monome teletype

Algorithmic ecosystem.

For a complete series of tutorials, see:

http://monome.org/docs/modular/teletype/studies-1

Nomenclature

SCRIPT: multiple *commands*
COMMAND: a series (one line) of *words*
WORD: a text string separated by a space: *value*, *operator*, *variable*, *pre*
VALUE: a number
OPERATOR: a function, may need value(s) as argument(s), may return value
VARIABLE: named memory storage
PRE: condition/rule that applies to rest of the *command*: del, prob, if, s

Parameters

Parameters are like variables, but tied to functionality of the software or hardware. CV & TR are arrays and require an index argument.

IN and PARAM provide CV and physical input into a script. Their state can be read with the listed parameters.

Reading and writing is similar to variables-- assignment happens when the parameter is leftmost in the command (and requires an additional argument: the value to take).

```
    TR A-D       set TR value (0-1)
    TR.TIME A-D  time for TR.PULSE
    TR.POL A-D   set polarity for TR.PULSE

    CV 1-4       CV target value
    CV.SLEW 1-4  CV slew time in ms (how long to reach the target)
    CV.SET 1-4   set CV value directly, ignoring slew time
    CV.OFF 1-4   CV offset (added to CV value at final stage)

    IN           get value of IN jack (0-16383)
    PARAM        get value of PARAM knob (0-16383)
    STATE 1-8    get input state of trigger inputs 1-8

    M            metro time (ms). M script executes at this interval
    M.ACT        [0/1] enable/disable metro
    M.RESET      hard reset metro count without triggering

    TIME         timer value. counts up in ms.
    TIME.ACT     [0/1] enable/disable timer counting

    SCENE        read/recall scene

    SCRIPT 1-8   execute script via command
    MUTE 1-8     mute incoming trigger signals
    UNMUTE 1-8   unmute incoming trigger signals

    KILL         clears stack and delays, cancels pulses and slews
```

Variables

```
    X, Y, Z      general purpose
    T            typically used for time values, but also general
    A-D          assigned 1-4 by default (for TR labeling), reassignable
```

Special variables:

```
    I            overwritten by the L (loop) PRE, but can be general.
    O            auto-increments on each read.
        O.MIN, O.MAX, O.WRAP, O.DIR parameters affect operation.
    DRUNK        changes by -1, 0, or 1 upon each read, saving state.
        DRUNK.MIN, DRUNK.MAX, DRUNK.WRAP parameters affect operation.
    Q            implements a queue or shift register.
      Q.N        sets the read position.
      Q.AVG      will return the average of the entire queue
    FLIP         changes state on each read (0/1)

    NB: Set Q.AVG to set the entire queue to the specified value.
```

Data and Tables

Working range is signed 16 bit: -32768 to 32767

Built-in constant tables for easy note and voltage conversion:

```
    N 0-127      equal temp semi (negatives accepted as well)
    V 0-10       volt lookup (0V to 10V)
    VV 0-1000    volt lookup with decimal precision (0.00V to 10.00V)
    EXP 0-16383  exponential lookup (good for 0V to 10V)
```

Operators

Operators take a variable number of parameters (including none) and typically return one value.

```
    RAND a           generate random number 0-(a)
    RRAND a b        generate random number from (a) to (b)
    TOSS             return random: 0 or 1
    AVG a b          return average of two arguments (a) and (b)
    MIN/MAX a b      choose lesser/greater of two inputs (a) and (b)
    ABS a            return absolute value of (a)
    ADD/SUB/MUL a b  arithmetic
    DIV/MOD a b      arithmetic
    EQ/NE/GT/LT a b  logic: equals, not equals, greater than, less than
    EZ/NZ a          logic: equals zero, not zero
    AND/OR/XOR a b   logic: and, or, xor
    RSH/LSH a b      shift (a) by (b), like MUL/DIV by powers of two
    LIM a b c        clamp to a defined range: (a) input (b) min (c) max
    WRAP a b c       wrapped range defining: (a) input (b) min (c) max
    QT a b           round (a) to closest multiple of (b): quantize
    SCALE a b x y i  scale value (i) from range (a)-(b) to range (x)-(y)
    JI x y           just intonation helper, ratio normalized to 1 V
```

Special case operators:

These act only the hardware and don't return a value.

```
    TR.TOG a         toggle TR (a)
    TR.PULSE a       pulse TR (a) using TR.TIME as an interval
```

Modified commands: PRE

A *PRE* is a short command that modifies the remainder of a command. A PRE needs a separator (colon) to indicate the command it will act upon.

```
PROB a : ..   potential to execute with (a) probability [0-100]

DEL a : ..    delay (postpone) command by (a) ms
  DEL.CLR     kill all delays

S : ..        put command on the stack
  S.CLR       clear the stack
  S.ALL       execute every command on the stack
  S.POP       execute most recent command (pop)
  S.L         length of queue (read only)

IF a : ..     if (a) is not zero, execute command
ELIF a : ..   execute on failed IF/ELIF, and (a) is not zero
ELSE ..       execute on failed IF/ELIF

L a b : ...   LOOP. execute command with I values (a) to (b)
```

Patterns

```
P a          get value at index (a)
P a b        set value at index (a) to (b)
P.N a        select bank (a)
PN a b       get pattern (a) index (b)
PN a b c     set pattern (a) index (b) to (c)
```

Note: For `P` and `PN`, negative index values index from the end (backwards) rather than beginning.

Pattern manipulation:

These commands change pattern length:

```
P.INS a b    insert value (b) at index (a), shift later values down
P.RM a       delete value at (a), shift later values up
P.PUSH a     add value (a) to end of pattern (like a stack)
P.POP        remove and return value from end of pattern (stack)
```

pattern attributes: get current values by omitting a value

```
P.L a        get/set length, nondestructive to data
P.WRAP a     enable/disable (or get) wrapping [0/1]
    NB: P.WRAP changes behavior of P.PREV / P.NEXT
P.START a    get/set start location
P.END a      get/set end location
```

patterns have a "read head" pointer that can be manipulated

```
P.I a        get/set index position
P.HERE       read value at index
P.NEXT       increment index then read
P.PREV       decrement index then read
```

Note: an argument to P.HERE, P.NEXT or P.PREV will move the "read head" pointer and then set the new index to the input value.

Remote

If a monome trilogy module (WW/MP/ES) is attached via the internal ribbon cable, you can remotely control many parameters using the II command.

```
        Eg: II WW.POS 5
```

All of these commands require one parameter. SYNC, RESET, and CLOCK commands need a non-zero parameter to execute.

White Whale

```
WW.PRESET    recall preset
WW.POS       cut to position
WW.SYNC      cut to position, hard sync clock (if clocked internally)
WW.START     set loop start
WW.END       set loop end
WW.PMODE     set play mode (0: normal, 1: reverse, 2: drunk, 3: rand)
WW.PATTERN   change pattern
WW.QPATTERN  change pattern (queued) after current pattern ends
WW.MUTE1     mute trigger 1 (0 = on, 1 = mute)
WW.MUTE2     mute trigger 2 (0 = on, 1 = mute)
WW.MUTE3     mute trigger 3 (0 = on, 1 = mute)
WW.MUTE4     mute trigger 4 (0 = on, 1 = mute)
WW.MUTEA     mute cv A (0 = on, 1 = mute)
WW.MUTEB     mute cv B (0 = on, 1 = mute)
```

Meadowphysics

```
MP.PRESET    recall preset
MP.RESET     reset positions
MP.SYNC      reset positions & hard sync (if clocked internally)
MP.MUTE      mutes the output of a channel (1 - 8)
MP.UNMUTE    unmutes (enables) the output (1 - 8)
MP.FREEZE    freezes the advancement of a channel (1 - 8)
MP.UNFREEZE  unfreezes (enables) advancement of the channel (1 – 8)
MP.STOP      (MP v2) stop channel, 0 for all
```

Earthsea

```
ES.PRESET    recall preset
ES.MODE      set pattern clock mode (0 = normal, 1 = II clock)
ES.CLOCK     (if II clocked) next pattern event
ES.RESET     reset pattern to start (and start playing)
ES.PATTERN   set playing pattern
ES.TRANS     set transposition
ES.STOP      stop pattern playback
ES.TRIPLE    recall triple shape (1-4)
ES.MAGIC     magic shape (1: halfspeed, 2: doublespeed, 3: linearize)
```

One of each module (WW/MP/ES) can be connected on the same cable to one Teletype unit at the same time.

For further documentation & examples see:

http://monome.org/docs/modular