<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIQB1 cmd value1 value2</td>
<td>Execute the specified query with 1 parameter and get a byte value back</td>
</tr>
<tr>
<td>IIQB2 cmd value1 value2</td>
<td>Execute the specified query with 2 byte parameters and get a byte value back</td>
</tr>
<tr>
<td>IIQB3 cmd value1 value2 value3</td>
<td>Execute the specified query with 3 byte parameters and get a byte value back</td>
</tr>
</tbody>
</table>

**Ansible**

- **ANS.G.LED x y**
  - get grid LED buffer at position x, y
- **ANS.G xy**
  - get/set grid key on/off status at position x, y
- **ANS.G P x y**
  - simulate grid key press at position (x, y)
- **ANS.A LED n x**
  - read asc LED buffer for ring n, LED x clockwise from north
- **ANS.A / ANS.A n d**
  - set asc encoder event for ring n, delta d
- **ANS.APP x**
  - get/set active app
- **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**
  - get/set position z for track z, parameter y
- **KR.L.ST x y**
  - get loop start for track x, parameter y / set to z
- **KR.L.LEN x y**
  - get length of track x, parameter y / set to z
- **KR.ER x y**
  - reset position to loop start for track x, parameter y
- **KR.CV x**
  - get the current CV value for channel x
- **KR.MUTE x**
  - toggle mute state for channel x
- **KR.CLK x**
  - advance the clock for channel x (channel must have teletype clocking enabled)
- **KR.PG / KR.PG x**
  - get/set the active page
- **KR.CUE / KR.CUE x**
  - get/set the cued pattern
- **KR.DIR / KR.DIR x**
  - get/set the step direction
- **KR.DUR x**
  - get the current duration value for channel x
- **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”
- **ME.CV x**
  - get the current CV value for channel x
- **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
- **CV.PRE / CV.PRE x**
  - return current preset / load preset x
- **CV.RES x**
  - reset channel x (0 = all)
- **CV.POS x / CV.POS x y**
  - get / set position of channel x (x = 0 to set all), position between 0-255
- **CV.REV x**
  - reverse channel x (0 = all)
- **CV.CV x**
  - get the current CV value for channel x

**Magic Slew**

- **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**
  - disable 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.EV 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)

**White Whale**

- **WW.PRESET x**
  - Recall preset (0-7)
- **WW.CHANGE 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.MODE x**
  - Set the loop play mode (0-5)
- **WW.GUIDE x**
  - Get the current guide value for channel x
- **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.MUTEV x**
  - Mute CV B (0 = on, 1 = mute)
- **WW.MUTEH x**
  - Mute CV C (0 = on, 1 = mute)
- **White Whale (indexed from 0) reset countdown for channel x (0 = all, 1-8 = individual channels) reset channel x (0 = all, 1-8 = individual channels) White Whale**

**Earthsea**

- **ES.PRESET x**
  - Recall preset x (0-7)
- **ES.MODE x**
  - Set pattern clock mode. (0=normal, 1=II clock)
- **ES.CLOCK x**
  - Set pattern clock mode. (0=normal, 1=II clock)
- **ES.SHIFT x**
  - Shift 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.TRIPLEX x**
  - Recall triple shape (1-4)
- **ES.MAGIC x**
  - Magic shape (1= halfspeed, 2=doublespeed, 3=linearize)
- **ES.CV x**
  - Get the current CV value for channel x
Orca

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

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

**OR.GRT 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.ADDR x**
Sets JF II address (1 = primary, 2 = secondary). Use with only one JF on the bus! Saves to JF internal memory, so only one-time config is needed

**JF.SEL x**
Sets target JF unit (1 = primary, 2 = secondary)

**JF0: . . .**
Send following JF OPs to both units starting with selected unit

**JF1: . . .**
Send following JF OPs to unit 1 ignoring the currently selected unit

---

**JF2: . . .**
Send following JF OPs to unit 2 ignoring the currently selected unit

**JF.RAMP**
Gets value of RAMP knob.

**JF.CURVE**
Gets value of CURVE knob.

**JF.FR**
Gets value of FM knob.

**JF.INTONE**
Gets value of INTONE knob and CV offset.

**JF.TIME**
Gets value of TIME knob and CV offset.

**JF.SPEED**
Gets value of SPEED switch (1 = sound, 0 = shape).

**JF.TSC**
Gets value of MODE switch (0 = transient, 1 = sustain, 2 = cycle).

**JF.TR x y**
Simulate a TRIGGER input. x is channel (0 = all primary JF channels, 1-6 = primary JF, 7-12 = secondary JF, 1 = all channels both JF) and y is state (0 or 1)

**JF.RMODE x**
Set the RUN state of Just Friends when no physical jack is present (1 = 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 & control voltage to set the RUN parameter. use JF.RUN V x to set to x volts.

**JF.SHIFT x**
Shifts the transposition of Just Friends, regardless of sound 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 x for octaves.

**JF.VTR x y**
Send with added volume control. Velocity is scaled with volts, so try V 5 for an output trigger of 5 volts. Channels remember their last velocity setting and apply it regardless of TRIGGER origin (digital or physical). x = channel, y = velocity, amplitude or 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.TUNE 0 @ 0 resets to default ratios.

---

**JF.MODE x**
Set the current choice of standard functionality, or Just Type alternate modes (Speed switch to Sound for Synth, Shape for Geode). You’ll likely want to put JF.MODE x in your Teletype initial scripts. x = nonzero activates alternative modes. 0 resets normal.

**JF.VOX x y z**
Synth mode: 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). Geode mode: Create a stream of rhythmic envelopes on the named channel. x = channel, y = division, z = number of repeats.

**JF.NOTE x y**
Synth: 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. Geode: works as JF.VOX with dynamic allocation of channel. Assigns the rhythmic stream to the oldest unused channel, if all are busy, the longest running channel. x = division, y = number of repeats.

**JF.POLY x y**
As JF.NOTE but across dual JF. Switches between primary and secondary units every 6 notes or until reset using JF.POLY.RESET.

**JF.POLY.RESET**
Resets JF.POLY note count.

**JF.PITCH x y**
Change pitch without retriggering. x = channel, y = pitch relative to C3.

**JF.GOD x**
Redefines C3 to align with the ‘God’ note.

**JF.POLY x y**
Set the polyphony of JF when no physical jack is present. primary JF, secondary JF, or both JF.

**JF.VTR x y**
Sets target JF unit (1-16)

---

**Faderbank**

**FAKER x F**
Reads the value of the FADER slider x; default return range is from 0 to 16383. Up to four Faderbanks can be addressed; x value between 1 and 16 correspond to Faderbank 1, x between 17 and 32 to Faderbank 2, etc.

**FADE.RANGE x y z**
FB.S
Set static scaling of the FADER x to between min and max.

**FADER.CAL.MIN x F**
FB.C.MIN
Reads FADER x minimum position and assigns a zero value

**FAADER.CAL.MAX x F**
FB.C.MAX
Reads FADER x maximum position and assigns the maximum point

**FAADER.CAL.RESET x F**
FB.C.R
Resets the calibration for FADER x

**ER-301**

**SC.TR x y**
Set trigger output for the ER-301 virtual output x to y (0-1)

---

**TILEX**

**TI.PARAM x F**
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.MAP x y**
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 x to y as 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.IN.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**
TI.PRM.CALIB calibrates the scaling for IN jack x; y of 0 sets the -0 bound; y of 1 sets the +0 bound

**TI.STORE d**
stores the calibration data for TXi number d

**TI.RESET d**
resets the calibration data for TXi number d (1-8) to its factory defaults (no calibration)
TO.TR.M.ACT x y
sets the polarity for the device d to y in milliseconds; default 0 (mute);

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

TO.TR.M.M x y
sets the 4 independent metronome intervals for device d to y in milliseconds;

TO.TR.M.M x y
Locks the current offset (CV, OFF) as a calibration offset and saves it to persist between power cycles for output.

TO.CV.RESET x y
Clears the calibration offset for output to.

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 CV to logarithmic mode; y defaults to off (0); mode 1 is for 0-16384 (0-10V), mode 2 is for 0-8192 (0-5V), mode 3 is for 0-4096 (0-2.5V), etc.

TO.CV.CALIB x y
sets the oscillator cycle length to y for CV output x with the portamento rate determined by the TO.CV.SCALE value; y is in minutes

TO.CV.M.S x y
sets the oscillator cycle length to y for CV output x; y is in seconds

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

TO.C.V.CY.C.S x y
sets the oscillator cycle length to y for CV output x (ignores CV.CS.SLEW); y is in seconds

TO.CV.CY.C.M x y
targets the oscillator cycle length to y for CV output x with the portamento rate determined by the TO.CV.SCALE value; y is in milliseconds

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.ACT x y
sets the envelope attack time to y for CV output x; y is in milliseconds (default 12 ms)

TO.ENV.ADT x y
sets the envelope attack time to y for CV output x; y is in seconds

TO.ENV.M x y
sets the envelope attack time to y for CV output x; y is in minutes

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

TO.ENV.TOG x n
at the end of cycle of CV output x, fires a PULSE to the trigger output n

TO.ENV.EOR x n
at the end of cycle of CV output x, fires a PULSE to the trigger output n

TO.ENV.EOR x n
at the end of cycle of CV output x, fires a PULSE to the trigger output n

TO.ENV.LOP x y
causes the envelope on CV output x to loop for y times

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

TO.CV.INIT x y
initializes CV output x back to the default boot settings and behaviors; neutralizes offsets, slopes, 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)
Sets output x to value y. Use V y for volts.

Crow.SEL x
Sets target crow unit (1 = default, to 4).

Crow: ... Send following CROW OPs to all units starting with selected unit.

Crow: ... Send following CROW OPs to unit 1 ignoring the currently selected unit.

Crow2: ... Send following CROW OPs to unit 2 ignoring the currently selected unit.

Crow3: ... Send following CROW OPs to unit 3 ignoring the currently selected unit.

Crow4: ... Send following CROW OPs to unit 4 ignoring the currently selected unit.

Crow.V x y
Sets output x to value y. Use V y for volts.

Crow.SLEW x y
Sets output x slew rate to y milliseconds.

Crow.C1 x
Calls the function ii.self.call1(x) on crow.

Crow.C2 x y
Calls the function ii.self.call2(x, y) on crow.

Crow.C3 x y z
Calls the function ii.self.call3(x, y, z) on crow.

Crow.C4 x y z t
Calls the function ii.self.call4(x, y, z, t) on crow.

Crow.RST
Calls the function crow.reset() returning crow to default state.

Crow.PULSE x y z t
Creates a trigger pulse on output x with duration y ms, falling in z ms, and reaching height t.

Crow.AR x y z t
Creates an envelope on output x, rising in y ms, falling in z ms, and reaching height t.

Crow.LFO x y z t
Starts an envelope on output x at rate y where \( \theta = 1Hz \) with 1 octave scaling. z sets amplitude and t sets skew for asymmetrical triangle waves.

Crow.IN x
Gets voltage at input x.

Crow.OUT x
Gets voltage of output x.

Crow.Q0
Returns the result of calling the function crow.self.query0().

Crow.Q1 x
Returns the result of calling the function crow.self.query1(x).

Crow.Q2 x y
Returns the result of calling the function crow.self.query2(x, y).

Crow.Q3 x y z
Returns the result of calling the function crow.self.query3(x, y, z).

W/1.0

W.SEL x
Sets target W/2.0 unit (1 = primary, 2 = secondary).

W/1: ... Send following W/2.0 OPs to unit 1 ignoring the currently selected unit.

W/2: ... Send following W/2.0 OPs to unit 2 ignoring the currently selected unit.

W/2.0

W/2.0 synth

W/2.0 OPs to unit 2 ignoring the currently selected unit.

W/2.0 tape

W/T.REC active
Sets recording state to active (s8)

W/T.PLAY playback
Set the playback state. -1 will flip playback direction (s8)

W/T.REV Reverse the direction of playback

W/T.SPEED speed deno
Set speed as a rate, or ratio. Negative values are reverse (s16V)

W/T.FREQ freq Set speed as a frequency (s16V) style value. Maintains reverse state

W/T.ERASE.LVL level
Strength of erase when recording. 0 is overlaps, 1 is overwrite. Opposite of feedback (s16V)

W/T.MONITOR.LVL gain
Level of input passed directly to output (s16V)

W/T.REC.LVL gain
Level of input material recorded to tape (s16V)

W/T.ECHOMODE is_echo
Set to 1 to playback before erase. 0 (default) erases first (s8)

W/T.LOOP.START Set the current time as the beginning of a loop

W/T.LOOP.END
Set the current time as the loop end, and jump to start

W/T.LOOP.ACTIVE state
Set the state of looping (s8)

W/T.LOOP.SCALE scale
Sets loop scale (s16V), or DiviNegative) loop brace by zero. Rates to zeros with window (s8)

W/T.LOOP.NEXT direction
Move loop brace forward/backward by length of loop. Zero jumps to loop start (s8)

W/T.TIME seconds sub
Move haystack to an arbitrary location on tape (s16)

W/2.0 synth

W/S.PITCH voice pitch
Set voice (s8) to pitch (s16V) in volts-per-octave

W/S.VEL voice velocity
Strike the vactrol of voice (s8) at velocity (s16V) in volts

W/S.VOX voice pitch velocity
Set voice (s8) to pitch (s16V) and strike the vactrol at velocity (s16V)

W/S.NOTE pitch level
Dynamically assign a voice, set to pitch (s16V), strike with velocity(s16V)

W/S.POLY pitch level
As W/S.NOTE but across dual W/. Switches between primary and secondary units every 4 notes or until reset using W/S.. POLY..RESET.

W/S.POLY..RESET
Resets W/S.. POLY note count.

W/S.AR.MODE is_ar
in attack-release mode, all notes are plucked and no release is required.

W/S.LPG.TIME time
turce time (s16V) constant. -5=drones, 0=max, 5=flutes

W/S.LPG.SYM symmetry
Vactrol attack-release ratio. -5=fastest attack, 5=long swells (s16V)

W/S.CURVE curve
cross-fade waveforms: -5=square, 0=triangle, 5=sine (s16V)

W/S.RAMP ramp
Waveform symmetry: -5=swell, 0=triangle, 5=sawtooth (NB: affects FM tone)

W/S.FM.INDEX index
Amount of FM modulation. -5=negative, 0=minimum, 5=maximum (s16V)

W/S.FM.RATIO num den
Ratio of the FM modulator to carrier as a ratio. Floating point values up to 20.0 supported (s16V)

W/S.FM..ENV amount
Amount of vactrol envelope applied to fm index, -5 to +5 (s16V)

W/S.PATCH jack param
Patch a hardware jack (s8) to a param (s8) destination

W/S.VOICES count
Number of polyphonic voices to allocate. Use 0 for unison mode (s8)
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EX / EX x</td>
<td>Get or set currently selected unit to x (1-4)</td>
</tr>
<tr>
<td>EX: ...</td>
<td>Send following Disting ops to unit 1 ignoring the currently selected unit</td>
</tr>
<tr>
<td>EX2: ...</td>
<td>Send following Disting ops to unit 2 ignoring the currently selected unit</td>
</tr>
<tr>
<td>EX3: ...</td>
<td>Send following Disting ops to unit 3 ignoring the currently selected unit</td>
</tr>
<tr>
<td>EX4: ...</td>
<td>Send following Disting ops to unit 4 ignoring the currently selected unit</td>
</tr>
<tr>
<td>EX.PRESET / EX.PRESET x</td>
<td>EX.PRESET send load preset x or get the currently loaded preset</td>
</tr>
<tr>
<td>EX.SAVE x</td>
<td>Save to preset x</td>
</tr>
<tr>
<td>EX.RESET</td>
<td>Reset the currently loaded preset</td>
</tr>
<tr>
<td>EX.ALG / EX.ALG x</td>
<td>EX.A get or set the current algorithm to x (single algorithms only)</td>
</tr>
<tr>
<td>EX.CTRL x y</td>
<td>EX.C Set I2C controller x to value y</td>
</tr>
<tr>
<td>EX.PARAM x / EX.PARAM x y</td>
<td>EX.P Set parameter x to value y or get the current parameter value</td>
</tr>
<tr>
<td>EX.PV x y</td>
<td>Set parameter x using a value determined by scaling y from 0..16384 range</td>
</tr>
<tr>
<td>EX.MIN x</td>
<td>Get the minimum possible value for parameter x</td>
</tr>
<tr>
<td>EX.MAX x</td>
<td>Get the maximum possible value for parameter x</td>
</tr>
<tr>
<td>EX.VOX x y z</td>
<td>EX.V Send a note to channel x using pitch y and velocity z (voice allocated by the Disting)</td>
</tr>
<tr>
<td>EX.VOX.P x y</td>
<td>EX.VP Set voice x to pitch y</td>
</tr>
<tr>
<td>EX.VOX.O x y</td>
<td>EX.V0 Send a note off to voice x</td>
</tr>
<tr>
<td>EX.CH x</td>
<td>EX.CH # Select default note channel (for multi channel algorithms like Poly FM)</td>
</tr>
<tr>
<td>EX.NOTE x y</td>
<td>EX.N Send a note using pitch x and velocity y (voice allocated by the Disting)</td>
</tr>
<tr>
<td>EX.M.N# x y</td>
<td>EX.M Send MIDI Note off message on channel x for note x (0..127)</td>
</tr>
<tr>
<td>EX.M.CC x y</td>
<td>EX.M Send MIDI CC message for controller x (0..127) and value y (0..127)</td>
</tr>
<tr>
<td>EX.M.N# x y z</td>
<td>EX.M Send MIDI Note off message on channel x for note x (0..127) and velocity z (0..127)</td>
</tr>
<tr>
<td>EX.M.NO x</td>
<td>EX.M Send MIDI Note off message for note x (0..127)</td>
</tr>
<tr>
<td>EX.M.NO# x y</td>
<td>EX.M Send MIDI Note off message on channel x for note y (0..127)</td>
</tr>
<tr>
<td>EX.M.CC# x y z</td>
<td>EX.M Send MIDI CC message on channel x for controller y (0..127) and value z (0..127)</td>
</tr>
<tr>
<td>EX.M.PB x</td>
<td>EX.M Send MIDI Pitch bend message</td>
</tr>
<tr>
<td>EX.M.PRG x</td>
<td>EX.M Send MIDI Program Change message</td>
</tr>
<tr>
<td>EX.M.CLK</td>
<td>EX.M Send MIDI clock message</td>
</tr>
<tr>
<td>EX.M.START</td>
<td>EX.M Start MIDI Start message</td>
</tr>
<tr>
<td>EX.M.STOP</td>
<td>EX.M Stop MIDI Stop message</td>
</tr>
<tr>
<td>EX.M.CONT</td>
<td>EX.M Send MIDI Continue message</td>
</tr>
<tr>
<td>EX.MB.CH / EX.MB.CH x</td>
<td>EX.MB Get or set the currently selected MIDI Bus channel (1-16)</td>
</tr>
<tr>
<td>EX.SB.CH / EX.SB.CH x</td>
<td>EX.SB Get or set the currently selected Select Bus channel (1-16)</td>
</tr>
<tr>
<td>EX.SB.N x y</td>
<td>EX.SB Send Select Bus Note On message for note x (0..127) and velocity y (0..127)</td>
</tr>
<tr>
<td>EX.SB.NO x y</td>
<td>EX.SB Send Select Bus Note off message for note x (0..127) and velocity y (0..127)</td>
</tr>
<tr>
<td>EX.SB.PB x</td>
<td>EX.SB Send Select Bus Pitch bend message</td>
</tr>
<tr>
<td>EX.SB.CC x y</td>
<td>EX.SB Send Select Bus CC message for controller x (0..127) and value y (0..127)</td>
</tr>
<tr>
<td>EX.SB.PRG x</td>
<td>EX.SB Send Select Bus Program Change message</td>
</tr>
<tr>
<td>EX.SB.CK</td>
<td>EX.SB Send Select Bus Clock message</td>
</tr>
<tr>
<td>EX.SB.START</td>
<td>EX.SB Send Select Bus Start message</td>
</tr>
<tr>
<td>EX.SB.STOP</td>
<td>EX.SB Send Select Bus Stop message</td>
</tr>
<tr>
<td>EX.SB.CONT</td>
<td>EX.SB Send Select Bus Continue message</td>
</tr>
<tr>
<td>EX.A1 / EX.A1 x</td>
<td>EX.A Get or set the left dual algorithm</td>
</tr>
<tr>
<td>EX.A2 / EX.A2 x</td>
<td>EX.A Get or set the right dual algorithm</td>
</tr>
<tr>
<td>EX.A12 x y</td>
<td>EX.A Set both dual algorithms</td>
</tr>
<tr>
<td>EX.P1 x / EX.P1 x y</td>
<td>EX.P1 Get left algorithm parameter x or set it to value y</td>
</tr>
<tr>
<td>EX.P2 x / EX.P2 x y</td>
<td>EX.P2 Get right algorithm parameter x or set it to value y</td>
</tr>
<tr>
<td>EX.PV1 x y</td>
<td>EX.PV Set left algorithm parameter x using a value determined by scaling y from 0..16384 range</td>
</tr>
<tr>
<td>EX.PV2 x y</td>
<td>EX.PV Set right algorithm parameter x using a value determined by scaling y from 0..16384 range</td>
</tr>
<tr>
<td>EX.MIN1 x</td>
<td>EX.MIN Get left algorithm parameter minimum value</td>
</tr>
<tr>
<td>EX.MAX1 x</td>
<td>EX.MAX Get left algorithm parameter maximum value</td>
</tr>
<tr>
<td>EX.MIN2 x</td>
<td>EX.MIN Get right algorithm parameter minimum value</td>
</tr>
<tr>
<td>EX.MAX2 x</td>
<td>EX.MAX Get right algorithm parameter maximum value</td>
</tr>
<tr>
<td>EX.Z1 / EX.Z1 x</td>
<td>EX.Z1 Get left Z knob value or set left Z parameter (0..127 range)</td>
</tr>
<tr>
<td>EX.Z2 / EX.Z2 x</td>
<td>EX.Z2 Get right Z knob value or set right Z parameter (0..127 range)</td>
</tr>
<tr>
<td>EX.ZO1</td>
<td>EX.ZO Restore control for left Z knob and input</td>
</tr>
<tr>
<td>EX.ZO2</td>
<td>EX.ZO Restore control for right Z knob and input</td>
</tr>
<tr>
<td>EX.PRE1 x</td>
<td>EX.PRE Load left preset from x slot</td>
</tr>
<tr>
<td>EX.PRE2 x</td>
<td>EX.PRE Load right preset from x slot</td>
</tr>
<tr>
<td>EX.SAVE1 x</td>
<td>EX.SAVE Save left preset to x slot</td>
</tr>
<tr>
<td>EX.SAVE2 x</td>
<td>EX.SAVE Save right preset to x slot</td>
</tr>
<tr>
<td>MA.SELECT x</td>
<td>MA.SELECT Select the default matrixarchate module, default 1</td>
</tr>
<tr>
<td>MA.STEP</td>
<td>MA.STEP Advance program sequencer</td>
</tr>
<tr>
<td>MA.RESET</td>
<td>MA.RESET Reset program sequencer</td>
</tr>
<tr>
<td>MA.PGM pgm</td>
<td>MA.PGM PGM Select the current program (1-based)</td>
</tr>
<tr>
<td>MA.ON x y</td>
<td>MA.ON Get or set row x and column y in the current program (rows/columns are 0-based)</td>
</tr>
<tr>
<td>MA.PON pgm x y</td>
<td>MA.PON Connect row x and column y in program pgm</td>
</tr>
<tr>
<td>MA.OFF x y</td>
<td>MA.OFF Disconnect row x and column y in the current program</td>
</tr>
<tr>
<td>MA.POFF x y pgm</td>
<td>MA.POFF Connect row x and column y in program pgm</td>
</tr>
<tr>
<td>MA.SAVE x y</td>
<td>MA.SAVE Get or set row x and column y to state (1 - on, 0 - off)</td>
</tr>
<tr>
<td>MA.PSET pgm x y state</td>
<td>MA.PSET Get or set row x and column y in program pgm to state (1 - on, 0 - off)</td>
</tr>
<tr>
<td>MA.COL col / MA.COL col value</td>
<td>MA.COL Set or get column col (as a 16 bit unsigned value where each bit represents a connection)</td>
</tr>
<tr>
<td>MA.PCOL pgm col / MA.PCOL pgm col value</td>
<td>MA.PCOL Get or set column col in program pgm</td>
</tr>
<tr>
<td>MA.ROW row / MA.ROW row value</td>
<td>MA.ROW Get or set row x in program pgm</td>
</tr>
<tr>
<td>MA.PROW pgm row / MA.PROW pgm row value</td>
<td>MA.PROW Get or set row x in program pgm</td>
</tr>
<tr>
<td>MA.CLR</td>
<td>MA.CLR Clear all connections</td>
</tr>
<tr>
<td>MA.PCLR pgm</td>
<td>MA.PCLR Clear all connections in program pgm</td>
</tr>
</tbody>
</table>
Set the playing speed \( x \) (1..32767) of the buffer. \( x = 100 \) is equivalent to 'normal speed', \( x = 58 \) means double the speed, \( x = 200 \) means half the speed, etc.

Set the feedback length \( x \) (0..255) of the buffer

Set the note shift of recorded notes to \( x \) semitones (-127..127)

Set the velocity shift of recorded notes to \( x \) (-127..127)

Set the note duration shift ('time shift') of recorded notes to \( x \) ms (-16384..16383)

Set the note offset of recorded notes to \( x \) semitones (-127..127)

Set the velocity offset of recorded notes to \( x \) (-127..127)

Set the note duration offset ('time offset') of recorded notes to \( x \) ms (-16384..16383)

Clear the buffer, erasing all recorded notes in the buffer

Set the buffer mode to \( x \) (0..1). 1) Digital 2) Tape

Get currently set MIDI channel / Set MIDI channel \( x \) (1..16) for MIDI in

Turn on or off 'latching' for MIDI notes received via MIDI in

Get \( x \) (0..7) last note number (0..127) received via MIDI in

Get \( x \) (0..7) last note velocity (1..127) received via MIDI in

Get current value (0..127) of controller \( x \) (0..127) received via MIDI in

Send MIDI Note Off messages for all notes on all channels, and reset note duration, shift, repetition, ratcheting, min/max