## Variables

**A / A x**
get / set the variable A, default 1

**B / B x**
get / set the variable B, default 2

**C / C x**
get / set the variable C, default 3

**D / D x**
get / set the variable D, default 4

**DRUNK / DRUNK x**
changes by -1, 0, or 1 upon each read saving its state, setting will give it a new value for the next read

**DRUNK.MIN / DRUNK.MIN x**
set the lower bound for DRUNK, default 0

**DRUNK.MAX / DRUNK.MAX x**
set the upper bound for DRUNK, default 255

**DRUNK.WRAP / DRUNK.WRAP x**
should DRUNK wrap around when it reaches it's bounds, default 0

**FLIP / FLIP x**
returns inverted state (0 or 1) on each read (also settable)

**I / I x**
get / set the variable I, this variable is overwritten by L, but can be used freely outside an L loop

**O / O x**
auto-increments *after* each access, can be set, starting value 0

**O.INC / O.INC x**
how much to increment O by on each invocation, default 1

**O.MIN / O.MIN x**
the lower bound for O, default 0

**O.MAX / O.MAX x**
the upper bound for O, default 63

**O.WRAP / O.WRAP x**
should O wrap when it reaches its bounds, default 1

**T / T x**
get / set the variable T, typically used for time, default 0

**TIME / TIME x**
timer value, counts up in ms., wraps after 32s, can be set

**TIME.ACT / TIME.ACT x**
enable or disable timer counting, default 1

**X / X x**
get / set the variable X, default 0

**Y / Y x**
get / set the variable Y, default 0

**Z / Z x**
get / set the variable Z, default 0

## Hardware

**CV x / CV x y**
CV target value

**CV.OFF x / CV.OFF x y**
CV offset added to output

**CV.SET x**
Set CV value

**CV.SLEW x / CV.SLEW x y**
Get/set the CV slew time in ms

**IN**
Get the value of IN jack (0-16383)

**PARAM** PRM
Get the value of PARAM knob (0-16383)

**TR x / TR x y**
Set trigger output x to y (0-1)

**TR.POL x / TR.POL x y**
Set polarity of trigger output x to y (0-1)

**TR.TIME x / TR.TIME x y**
Set the pulse time of trigger x to y ms

**TR.TOG x**
Flip the state of trigger output x

**TR.PULSE x** TR.P
Pulse trigger output x

**MUTE x / MUTE x y**
Disable trigger input x

**STATE x**
Read the current state of input x

## Patterns

**P.N / P.N x**
get/set the pattern number for the working pattern, default 0

**P x / P x y**
get/set the value of the working pattern at index x

**PN x y / PN x y z**
get/set the value of pattern x at index y

**P.L / P.L x**
get/set pattern length of the working pattern, non-destructive to data

**PN.L x / PN.L x y**
get/set pattern length of pattern x. non-destructive to data

**P.WRAP / P.WRAP x**
when the working pattern reaches its bounds does it wrap (0/1), default 1 (enabled)

**PN.WRAP x / PN.WRAP x y**
when pattern x reaches its bounds does it wrap (0/1), default 1 (enabled)

**P.START / P.START x**
get/set the start location of the working pattern, default 0

**PN.START x / PN.START x y**
get/set the start location of pattern x, default 0

**P.END / P.END x**
get/set the end location of the working pattern, default 63

**PN.END x / PN.END x y**
get/set the end location of the pattern x, default 63

**P.I / P.I x**
get/set index position for the working pattern.

**PN.I x / PN.I x y**
get/set index position for pattern x

**P.HERE / P.HERE x**
get/set value at current index of working pattern

**PN.HERE x / PN.HERE x y**
get/set value at current index of pattern x

**P.NEXT / P.NEXT x**
increment index of working pattern then get/set value

**PN.NEXT x / PN.NEXT x y**
increment index of pattern x then get/set value

**P.PREV / P.PREV x**
decrement index of working pattern then get/set value

**PN.PREV x / PN.PREV x y**
decrement index of pattern x then get/set value

**P.INS x y**
insert value y at index x of working pattern, shift later values down, destructive to loop length

**PN.INS x y z**
insert value z at index y of pattern x, shift later values down, destructive to loop length

**P.RM x**
delete index x of working pattern, shift later values up, destructive to loop length

**PN.RM x y**
delete index y of pattern x, shift later values up, destructive to loop length

**P.PUSH x**
insert value x to the end of the working pattern (like a stack), destructive to loop length

**PN.PUSH x y**
insert value y to the end of pattern x (like a stack), destructive to loop length

**P.POP**
return and remove the value from the end of the working pattern (like a stack), destructive to loop length

**PN.POP x**
return and remove the value from the end of pattern x (like a stack), destructive to loop length

## Control flow

**IF x: ...**
if x is not zero execute command

**ELIF x: ...**
if all previous IF / ELIF fail, and x is not zero, execute command

**ELSE: ...**
if all previous IF / ELIF fail, excute command

**L x y: ...**
run the command sequentially with I values from x to y

**PROB x: ...**
potentially execute command with probability x (0-100)

**SCRIPT x**
execute script x (1-8), recursion allowed

**SCENE x**
load scene x (0-31)

**KILL**
clears stack, clears delays, cancels pulses, cancels slews

## Maths

**ADD x y** +
add x and y together

**SUB x y** -
subtract y from x

**MUL x y** *
multiply x and y together

**DIV x y** /
divide x by y

**MOD x y** %
find the remainder after division of x by y

**RAND x**
generate a random number between 0 and x inclusive

**RRAND x y**
generate a random number between x and y inclusive

**TOSS**
randomly return 0 or 1

**MIN x y**
return the minimum of x and y

**MAX x y**
return the maximum of x and y

**LIM x y z**
limit the value x to the range y to z inclusive

**WRAP x y z**
limit the value x to the range y to z inclusive, but with wrapping

**QT x y**
round x to the closest multiple of y (quantise)

**AVG x y**
the average of x and y

**EQ x y** ==
does x equal y

**NE x y** != XOR
x is not equal to y

**LT x y** <
x is less than y

**GT x y** >
x is greater than y

**LTE x y** <=
x is less than or equal to y

**GTE x y** >=
x is greater than or equal to y

**EZ x** !
x is 0, equivalent to logical NOT

**NZ x**
x is not 0

**LSH x y** <<
left shift x by y bits, in effect multiply by 2 to the power of x

**RSH x y** >>
right shift x by y bits, in effect divide by 2 to the power of x

**ABS x**
absolute value of x

**AND x y** &&
logical AND of x and y

**OR x y** ||
logical OR of x and y

**JI x y**
just intonation helper, precision ratio divider normalised to 1V

**SCALE a b x y i**
scale i from range a to b to range x to y, i.e. i * (y - x) / (b - a)

**ER f l i**
Euclidean rhythm, f is fill (1-32), l is length (1-32) and i is step (any value), returns 0 or 1

**N x**
converts an equal temperament note number to a value usable by the CV outputs (x in the range -127 to 127)

**V x**
converts a voltage to a value usable by the CV outputs (x between 0 and 10)

**VV x**
converts a voltage to a value usable by the CV outputs (x between `0` and `1000`, `100` represents 1V)

**EXP x**
exponentiation table lookup. `0-16383` range (V `0-10`)

## Metronome

**M / M x**
get/set metronome interval to x (in ms), default `1000`, minimum value `25`

**M! / M! x**
get/set metronome to experimental interval x (in ms), minimum value 2

**M.ACT / M.ACT x**
get/set metronome activation to x (`0/1`), default 1 (enabled)

**M.RESET**
hard reset metronome count without triggering

## Delay

**DEL x: ...**
Delay command by x ms

**DEL.CLR**
Clear the delay buffer

## Stack

**S: ...**
Place a command onto the stack

**S.CLR**
Clear all entries in the stack

**S.ALL**
Execute all entries in the stack

**S.POP**
Execute the most recent entry

**S.L**
Get the length of the stack

## Queue

**Q / Q x**
Modify the queue entries

**Q.N / Q.N x**
The queue length

**Q.AVG / Q.AVG x**
Return the average of the queue

## Ansible

**KR.PRE / KR.PRE x**
return current preset / load preset x

**KR.PERIOD / KR.PERIOD x**
get/set internal clock period

**KR.PAT / KR.PAT x**
get/set current pattern

**KR.SCALE / KR.SCALE x**
get/set current scale

**KR.POS x y / KR.POS x y z**
get/set position z for track z, parameter y

**KR.L.ST x y / KR.L.ST x y z**
get loop start for track x, parameter y / set to z

**KR.L.LEN x y / KR.L.LEN x y z**
get length of track x, parameter y / set to z

**KR.RES x y**
reset position to loop start for track x, parameter y

**ME.PRE / ME.PRE x**
return current preset / load preset x

**ME.SCALE / ME.SCALE x**
get/set current scale

**ME.PERIOD / ME.PERIOD x**
get/set internal clock period

**ME.STOP x**
stop channel x (0 = all)

**ME.RES x**
reset channel x (0 = all), also used as "start"

**LV.PRE / LV.PRE x**
return current preset / load preset x

**LV.RES x**
reset, 0 for soft reset (on next ext. clock), 1 for hard reset

**LV.POS / LV.POS x**
get/set current position

**LV.L.ST / LV.L.ST x**
get/set loop start

**LV.L.LEN / LV.L.LEN x**
get/set loop length

**LV.L.DIR / LV.L.DIR x**
get/set loop direction

**LV.CV x**
get the current CV value for channel x

**CY.PRE / CY.PRE x**
return current preset / load preset x

**CY.RES x**
reset channel x (0 = all)

**CY.POS x / CY.POS x y**
get / set position of channel x (x = 0 to set all), position between 0-255

**CY.REV x**
reverse channel x (0 = all)

**CY.CV x**
get the current CV value for channel x

**MID.SLEW t**
set pitch slew time in ms (applies to all allocation styles except FIXED)

**MID.SHIFT o**
shift pitch CV by standard Teletype pitch value (e.g. N 6, V -1, etc)

**ARP.HLD h**
0 disables key hold mode, other values enable

**ARP.STY y**
set base arp style [0-7]

**ARP.GT v g**
set voice gate length [0-127], scaled/synced to course divisions of voice clock

**ARP.SLEW v t**
set voice slew time in ms

**ARP.RPT v n s**
set voice pattern repeat, n times [0-8], shifted by s semitones [-24, 24]

**ARP.DIV v d**
set voice clock divisor (euclidean length), range [1-32]

**ARP.FIL v f**
set voice euclidean fill, use 1 for straight clock division, range [1-32]

**ARP.ROT v r**
set voice euclidean rotation, range [-32, 32]

**ARP.ER v f d r**
set all euclidean rhythm

**ARP.RES v**
reset voice clock/pattern on next base clock tick

**ARP.SHIFT v o**
shift voice cv by standard tt pitch value (e.g. N 6, V -1, etc)

## Whitewhale

**WW.PRESET x**
Recall preset (0-7)

**WW.POS x**
Cut to position (0-15)

**WW.SYNC x**
Cut to position (0-15) and hard-sync the clock (if clocked internally)

**WW.START x**
Set the loop start position (0-15)

**WW.END x**
Set the loop end position (0-15)

**WW.PMODE x**
Set the loop play mode (0-5)

**WW.PATTERN x**
Change pattern (0-15)

**WW.QPATTERN x**
Change pattern (0-15) after current pattern ends

**WW.MUTE1 x**
Mute trigger 1 (0 = on, 1 = mute)

**WW.MUTE2 x**
Mute trigger 2 (0 = on, 1 = mute)

**WW.MUTE3 x**
Mute trigger 3 (0 = on, 1 = mute)

**WW.MUTE4 x**
Mute trigger 4 (0 = on, 1 = mute)

**WW.MUTEA x**
Mute CV A (0 = on, 1 = mute)

**WW.MUTEB x**
Mute CV B (0 = on, 1 = mute)

## Meadowphysics

**MP.PRESET x**
set Meadowphysics to preset x (indexed from 0)

**MP.RESET x**
reset countdown for channel x (0 = all, 1-8 = individual channels)

**MP.STOP x**
reset channel x (0 = all, 1-8 = individual channels)

## Earthsea

**ES.PRESET x**
Recall preset x (0-7)

**ES.MODE x**
Set pattern clock mode. (0=normal, 1=II clock)

**ES.CLOCK x**
If II clocked, next pattern event

**ES.RESET x**
Reset pattern to start (and start playing)

**ES.PATTERN x**
Select playing pattern (0-15)

**ES.TRANS x**
Transpose the current pattern

**ES.STOP x**
Stop pattern playback.

**ES.TRIPLE x**
Recall triple shape (1-4)

**ES.MAGIC x**
Magic shape (1= halfspeed, 2=doublespeed, 3=linearize)

## Orca

**OR.CLK x**
Advance track x (1-4)

**OR.RST x**
Reset track x (1-4)

**OR.GRST x**
Global reset (x can be any value)

**OR.TRK x**
Choose track x (1-4) to be used by OR.DIV, OR.PHASE, OR.WGT or OR.MUTE

**OR.DIV x**
Set divisor for selected track to x (1-16)

**OR.PHASE x**
Set phase for selected track to x (0-16)

**OR.WGT x**
Set weight for selected track to x (1-8)

**OR.MUTE x**
Mute trigger selected by OR.TRK (0 = on, 1 = mute)

**OR.SCALE x**
Select scale x (1-16)

**OR.BANK x**
Select preset bank x (1-8)

**OR.PRESET x**
Select preset x (1-8)

**OR.RELOAD x**
Reload preset or bank (0 - current preset, 1 - current bank, 2 - all banks)

**OR.ROTS x**
Rotate scales by x (1-15)

**OR.ROTW x**
Rotate weights by x (1-3)

**OR.CVA x**
Select tracks for CV A where x is a binary number representing the tracks

**OR.CVB x**
Select tracks for CV B where x is a binary number representing the tracks

## Just Friends

**JF.TR x y**
Simulate a TRIGGER input. x is channel (0 = all) and y is state (0 or 1)

**JF.RMODE x**
Set the RUN state of Just Friends when no physical jack is present. (0 = run off, non-zero = run on)

**JF.RUN x**
Send a 'voltage' to the RUN input. Requires JF.RMODE 1 to have been executed, or a physical cable in JF's input. Thus Just Friend's RUN modes are accessible without needing a physical cable & control voltage to set the RUN parameter. use JF.RUN V x to set to x volts. The expected range is V -5 to V 5

**JF.SHIFT x**
Shifts the transposition of Just Friends, regardless of speed setting. Shifting by V 1 doubles the frequency in sound, or doubles the rate in shape. x = pitch, use N x for semitones, or V y for octaves.

**JF.VTR x y**
Like JF.TR with added volume control. Velocity is scaled with volts, so try V 5 for an output trigger of 5 volts. Channels remember their latest velocity setting and apply it regardless of TRIGGER origin (digital or physical). x = channel, 0 sets all channels. y = velocity, amplitude of output in volts. eg JF.VTR 1 V 4.

**JF.TUNE x y z**
Adjust the tuning ratios used by the INTONE control. x = channel, y = numerator (set the multiplier for the tuning ratio), z = denominator (set the divisor for the tuning ratio).

**JF.MODE x**
Set the current choice of standard functionality, or Just Type alternate modes. You'll likely want to put JF.MODE x in your Teletype INIT scripts. x = nonzero activates alternative modes. 0 restores normal.

**JF.VOX x y z**
Create a note at the specified channel, of the defined pitch & velocity. All channels can be set simultaneously with a chan value of 0. x = channel, y = pitch relative to C3, z = velocity (like JF.VTR).

**JF.NOTE x y**
Polyphonically allocated note sequencing. Works as JF.VOX with chan selected automatically. Free voices will be taken first. If all voices are busy, will steal from the voice which has been active the longest. x = pitch relative to C3, y = velocity.

**JF.GOD x**
Redefines C3 to align with the 'God' note. x = 0 sets A to 440, x = 1 sets A to 432.

**JF.TICK x**
Sets the underlying timebase of the Geode. x = clock. 0 resets the timebase to the start of measure. 1 to 48 shall be sent repetitively. The value representing ticks per measure. 49 to 255 sets beats-per-minute and resets the timebase to start of measure.

**JF.QT x**
When non-zero, all events are queued & delayed until the next quantize event occurs. Using values that don't align with the division of rhythmic streams will cause irregular patterns to unfold. Set to 0 to deactivate quantization. x = division, 0 deactivates quantization, 1 to 32 sets the subdivision & activates quantization.

## TELEXi

**TI.PARAM x** TI.PRM
reads the value of PARAM knob x; default return range is from 0 to 16383; return range can be altered by the TI.PARAM.MAP command

**TI.PARAM.QT x** TI.PRM.QT
return the quantized value for PARAM knob x using the scale set by TI.PARAM.SCALE; default return range is from 0 to 16383

**TI.PARAM.N x** TI.PRM.N
return the quantized note number for PARAM knob x using the scale set by TI.PARAM.SCALE

**TI.PARAM.SCALE x** TI.PRM.SCALE
select scale # y for PARAM knob x; scales listed in full description

**TI.PARAM.MAP x y z** TI.PRM.MAP
maps the PARAM values for input x across the range y - z (defaults 0-16383)

**TI.IN x**
reads the value of IN jack x; default return range is from -16384 to 16383 - representing -10V to +10V; return range can be altered by the TI.IN.MAP command

**TI.IN.QT x**
return the quantized value for IN jack x using the scale set by TI.IN.SCALE; default return range is from -16384 to 16383 - representing -10V to +10V

**TI.IN.N x**
return the quantized note number for IN jack x using the scale set by TI.IN.SCALE

**TI.IN.SCALE x**
select scale # y for IN jack x; scales listed in full description

**TI.IN.MAP x y z**
maps the IN values for input jack x across the range y - z (default range is -16384 to 16383 - representing -10V to +10V)

**TI.PARAM.INIT x** TI.PRM.INIT
initializes PARAM knob x back to the default boot settings and behaviors; neutralizes mapping (but not calibration)

**TI.IN.INIT x**
initializes IN jack x back to the default boot settings and behaviors; neutralizes mapping (but not calibration)

**TI.INIT d**
initializes all of the PARAM and IN inputs for device number d (1-8)

**TI.PARAM.CALIB x y** TI.PRM.CALIB
calibrates the scaling for PARAM knob x; y of 0 sets the bottom bound; y of 1 sets the top bound

**TI.IN.CALIB x y**
calibrates the scaling for IN jack x; y of -1 sets the -10V point; y of 0 sets the 0V point; y of 1 sets the +10V point

**TI.STORE d**
stores the calibration data for TXi number d (1-8) to its internal flash memory

**TI.RESET d**
resets the calibration data for TXi number d (1-8) to its factory defaults (no calibration)

## TELEXo

**TO.TR x y**
sets the TR value for output x to y (0/1)

**TO.TR.TOG x**
toggles the TR value for output x

**TO.TR.PULSE x** TO.TR.P
pulses the TR value for output x for the duration set by TO.TR.TIME/S/M

**TO.TR.PULSE.DIV x y** TO.TR.P.DIV
sets the clock division factor for TR output x to y

**TO.TR.PULSE.MUTE x y** TO.TR.P.MUTE
mutes or un-mutes TR output x; y is 1 (mute) or 0 (un-mute)

**TO.TR.TIME x y**
sets the time for TR.PULSE on output n; y in milliseconds

**TO.TR.TIME.S x y**
sets the time for TR.PULSE on output n; y in seconds

**TO.TR.TIME.M x y**
sets the time for TR.PULSE on output n; y in minutes

**TO.TR.WIDTH x y**
sets the time for TR.PULSE on output n based on the width of its current metronomic value; y in percentage (0-100)

**TO.TR.POL x y**
sets the polarity for TR output n

**TO.TR.M.ACT x y**
sets the active status for the independent metronome for output x to y (0/1); default 0 (disabled)

**TO.TR.M x y**
sets the independent metronome interval for output x to y in milliseconds; default 1000

**TO.TR.M.S x y**
sets the independent metronome interval for output x to y in seconds; default 1

**TO.TR.M.M x y**
sets the independent metronome interval for output x to y in minutes

**TO.TR.M.BPM x y**
sets the independent metronome interval for output x to y in Beats Per Minute

**TO.TR.M.COUNT x y**
sets the number of repeats before deactivating for output x to y; default 0 (infinity)

**TO.TR.M.MUL x y**
multiplies the M rate on TR output x by y; y defaults to 1 - no multiplication

**TO.TR.M.SYNC x**
synchronizes the PULSE for metronome on TR output number x

**TO.M.ACT d y**
sets the active status for the 4 independent metronomes on device d (1-8) to y (0/1); default 0 (disabled)

**TO.M d y**
sets the 4 independent metronome intervals for device d (1-8) to y in milliseconds; default 1000

**TO.M.S d y**
sets the 4 independent metronome intervals for device d to y in seconds; default 1

**TO.M.M d y**
sets the 4 independent metronome intervals for device d to y in minutes

**TO.M.BPM d y**
sets the 4 independent metronome intervals for device d to y in Beats Per Minute

**TO.M.COUNT d y**
sets the number of repeats before deactivating for the 4 metronomes on device d to y; default 0 (infinity)

**TO.M.SYNC d**
synchronizes the 4 metronomes for device number d (1-8)

**TO.CV x**
CV target output x; y values are bipolar (-16384 to +16383) and map to -10 to +10

**TO.CV.SLEW x y**
set the slew amount for output x; y in milliseconds

**TO.CV.SLEW.S x y**
set the slew amount for output x; y in seconds

**TO.CV.SLEW.M x y**
set the slew amount for output x; y in minutes

**TO.CV.SET x y**
set the CV for output x (ignoring SLEW); y values are bipolar (-16384 to +16383) and map to -10 to +10

**TO.CV.OFF x y**
set the CV offset for output x; y values are added at the final stage

**TO.CV.QT x y**
CV target output x; y is quantized to output's current CV.SCALE

**TO.CV.QT.SET x y**
set the CV for output x (ignoring SLEW); y is quantized to output's current CV.SCALE

**TO.CV.N x y**
target the CV to note y for output x; y is indexed in the output's current CV.SCALE

**TO.CV.N.SET x y**
set the CV to note y for output x; y is indexed in the output's current CV.SCALE (ignoring SLEW)

**TO.CV.SCALE x y**
select scale # y for CV output x; scales listed in full description

**TO.CV.LOG x y**
translates the output for CV output x to logarithmic mode y; y defaults to 0 (off); mode 1 is for 0-16384 (0V-10V), mode 2 is for 0-8192 (0V-5V), mode 3 is for 0-4096 (0V-2.5V), etc.

**TO.OSC x y**
targets oscillation for CV output x to y with the portamento rate determined by the TO.OSC.SLEW value; y is 1v/oct translated from the standard range (1-16384); a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.SET x y**
set oscillation for CV output x to y (ignores CV.OSC.SLEW); y is 1v/oct translated from the standard range (1-16384); a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.QT x y**
targets oscillation for CV output x to y with the portamento rate determined by the TO.OSC.SLEW value; y is 1v/oct translated from the standard range (1-16384) and quantized to current OSC.SCALE; a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.QT.SET x y**
set oscillation for CV output x to y (ignores CV.OSC.SLEW); y is 1v/oct translated from the standard range (1-16384) and quantized to current OSC.SCALE; a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.N x y**
targets oscillation for CV output x to note y with the portamento rate determined by the TO.OSC.SLEW value; see quantization scale reference for y; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.N.SET x y**
sets oscillation for CV output x to note y (ignores CV.OSC.SLEW); see quantization scale reference for y; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.FQ x y**
targets oscillation for CV output x to frequency y with the portamento rate determined by the TO.OSC.SLEW value; y is in Hz; a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.FQ x y**
sets oscillation for CV output x to frequency y (ignores CV.OSC.SLEW); y is in Hz; a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.LFO x y**
targets oscillation for CV output x to LFO frequency y with the portamento rate determined by the TO.OSC.SLEW value; y is in mHz (millihertz: 10^-3 Hz); a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.LFO.SET x y**
sets oscillation for CV output x to LFO frequency y (ignores CV.OSC.SLEW); y is in mHz (millihertz: 10^-3 Hz); a value of 0 disables oscillation; CV amplitude is used as the peak for oscillation and needs to be > 0 for it to be perceivable

**TO.OSC.CYC x y**
targets the oscillator cycle length to y for CV output x with the portamento rate determined by the TO.OSC.SLEW value; y is in milliseconds

**TO.OSC.CYC.SET x y**
sets the oscillator cycle length to y for CV output x (ignores CV.OSC.SLEW); y is in milliseconds

**TO.OSC.CYC.S x y**
targets the oscillator cycle length to y for CV output x with the portamento rate determined by the TO.OSC.SLEW value; y is in seconds

**TO.OSC.CYC.S.SET x y**
sets the oscillator cycle length to y for CV output x (ignores CV.OSC.SLEW); y is in seconds

**TO.OSC.CYC.M x y**
targets the oscillator cycle length to y for CV output x with the portamento rate determined by the TO.OSC.SLEW value; y is in minutes

**TO.OSC.CYC.M.SET x y**
sets the oscillator cycle length to y for CV output x (ignores CV.OSC.SLEW); y is in minutes

**TO.OSC.SCALE x y**
select scale # y for CV output x; scales listed in full description

**TO.OSC.WAVE x y**
set the waveform for output x to y; y values range 0–4999; values translate to sine (0), triangle (1000), saw (2000), pulse (3000), or noise (4000); oscillator shape between values is a blend of the pure waveforms

**TO.OSC.RECT x y**
rectifies the polarity of the oscillator for output x to y; range for y is -2 to 2; default is 0 (no rectification); 1 & -1 are partial rectification - omitting all values on the other side of the sign; 2 & -2 are full rectification - inverting values from the other pole

**TO.OSC.WIDTH x y**
sets the width of the pulse wave on output x to y; y is a percentage of total width (0 to 100); only affects waveform 3000

**TO.OSC.SYNC x**
resets the phase of the oscillator on CV output x (relative to TO.OSC.PHASE)

**TO.OSC.PHASE x y**
sets the phase offset of the oscillator on CV output x to y (0 to 16383); y is the range of one cycle

**TO.OSC.SLEW x y**
sets the frequency slew time (portamento) for the oscillator on CV output x to y; y in milliseconds

**TO.OSC.SLEW.S x y**
sets the frequency slew time (portamento) for the oscillator on CV output x to y; y in seconds

**TO.OSC.SLEW.M x y**
sets the frequency slew time (portamento) for the oscillator on CV output x to y; y in minutes

**TO.OSC.CTR x y**
centers the oscillation on CV output x to y; y values are bipolar (-16384 to +16383) and map to -10 to +10

**TO.ENV.ACT x y**
activates/deactivates the AD envelope generator for the CV output x; y turns the envelope generator off (0 - default) or on (1); CV amplitude is used as the peak for the envelope and needs to be > 0 for the envelope to be perceivable

**TO.ENV.TRIG x**
triggers the envelope at CV output x to cycle; CV amplitude is used as the peak for the envelope and needs to be > 0 for the envelope to be perceivable

**TO.ENV.ATT x y**
set the envelope attack time to y for CV output x; y in milliseconds (default 12 ms)

**TO.ENV.ATT.S x y**
set the envelope attack time to y for CV output x; y in seconds

**TO.ENV.ATT.M x y**
set the envelope attack time to y for CV output x; y in minutes

**TO.ENV.DEC x y**
set the envelope decay time to y for CV output x; y in milliseconds (default 250 ms)

**TO.ENV.DEC.S x y**
set the envelope decay time to y for CV output x; y in seconds

**TO.ENV.DEC.M x y**
set the envelope decay time to y for CV output x; y in minutes

**TO.ENV.EOR x n**
fires a PULSE at the End of Rise to the unit-local trigger output 'n' for the envelope on CV output x; n refers to trigger output 1-4 on the same TXo as CV output 'y'

**TO.ENV.EOC x n**
fires a PULSE at the End of Cycle to the unit-local trigger output 'n' for the envelope on CV output x; n refers to trigger output 1-4 on the same TXo as CV output 'y'

**TO.ENV.LOOP x y**
causes the envelope on CV output x to loop for y times; a y of 0 will cause the envelope to loop infinitely; setting y to 1 (default) disables looping and (if currently looping) will cause it to finish its current cycle and cease

**TO.TR.INIT x**
initializes TR output x back to the default boot settings and behaviors; neutralizes metronomes, dividers, pulse counters, etc.

**TO.CV.INIT x**
initializes CV output x back to the default boot settings and behaviors; neutralizes offsets, slews, envelopes, oscillation, etc.

**TO.INIT d**
initializes all of the TR and CV outputs for device number d (1-8)

**TO.KILL d**
cancels all TR pulses and CV slews for device number d (1-8)