - Format typically seen on newer Harbor Freight calipers.
Quadrature - Glass scales, linear and rotary encoders, or any other measuring device that sends two signals that are 90 degrees out of phase from each other.

</td></tr>
<tr><td>CPI
"CPI"</td><td>Set the counts per inch for the scale input:

24BIT - Use 20480 CPI
21BIT - Use 2560 CPI
BCD7 - Use 2540 CPI
BIN6 - Use 2540 CPI
Quadrature - depends on the scale so check with your

</td></tr>
</table>
</td></tr>
</table>

	Parameter	Description
		manufacturer If the scale type is Auto, then this setting is automatically changed for the detected scale type except for quadrature scales.
	Invert "INVErt"	This inverts the scale counts which reverses the direction of travel for the scale. You should configure this only once when you initially install your scales to insure that the direction of positive travel is correct with respect to the axis when the axis reverse settings are all turned off (see above in the axis setup). If you later want to change the direction of travel for an axis, change the axis reverse setting and leave this setting as is. This assures that the DRO knows the absolute directions of front, rear, left, and right for features like edge compensation.
	Filter "FILtEr"	Configure an adaptive moving average filter for a non-quadrature scale. Chinese scales are very sensitive and can suffer from display jitter due to electrical noise. Display jitter is where the least significant digit can rapidly change its value on the display. To work around this problem, you can configure an adaptive moving average filter that will average several readings together the number of which is dynamically determined based on the amount of movement. The filter will not apply to quadrature scales even if configured.
Display "dISPLy"	**Parameter**	**Description**
	Intensity "INtENS"	Set the intensity of the LED display in steps of 20%.
Keyboard "kEyPAd"	**Parameter**	**Description**
	Key Beep "bEEP"	This parameter turns key beep on or off. When key beep is on, you will hear a brief "chirp" sound after every key press. Note that the beeper must be turned on in the machine setup before this feature will function.

5.6 System Menu

Note that the system menu is displayed on screen as "SyS". The on-screen display for each menu item is shown underneath the menu name.

Item	Description
Off "oFF"	Power off the DRO. When powered off, the processor still monitors the scale positions and will wake up if a key is pressed or, if auto off is turned on, after a scale moves. To completely power down the DRO, unplug the power supply.
Machine "MACh"	Select the current machine as defined in the setup.
Send Fast "SNdFSt"	Send the fast mode sequence to all attached Chinese scales.
Version "VEr"	Display the OpenDRO software version and the hardware identifier.
Update "UPdAtE"	Erase the OpenDRO program and prepare the DRO to download a new software update. (DRO-550 only)
Save "SAVE"	Save the DRO configuration onto the SD card with a file named "config.dro". (DRO-550 with SD card only)
Load "LoAd"	Restore the DRO configuration from the SD card with a file named "config.dro". (DRO-550 with SD card only)
Erase "ErASE"	Erase the contents of the non-volatile memory. This will erase all definitions, positions, setup, and other saved information and return the DRO to a factory default state. It does not erase the OpenDRO program. You should reboot the system after executing this command.

6 Scales

OpenDRO supports both Chinese and quadrature scales.

6.1 Chinese Scales

Chinese scales refer to any of a large number of linear scales and calipers that are sourced from China and based on capacitive sensing technology originally developed by Sylvac. These scales come in a huge variety of lengths and styles. Chinese scales usually have a four-pin output port on the side of the scale that sends the scale position data to the DRO. There are four different protocols used for sending this data:

- **24BIT** - Two 24-bit, binary numbers are sent with the first representing the absolute position and the second representing the relative position. The two numbers are counts with a resolution of 20480 counts per inch.
- **21BIT** - A single 21-bit, binary number is sent representing the absolute position. The number has a resolution of 2560 counts per inch. The 21BIT scales differ in that the DRO must provide a clock to read out the data rather than the scale providing the clock like other Chinese scale types. See the 21 bit protocol information at the ShumaTech web site for information on how to connect a 21 bit protocol scale.
- **BIN6** - A single 20-bit, binary number followed by a 4-bit flag field. The 20-bit number is the count with a resolution of 2540 counts per inch when the scale is in metric mode.
- **BCD7** - A single 24-bit, BCD number followed by a 4-bit flag field. The BCD number represents the seven digits shown on the scale display.

In addition, the 24BIT and BCD7 scales support a "fast" mode where they send the data to the DRO at a much higher rate than normal. The 21BIT and BIN6 scales do not support that feature.

The metal backs of Chinese scales are connected to one of the battery terminals. For 24BIT and BCD7 scales, the metal backs are connected to the positive battery terminal. For 21BIT and BIN6 scales, the scale beams are connected to the negative battery terminal. If you mix both types on one DRO, then you must insulate them to avoid shorting the DRO power supply through the machine.

WARNING!!! Do not mix 21BIT or BIN6 scales with 24BIT or BCD7 scales without electrically isolating one scale type or the other.

Except for iGaging-branded 21BIT scales, it is difficult to determine which protocol a Chinese scale uses just by looking at it. You can differentiate between BIN6 and a 24BIT/BCD7 by using a multimeter to determine which side of the battery the scale back is connected to. Differentiating a BCD7 from a 24BIT is more difficult and is best done with an oscilloscope. Note, however, that 24BIT scales are much more common than BCD7 scales. Most of the time, you can set the scale type to "Auto" and OpenDRO will automatically detect and configure the scale type.

6.2 Quadrature Scales

Quadrature scales are linear or rotary encoders that output two signals that are 90 degrees out of phase with each other and includes most glass scales encountered on machining equipment. OpenDRO supports any quadrature encoder that outputs a signal that is electrically compatible with the hardware it is running on. The DPU/DRO-550 supports quadrature encoders that output a digital, 5V output. A partial list of scale manufactures and their compatibility with the DPU/DRO-550 is shown in the following table.

Manufacturer	Support
Acu-rite	Some models
Anilam	Yes

Ditron	Yes
DRO Pros	Yes
Easson	Yes
Futaba	No
Heidenhain	Some models
Jenix	Yes
Meister	Yes
Mitutoyo	Some models
mTech	Yes
Newall	Some models
QMS	Yes
RSF	Some models
Sargon	Yes
Sino	Yes
Sinpo	Yes
Sokki	No
Sony	No
TDS	Yes
Teledyne Gurley	No
Uniq	Yes
US Digital	Yes

7 Auxiliary Devices

OpenDRO supports several auxiliary devices that you can connect to your DRO. These devices include tachometers, electronic edge finders, and SD card readers for automation.

7.1 Tachometer

The tachometer function is used to display the spindle speed of a mill or lathe. A small probe that optically measures the rotational speed is mounted to the machine. The probe sends a 5V pulse to the DRO through the tachometer connector for every rotation of the spindle. The DRO measures the period between these pulses and converts it to either RPM or SFM for display on the Z/Z2 axis. The tachometer display is updated once per second.

7.2 Electronic Edge Finder

Absolute zeros can be set on the DRO with an electronic edge finder by contacting the work piece with the edge finder while it is mounted in the spindle. Many commercially available edge finders can be adapted for use with the DRO. A circuit inside the edge finder sends a 5V pulse to the DRO through the edge finder connector. To set the absolute zero for an axis with the electronic edge finder, set the DRO to absolute mode and press the ZERO key once for that axis. When the edge finder contacts the work piece, the axis is zeroed at the current position. To compensate for the diameter of the edge finder, use a tool offset with the cutting edge compensation feature.

7.3 SD Card Reader

An SD card reader is available for the DRO-550 if you want to add the ability to read SD memory cards but do not want the LCD-200 and two extra axis displays. The SD card reader mounts inside the DRO-550 case and you only have to cut a small slot in the case to insert the card into.

8 Automation

OpenDRO can interface with a PC for automating its operation and reporting information. The PC interface works over both the USB port and the RS-232 port. When connected over the USB port, the DRO appears to the PC as a virtual COM port. A Windows INF file is available at OpenDRO sourceforge site to install the proper driver for the virtual COM port. Both the USB and RS-232 interfaces use a baud rate of 57600 and communication parameters at 8-N-1. You can use Windows HyperTerminal, TeraTerm, or any other serial communication software to enter commands.

8.1 Lua Language

The OpenDRO automation interface uses the Lua programming language. Lua is a lightweight and simple scripting language that is commonly embedded into larger applications due to the ease of integration of its C API. The language is easy to learn and very flexible. The rest of this document will assume some minimal knowledge of Lua so if this is your first exposure to it, I would suggest looking at the on-line books Programming in Lua and the Lua Reference Manual.

In addition to the core language operators, Lua defines a set of standard libraries that incorporates commonly used functionality, a list of the standard libraries available in OpenDRO are shown below. Reference the Lua documentation for exact usage and syntax.

- **base** - The base set of global Lua functions.
  - pairs ipairs newproxy assert collectgarbage dofile error gcinfo getfenv getmetatable loadfile load loadstring next pcall print rawequal rawget rawset select setfenv setmetatable tonumber tostring type unpack xpcall
- **io** - Standard file input and output functions.
  - stderr stdout stdin close flush input lines open output popen read tmpfile type write
- **string** - Text string operations.

  - byte char dump find format gfind gmatch gsub len lower match rep reverse sub upper
- **math** - Floating point mathematical operations.
  - abs acos asin atan2 atan ceil cosh cos deg exp floor fmod mod frexp ldexp log10 log max min modf pow rad random randomseed sinh sin sqrt tanh tan pi huge
- **table** - Lua table manipulation.
  - concat foreach foreachi getn maxn insert remove setn sort
- **load** - Functions to aid in modular programming.
  - module require package

The Lua source code in OpenDRO incorporates two important patches to Lua that conserve memory in OpenDRO, the LTR patch and the EGC patch. The LTR (Lua Tiny RAM) patch saves memory by making most standard Lua tables read-only so they can be store in Flash rather than RAM. The EGC (Emergency Garbage Collector) is a patch that aggressively frees memory with the garbage collector when memory is low. These patches are important because RAM is very scarce in OpenDRO due to the limited amounts in the Atmel ARM7 processors (32KB in the DRO-550 and 64KB in the DPU-550).

One should be realistic with the capabilities of the automation interface due to the small amounts of free RAM available. The interface is designed to write simple scripts to assist the machinist with common tasks or to customize the behavior of the DRO. It is not designed to write large blocks of functionality. That is much more efficient and compact if it is written in C and added to the main OpenDRO source code. The examples below will demonstrate some techniques to minimize memory usage when using the automation interface such as loading scripts from the SD card and setting variables to nil when they are no longer needed.

8.2 Lua Shell

The automation interface incorporates an advanced shell with many of the features commonly found on Unix and Windows command line interfaces. This includes editing control keys, command history, and tab completion. The shell displays an 'OpenDRO>' prompt to let you know that it is ready to accept a command. If you enter an incomplete command, then the shell will display 'OpenDRO>>' to let you know more input is required. For example:

```
OpenDRO> print(
OpenDRO>> 'hello'
OpenDRO>> )
hello
OpenDRO>
```

8.2.1 Editing

The command line editing operations are summarized below. If your terminal software is configured for VT100 operation, which most are, then the keys on your keyboard shown below should work. You can also use the control key combination shown to achieve the same result.

Key	Control	Description
Left Arrow	Ctrl-B	Move the cursor one position to the left.
Right Arrow	Ctrl-F	Move the cursor one position to the right.
Home	Ctrl-A	Move the cursor to the beginning of the line.
End	Ctrl-E	Move the cursor to the end of the line.
Backspace	Ctrl-H	Delete one character to the left of the cursor.
-	Ctrl-K	Delete everything from the cursor to the end of the line.
-	Ctrl-U	Delete the entire line.

8.2.2 History

The shell remembers the last ten commands you execute so you can easily recall them. To do this, press the keys shown below. When reaching the top or bottom of the command history, the shell will beep.

Key	Control	Description
Up Arrow	Ctrl-P	Recall the previous command line.
Down Arrow	Ctrl-N	Recall the next command line.

8.2.3 Completion

The tab key, or Ctrl-I, will perform command completion at the current cursor position. This fills in the command line in with the remainder of the name typed up to that point or a list of possible completions if the name typed is ambiguous. If there is no possible completion, then the shell will beep. For example, if you type 's' and then press tab, you will see a list of possible completions since 's' is ambiguous:

```
OpenDRO> s
string system scale sd select setfenv setmetatable
OpenDRO> s
```

If you then type a 'c' after the 's' and press tab again, then the completion of 'scale' is filled in:

```
OpenDRO> scale
```

Now type a '.' for access to the scale object and then press tab again. You will see a list of the possible scale methods:

```
OpenDRO> scale.
count cpi show
OpenDRO> scale.
```

Now type an 's', press tab, and 'show' is filled in:

```
OpenDRO> scale.show
```

Add the scale number '1' inside a set of parentheses and press enter to execute the command:

```
OpenDRO> scale.show(1)
Scale        : 0
Config Type  : auto
Active Type  : 21bit
Auto Weight  : 0
Invert       : no
Count        : 1162
CPI          : 2560
Filter       : yes
AMA Count    : 1162.285476
```

8.3 Lua Extensions

OpenDRO extends Lua with a number of built-in objectsthat make it easy to get status information and control the behavior of OpenDRO. These extensions are summarized below.

8.3.1 User Functions

OpenDRO supports up to ten user functions that are started from the DRO keypad. The user functions are identified by functions at global scope named 'user0' through 'user9'. After creating a user function, you run it by pressing the FUNC key twice followed by the number. For example, suppose you create a user function to beep for 500ms. Enter the function in the shell as follows:

```
OpenDRO> function user1() machine.beep(500) end
```

Now press FUNC-FUNC-1 and the DRO will beep (assuming you have the beeper on). Note that on the second press of the FUNC key, the OpenDRO will prompt for the user function to execute. If after making a selection the function has a run-time error, OpenDRO will display an error message on the display which you can clear by pressing any key. You can then go back and correct your user function or try running it from the shell by typing the function name and empty parentheses (i.e. 'user1()').

8.3.2 Auto-Execute Script

OpenDRO can automatically execute a Lua script on boot up. This script is stored in the non-volatile memory on the DRO and is used to initialize user functions, load programs off of the SD card, or to do anything else you like. On the DPU-550, this is the only place to store permanent Lua code. For the DRO-550, one can execute Lua scripts from the SD card in addition. The auto-execute script is written by the luash.execwr() function and read by the luash.execrd() function. See the OpenDRO object reference below for more information.

8.3.3 OpenDRO Types

The data types listed below are used by the OpenDRO object reference that follows this section. These are not meant to represent official Lua data types and are simply used to clarify the expected type and range for the parameters passed to and returned from the OpenDRO objects.

- FLOAT - A decimal floating point number. For example, -1.23 or 5.78555.
- INT - An integeral positive or negative number.
- BOOL - A boolean value. In Lua, 'nil' or 'false' are logically false, 'true' or any other value is logically false.
- STRING - A textual string encapsulated inside of single or double quotes. Most strings used by OpenDRO are not case sensitive.
- TABLE - A Lua table which can represent an indexed array or an arbitrary set of data.
- FILE - A string identify the complete path to a file on the SD card. Either forward slash '/' or backward slash '\' may be used as directory separators. The file path is not case sensitive.
- AXIS - A string or number identifying an axis. Mill axes are identified by 'X', 'Y', 'Z', 'W', or 'C' or by the numbers 1 through 5, respectively. Lathe axes are identified by 'X', 'Z1', Z2', 'W', or 'C'.
- LINE - A number from 1 through 3, or 1 through 5 if an LCD-200 is attached, that identifies a display line. Lines 1 through 3 are displays X, Y, and Z, respectively, on the main DRO display. Numbers 4 and 5 identify display lines W and C, respectively, on the LCD-200 display.
- MENU - A number from 1 through 3 that identifies the top, middle, or bottom menu line on the display, respectively.
- SCALE - A number from 1 through 5 that identifies a scale.

8.3.4 OpenDRO Objects

This section lists the OpenDRO objects added to Lua that enable you to control and report various aspects of its operation. The tables following each object list the functions available inside of the object.

axis

Function	Return	Description
read(AXIS)	FLOAT	Returns the current axis reading.
preseta(AXIS,FLOAT)	nil	Set an incremental preset at an absolute position.
preseti(AXIS,FLOAT)	nil	Set an incremental preset at an incremental position.
zeroa(AXIS,FLOAT)	nil	Set an absolute zero at an absolute position.
zeroi(AXIS)	nil	Set an incremental zero at the current position.
save()	nil	Save the current axis settings to the non-volatile configuration.
show()	nil	Show the axis state for troubleshooting.

config

Function	Return	Description
save(FILE)	nil	Save the entire OpenDRO configuration to the SD card. (DRO-550 only)
load(FILE)	nil	Load the entire OpenDRO configuration from the SD card. (DRO-550 only)

display

Function	Return	Description
readset(LINE,FLOAT,INT,BOOL)	nil	Set a floating point reading on a display line. The third integer parameter is the display precision in half digits. The fourth boolean parameter controls the display mark, which is the left-most decimal point that is used as an indicator for tool offsets and diameter mode.
readget(LINE)	STRING	Return the reading shown on the display line.
readblk(LINE)	nil	Blink the reading on the display line.
ind()	BOOL	Return a table of boolean values indicating the

	TABLE	state of the LED indicators across the top of the DRO. The indicators are in the following order starting at index 1: INCR, MM, ZERO, SET, and FUNC.
on()	nil	Turn the display on.
off()	nil	Turn the display off.
menuset(MENU,STRING)	nil	Set a text string on the menu line.
menuget(MENU)	STRING	Return the text string currently displayed on the menu line.
menuclr()	nil	Clear the entire menu display of text.
menustate()	BOOL	Return a boolean value that indicates the display state of the menu, true if the menu is currently displayed and false if it is not.
print(STRING)	nil	Display a text string across all three menu lines starting at the top. This is convenient to easily display a longer message to the user without having to call menuset() multiple times.

function

<table>
<tr><th>Function</th><th>Return</th><th>Description</th></tr>
<tr><td>run(STRING,STRING)</td><td>nil</td><td>Run a function from the menu where the first string identifies the group and the second identifies the function within the group. The complete list of group and function names is shown below.
<table>
<tr><th>'positn'</th><th>'status'</th><th>'define'</th><th>'setup'</th><th>'sys'</th></tr>
<tr><td>'tool'</td><td>'tach'</td><td>'tool'</td><td>'mach'</td><td>'off'</td></tr>
<tr><td>'c-line'</td><td>'feed'</td><td>'boltho'</td><td>'axis'</td><td>'mach'</td></tr>
<tr><td>'boltho'</td><td>'calc'</td><td>'grid'</td><td>'scale'</td><td>'sndfst'</td></tr>
<tr><td>'grid'</td><td></td><td>'radius'</td><td>'disply'</td><td>'ver'</td></tr>
<tr><td>'radius'</td><td></td><td></td><td>'keypad'</td><td>'update'</td></tr>
<tr><td>'wspace'</td><td></td><td></td><td></td><td>'save'</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td></td><td><table><tr><td>'dia'</td><td></td><td></td><td></td><td>'load'</td></tr><tr><td>'compnd'</td><td></td><td></td><td></td><td>'erase'</td></tr></table></td></tr>
<tr><td>tool(INT)</td><td>nil</td><td>Select a tool to use from 1 to the maximum allowed tool number. Tool number 0 removes the current tool selection.</td></tr>
<tr><td>edge(STRING)</td><td>nil</td><td>Select an edge compensation side if a tool is in use. The string is one of 'center', 'left', 'right', 'front' or 'rear'.</td></tr>
<tr><td>diameter(AXIS)</td><td>nil</td><td>Select an axis to apply diameter mode to. If diameter mode is already applied, calling this function again will remove it regardless of the axis in the parameter.</td></tr>
<tr><td>compound(FLOAT)</td><td>nil</td><td>Set the lathe compound vectoring angle. Setting an angle of 0 disables compound vectoring.</td></tr>
<tr><td>workspace(INT)</td><td>nil</td><td>Select a workspace from 1 to the maximum number supported.</td></tr>
<tr><td>tach(LINE,BOOL)</td><td>nil</td><td>Set the tach to show on a display line. The second boolean parameter enables surface speed tach mode. Calling this function again when the tach is already displayed will remove it regardless of the parameters.</td></tr>
<tr><td>feed(LINE,AXIS)</td><td>nil</td><td>Show the feed rate for an axis on a display line. Calling this function again when a feed rate is already displayed will remove it regardless of the parameters.</td></tr>
<tr><td>save</td><td>nil</td><td>Save the current function settings to the non-volatile configuration.</td></tr>
<tr><td>show</td><td>nil</td><td>Show the function state for troubleshooting.</td></tr>
</table>

keypad

Function	Return	Description
press(STRING)	nil	Simulate the pressing of a key on the keypad. The key names are as follows: '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '+-', '.', 'func', 'mminch', 'absincr', 'clear', 'enter', 'zero1', 'preset1', 'zero2', 'preset2', 'zero3', 'preset3', 'zero4', 'preset4', 'zero5', 'preset5'
get()	STRING	Get the next key pressed on the keypad. The key names are the same as documented above. If a key press is not

		available, then the function will wait until one is available and will stop processing readings.
wait()	nil	Wait until a key is pressed. While waiting, readings are still processed. Use this before get() to retrieve a key press without stopping reading update on the display.
float(MENU)	FLOAT	Get a floating point number that is edited on the indicated menu line. Nil is returned if the user cancels entry.
number(MENU,INT, INT)	INT	Get a integral number that is edited on the indicated menu line. The second parameter is the minimum allowed value and the third parameter is the maximum allowed value. Nil is returned if the user cancels entry.
select(MENU,STRING TABLE)	INT	Select between a set of strings on the indicated menu line. The integer index of the selection, starting at one for the first string, is returned. Nil is returned if the user cancels entry.

luash

Function	Return	Description
automate(BOOL)	nil	Control automate mode. If the parameter is true, then the line editing facilities of the shell are disabled and the characters are not echoed back. This mode is meant for automated PC control such as with the VDRO-550 Windows application.
execrd()	nil	Display the auto-execute script. The auto-execute script is a Lua script stored in the non-volatile memory that is executed every time OpenDRO boots.
execwr()	nil	Write the auto-execute script. If automate mode is off, then line editing facilities may be used for each line entered. To complete editing and save the script, hold down control and press the D key.
history()	nil	Display the lua shell command history.

machine

Function	Return	Description
active(INT)	nil	Set the active machine to 1 or 2.

inch()	nil	Set the machine to inch units.
mm()	nil	Set the machine to mm units.
abs()	nil	Set the machine to absolute mode.
inc()	nil	Set the machine to incremental mode.
on()	nil	Turn the machine on.
off()	nil	Turn the machine off.
stop()	nil	Stop processing readings.
beep(INT)	nil	Beep for the given number of milliseconds.
save()	nil	Save the current function settings to the non-volatile configuration.
show()	nil	Show the machine state for troubleshooting.

mem

Function	Return	Description
rb(INT)	INT	Read an 8-bit byte from memory at the given address.
rw(INT)	INT	Read a 32-bit word from memory at the given address.
wb(INT,INT)	nil	Write an 8-bit byte to memory. The first parameter is the address and the second is the value.
ww(INT,INT)	nil	Write a 32-bit word to memory. The first parameter is the address and the second is the value.
db(INT,INT)	nil	Display a range of 8-bit byte values from memory. The first parameter is the address and the second is the number of bytes to display.
dw(INT,INT)	nil	Display a range of 32-bit word values from memory. The first parameter is the address and the second is the number of words to display.
show()	nil	Display stack, heap, and Lua memory diagnostic information.

scale

Function	Return	Description

count(SCALE)	INT	Return the current counts for a scale.
cpi(SCALE)	INT	Get the counts per inch (CPI) setup for a scale.
show(SCALE)	nil	Show the scale state for troubleshooting.

sd

Function	Return	Description
show()	nil	Show the SD card state for troubleshooting.
dir(FILE)	nil	Display the files in the given subdirectory.
mkdir(FILE)	nil	Make a new subdirectory at the given location.
rename(FILE,FILE)	nil	Rename a file from the first to second parameter.
remove(FILE)	nil	Remove the given file and its contents.
read(FILE)	nil	Display the given file.
write(FILE)	nil	Write to the given file. If automate mode is off, then line editing facilities may be used for each line entered. To complete editing and save the file, hold down control and press the D key which sends the EOT character.
hd(FILE)	nil	Display a hexadecimal dump of the given file.

system

Function	Return	Description
reset()	nil	Reset the DRO hardware.
version()	STRING	Return a string of the form 'x.y.z' for the OpenDRO version.
hardware()	STRING	Return a string for the hardware type.
show()	nil	Display all diagnostic displays for all objects.

trace

Function	Return	Description
debug()	nil	Set the trace level to debug and above.
info()	nil	Set the trace level to info and above.
warning()	nil	Set the trace level to warning and above.
error()	nil	Set the trace level to error and above.
fatal()	nil	Set the trace level to fatal only.
level()	STRING	Return a string indicating the current trace level.

8.4 Examples

Below are some examples of using the automation interface to do some useful operations.

8.4.1 User Speed Keys

An easy trick is to use a user function for your own speed key. For example, suppose you want to have a speed key to quickly jump to machine setup:

```
function user1()
function.run('setup','mach')
end
```

Now when you press FUNC-FUNC-1, OpenDRO will jump straight to the machine setup. You can do this for any function by supplying the proper group and function names in the function.run() call as documented above.

8.4.2 Auto-Execute from SD Card

OpenDRO can automatically execute a script on boot up as previously described but let's say you want this script to come from a file on the SD card named 'autoexec.lua' instead. This is easily accomplished by writing a one line auto-execute script to run the SD card file:

```
OpenDRO> luash.execwr()
Press Ctrl-D to end
> dofile('autoexec.lua')
>
OpenDRO>
```

Note that the SD card is only supported on the DRO-550. You can also extend this concept to the autoexec.lua file itself where each user function is its own script on the SD card.

autoexec.lua

```
function user1() dofile('polar.lua') end
function user2() dofile('threads.lua') end
function user3() dofile('calc.lua') end
function user4() dofile('angle.lua') end
```

An approach like this has the added benefit of conserving memory since all of the scripts reside on the SD card instead of in RAM and is strongly recommended for the DRO-550. The DPU-550 has twice the amount of RAM as the DRO-550 so its lack of SD card support is much less of an issue.

8.4.3 Polar Conversion

The following example demonstrates how to convert a polar coordinate into a Cartesian coordinate that the DRO can use.

```
display.menuset(1,'enter')
display.menuset(2,'length')
local length = keypad.float(3)
display.menuset(2,'angle')
local angle = keypad.float(3)
angle = angle * math.pi / 180
axis.preseti('x', length * math.cos(angle))
axis.preseti('y', length * math.sin(angle))
machine.inc()
```

The first two lines show a message on the main display to enter the length. The third line waits for the user to enter a floating point number on the keypad for the length. Note the 'local' keyword in front of the length variable. This keyword makes the length variable local in scope instead of global. This conserves memory since when the function is finished, the memory for the variable is freed. You should get in the habit of adding this in front of all variables. You only need to add it to the first declaration of the variable and not on subsequent use of the variable.

The next two lines prompt and wait for the angle. The angle is then converted from degrees to radians before performing the conversion to X and Y coordinates with sine and cosine. The axis.preseti() lines set incremental preset values using the X and Y coordinates while the last line switches to incremental mode to show the results.

8.4.4 Tap Drill Chart

The following example demonstrates how to show information to the user via a selection menu. In this case, we will show the tap drill sizes for five screw sizes.

```
local screws = { '2-56', '4-40', '6-32', '8-32', '10-24' }
local drills = { 0.0730, 0.960, 0.1160, 0.1440, 0.1610 }
display.menuset(1,'select')
display.menuset(2,'screw')
```

```
local index = keypad.select(3, screws);
display.menuset(1,'tap')
display.menuset(2,'drill')
display.menuset(3,tostring(drills[index]))
keypad.get()
```

The first two lines define Lua tables for the screw names and their respective drill sizes. Note the 'local' keyword as previously discussed. The script shows a select screw message before prompting for a selection. The selection works by rotating through the table given as the second parameter with the preset and zero keys. When a selection is made with the enter key, the index corresponding to the selection is returned. We then display a result message by looking up the size in the drills table with the returned index. Lastly, we wait for any key press to continue.

8.4.5 Memory Usage

Here we will demonstrate how to minimize memory usage and troubleshoot memory problems. To minimize memory usage, always follow the following guidelines:

1. Use local variables and local functions whenever possible.
2. When you must use globals, set their value to nil when you are done with them so that they are freed.
3. Store scripts on the SD card instead of in memory.
4. Use Lua tables sparingly since they use a lot of memory.

You can watch memory usage with the mem.show() command. The last line, 'Lua free', shows the amount of free memory available for Lua. Keep in mind that Lua is a garbage collected language so don't be concerned if the number jumps around a bit. You should be more interested in the trend.

```
OpenDRO> mem.show()
 :               :
Lua free      : 7687
```

Now let's define a function in memory:

```
OpenDRO> function polar()
OpenDRO>> local screws = { '2-56', '4-40', '6-32', '8-32', '10-24' }
OpenDRO>> local drills = { 0.0730, 0.960, 0.1160, 0.1440, 0.1610 }
OpenDRO>> display.menuset(1,'select')
OpenDRO>> display.menuset(2,'screw')
OpenDRO>> local index = keypad.select(3, screws);
OpenDRO>> display.menuset(1,'tap')
OpenDRO>> display.menuset(2,'drill')
OpenDRO>> display.menuset(3,tostring(drills[index]))
OpenDRO>> keypad.get()
OpenDRO>> end
OpenDRO>
```

Now, let's check the memory again:

```
OpenDRO> mem.show()
 :              :
Lua free      : 6646
```

You can see that the function is taking a little over 1000 bytes of memory because it is now a global function. We can free the memory consumed by the function by setting it to nil.

```
OpenDRO> polar=nil
OpenDRO> mem.show()
 :              :
Lua free      : 7794
```

We are now back and slightly above our starting memory usage. Again, the reason we are above is because of the effects of garbage collection. If we instead save this script on the SD card and run it with dofile(), then it would only use memory when it is running and would immediately free it when done.