W S P manual.

by

M. Hinton and T. Ungvary

Stockholm 1985

WSP is a real-time interactive music performance program that runs on the HP-1000 in conjunction with the G. Svensson digital oscillator bank to simulate a modular voltage-controlled synthesizer.

There is also a non-real-time version that runs on the VAX-11, using the software oscillators in Michael Hinton's BADA program package.

The user creates boxes of various types, each with a specific function (such as generating a signal, controlling the output of another box, etc) and each with a name of the user's choice. Hardware devices such as keyboards, joysticks, potentiometers, etc are treated as boxes in exactly the same way as software devices: **they must be created and named before they can be used.**

Contents:

Conventions and definitions

Starting Commands

Boxes

COMPILE

WSP can be linked only with the help of the pre-processor COMPILE. This is a Command Language program which allows the user to create a WSP configuration that includes specified numbers of each of the available boxes. To run COMPILE, type:

```
$ COMPILE [v [b]]
```

where v is one of the words VAX or HP, indicating the computer that the output module is to be used on

b is either
BATCH if the compilation is to be carried out in Batch mode
or
NOBATCH if the compilation is to be carried out in User mode

If no parameters are defined, the user is asked:

```
Is output intended for VAX or HP?
```

to which the answer should be either VAX or HP. If any other answers are given, the question is repeated. Configurations created for the VAX cannot be used on the HP, and vice-versa.

BOX-DEFINITIONS

Next, the user is asked to define interactively how many of each of the WSP **boxtypes** are to be available in this particular configuration. For example:

```
RANDOM (default   10):
SDELAY (default    0):
```

The user may type, after the ':',
- carriage return, which means that the default number of boxes will be available.
- 0, which means that it will not be possible to use this particular boxtype at all, since all program lines that refer to it will be omitted. Exceptions: program lines for SWITCH, TRIGGER and CONNEC boxes are included in all configurations.
- a positive integer, defining the maximum number of boxes that will be available of this type. For INFO boxes, this value may not exceed 6.

In the event of an error, a message is displayed at the terminal, and the question is asked again. For example:

```
SDELAY (default    0): -2
Read error, or illegal value; try again!
SDELAY (default    0):
```

If 'CTRL Z' is typed, the complete series of questions starts again from top, though this time the default values are the ones already typed in.

The user is then asked to define the sizes of the four COMMON areas, ADATA, IDATA, LDATA and CDATA. The same definition rules apply to these as to the boxtypes above.

ORDER

Then the default calculation order is displayed. This is the order in which boxtypes are executed every studio sample. The user is invited to redefine the order, by writing boxtypes, one per line, in the required order. When the definition is complete, the new order is displayed, and the user is again invited to redefine it, if he so wishes.

SHARE and COPY

Then the user is asked:

Do you wish to include the SHARE and COPY facilities? (Y/N)

If the answer is Y, the SHARE and COPY facilities will be available for FUNCTIon generators, QUANTIfiers and SEQUENcers. Otherwise, these facilities will not be available, and any attempt to use them will result in error messages.

Versions that allow SHARE and COPY require more program memory than versions that do not allow them; however, they give the user the opportunity to make more efficient use of data memory, where constants and control signals are stored for the boxes in question.

NESTING LEVEL

For HP versions, the user is also asked to specify the nesting level for the CALL command. If a file which is CALLed contains one or more CALL commands itself, the nesting level is 2; if the file thus read in also contains CALL commands, the nesting level is 3; and so on.

The number of levels should be in the range 1-121. If the user specifies a number outside this range, it will automatically be adjusted to the nearest limit (1 or 121). The default value is 1.

The number of levels should be kept as low as possible, since each level requires a 128-word block to be reserved in memory.

Differences between VAX and HP programs

From the user's point of view, there are the following differences between HP and VAX versions

VAX	HP
non-real-time	real-time
the output of COMPILE is a run file - [WSP]BADA.EXE	the output of COMPILE is a text file - [WSP]WSP.FTN which can then be transferred to the HP
software oscillators with variable and controllable wave-forms	hardware sine-wave oscillators
music output is a .DAC file which must be opened and closed as in EMSDAC with TAPE and ENDPLY	output is sound
graphic display via the DISPLAy box	no graphic display

In compiling the user configuration, the following rules are used to determine whether or not files are to be included :

file name	for VAX system	for HP system
name.FTN	include 'name.FOR'	include 'name.FTN'
name.VAX	include 'name.VAX'	omit
name.DIS	include 'name.DIS'	omit
name.HP	omit	include 'name.HP'
EMA.ext	omit	include 'EMA.ext'
name.box*	include if box-definition .GT. 0	
name.SHR	include if SHARE and COPY allowed	
all other types	include	include

*'box' means here one of the box-specific extensions listed in array EXTENS in program unit WSPPREP.FOR, with the exception of 'DIS'

PROGRAMS

All source files for WSP are on directory [WSPFOR], except those that are referred to in INCLUDE statements; these are on directory [WSP.INC].

The programs involved in the execution of COMPILE are on [WSP.COMPILE]:

WSPPREP.EXE	COMPILE.COM
WSPCOMP.EXE	VAXWSP.COM
	HPWSP.COM

MAP

A file-map, showing where all WSP source files are INCLUDEd, can be printed on the line-printer. Do:

```
$ RUN DRA0:[WSP.COMPILE]WSPMAP
```

CONVENTIONS AND DEFINITIONS

CONVENTIONS

In this document, the following conventions apply:

- square brackets [ ] surrounding a symbol or group of symbols indicate an optional parameter in a command line

- in descriptions of command lines, capital letters indicate items that must be typed by the user in the format shown (or, in certain cases, in abbreviated form); lower-case letters indicate items that must be replaced by specific examples of the type shown

  [CREATE]boxtype boxname [parameters]

  would therefore mean that:
  1) the word CREATE is optional but, if typed, must be written in exactly this format
  2) boxtype and boxname must be replaced by specific examples of legal boxtypes and boxnames
  3) optionally, the command line may be completed with a list of actual parameter words and values

SIGNALS

This word is used to refer to what in analog systems are usually called control voltages. Signals are represented internally by floating-point numbers, which are usually in the range 0 to 1 (though there are in fact no restrictions on their size). They control such parameters as frequency, amplitude and time.

TRIGGERS

Triggers are represented internally by logical variables which can be either on (.TRUE.) or off (.FALSE.): they can, for example, be used to determine when boxes are to start or stop operating. All triggers are **normally in the OFF state**; when one is turned ON by the action of a box or by the user's manual intervention, it sends a pulse to one box only, which immediately sends back a counter-pulse to turn the trigger OFF. If connections are made to control several boxes with the same trigger, only the first connection will have the desired effect.

Triggers are discussed in greater detail under the heading 'Boxtypes TRIGGER'

SWITCHES

Switches are represented internally by integers and used, for example, to point to one of several alternatives, such as determining whether ramps are to be linear or exponential, or choosing from a range of triggers one that is to be turned on.

Switches are discussed in greater detail under the heading 'Boxtypes SWITCH'

BOXTYPE

- the word 'boxtype' is used as an abstract term to refer to any one of the twenty-six types of module available to the user
- the word '**box**is used to refer to a concrete example of one of the boxtypes
- '**boxname**is the name given by the user to a particular box, to distinguish it from all other boxes

Thus:

CREATE SWITCH POWER

means **create a box called POWER with the characteristics of the boxtype SWITCH.**

CONTROLS

This term is used to mean the **signal inputs and outputs** which control the functioning of any given box. When modifying, connecting or displaying a specific control input or output, the user may refer to it thus:

boxname/spec

'**boxname**' is the user-defined name of a box
'**spec**' is the control specifier.

For example:

RFREQ/SPEED	refers to the /SPEED input of box RFREQ
BOX77/A5	refers to the /A5 input of box BOX77 - this probably means '**the 5th amplitude input**'

The actual specifiers used vary from box to box, but it should be noted that only the first character of the specifier is significant, except where there are multiple numbered inputs (as in the example BOX77/A5 above), in which case the specifier must consist of ONE letter followed immediately by an integer. Thus, RFREQ/S and RFREQ/SILLY both refer to the same input - RFREQ/SPEED. However BOX77/AMP5 is an error, since there is more than one letter before the integer.

CHARACTER SET

Names devised by the user for boxes and files may consist of up to six characters taken from this list:

ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789;$%?[]@

COMMENTS

! everything in a line following an exclamation mark is interpreted as a comment, and ignored by the program

CONTINUATION

- it is possible to enter command lines on more than one physical terminal line by typing a hyphen (-) as the last character in the physical line

For example:

```
?:SHOW TEXT -
THIS -
IS -
INTERPRETED -
AS -
ONE -
COMMAND
```

causes the program to print:

```
THIS IS INTERPRETED AS ONE COMMAND
```

N.B. ' - ' the hyphen must be the last character in the physical line if it is to be interpreted as a continuation indicator; it must not be followed by spaces or comments

' - ' the hyphen is interpreted as a continuation indicator even if it is part of a comment. For example, the text:

```
?:CLEAR    !we've had enough -----
EXIT
```

is interpreted as

```
?:CLEAR    !we've had enough EXIT
```

i.e. the word EXIT is taken to be part of the comment, and is therefore not executed as a command

CONTROL CHARACTERS

The following control characters have effect only when typed at the terminal: they cannot be used in .WSP files.

'CTRL A' changes the value of the SYNTHE flag from SYNTHE to NOSYNT, or vice versa (see under SET)

'CTRL B' halts program execution, or, if program execution has already been halted by 'CTRL B', restarts program execution; this may be useful on the VAX-11 in order to give more time to other processes on the system

'CTRL N' changes the value of the ECHO flag during execution of SAVE, CALL and SHOW FILE commands; when the ECHO flag is ON, text read from disk or written to disk is displayed at the user's terminal

'CTRL T' deletes all the characters in the current input line

'CTRL V' changes the value of the VERIFY flag from VERIFY to NOVERIfy, or vice versa (see under SET)

'CTRL Z' identical to the command EXIT, except when typed during the execution of a CALL or SAVE command, in which case it has the effect of aborting the current operation

DELETE

The last character in the input line can be deleted with:

a) in the VAX-11, the key 'BACKSPACE'
b) in the HP, the key 'DEL'

WILD-CARDS

The asterisk sign (*) is used in certain commands as a '**wild card**' to refer to groups of boxes; at present it can be used in the following ways:

1) XX* refers to all boxes whose names begin with XX
2) *XX* refers to all boxes whose names include the group of letters XX
3) *XX refers to all boxes whose names end with XX
4) * EITHER refers to all boxes OR represents an empty character string
5) ** as type (4)

Extra characters after the final * in types (1) and (2) are ignored. Thus XX*YY is interpreted as XX*.

DEFAULT-VALUE

The 'number' sign (%) is sometimes be used in conjunction with the commands CREATE and MODIFY to denote 'default value'. For further information, see under CREATE and MODIFY.

NUMBERS

Numerical values may be written with or without a decimal point.

OLD

In all places where it is possible to specify numerical constants, it is also possible to specify the addition, subtraction, multiplication or division of an existing value with a constant. For example:

BOX1 OLD+v	adds 'v' to the current value of BOX1
BOX1/S OLD-v	subtracts 'v' from the current value of BOX1/S
BOX/A5 OLD*v	multiplies 'v' by the current values of all /A5 inputs of boxes whose names include the letters BOX
BOX1*/X3 OLD/v	divides the current value of all /X3 locations of boxes whose names begin with BOX1 by 'v', unless 'v' has the value zero, in which case no division takes place

'v' may be written as an integer or floating-point number

N.B.

a. SWITCHes may never have negative values. If an OLD operation attempts to set a switch to a negative value, it is set to 0.
b. calculations are performed with single precision integers and floating-point numbers; underflow and overflow will be detected by the operative system and not by WSP

SHARE

When creating boxes that contain lists or tables of data (such as SEQUENcers, FUNCTIon generators, etc), it is possible to specify that they will SHARE the memory allocated to a previously defined box of the same type. Format:

```
?:CREATE box newname [parameters]
:SHARE oldname [C] [S] [T]
?:
```

where

box	is the type of box being created
newname	is the name the new box is to be given
parameters	are the various parameter values required for this particular box type
oldname	is the name the box whose memory is to be shared
C	if present, C indicates that memory allocated for control inputs and outputs will be shared - this means that whenever a signal is connected to or from a control belonging to one box, it will automatically be connected to the equivalent control belonging to the other box as well; if C is **not** present somewhere in the list, box 'newname' will be allocated its own unique controls.
S	if present, S indicates that memory allocated for switches will be shared; this applies only to switches defined in the second and subsequent lines of a CREATE command - all switches defined in the first line of a CREATE command are unique to the box being created; if S is **not** present somewhere in the list, box 'newname' will be allocated space for its own unique switch inputs and outputs
T	if present, T indicates that memory allocated for triggers will be shared; this applies only to triggers defined in the second and subsequent lines of a CREATE command - all triggers defined in the first line of a CREATE command are unique to the box being created; if T is **not** present somewhere in the list, box 'newname' will be allocated space for its own unique trigger inputs and outputs

The number of segments or cells in the new box must be either the same as the number of segments or cells in the shared box, or defined with the default sign % .
The purpose of this facility is primarily to save space in memory; the new box makes use of the tables associated with the SHAREd box, which means that both boxes are affected when

a) modifications are made to one of the boxes
b) connections are made to or from one of the boxes (except when connections are made from the box outputs)

```
e.g.  ?:CREATE SEQ SQ1 5          !create a sequencer with 5 cells
      :200.  0.5  SW1             !define the 5 cells
      :100.  0.1  SW2
      :200.  0.3  SW3
      :800.  0.8  SW4
      :500.  0.7  SW5
```

```
?:CREATE SEQ SQ2 5          !create a second sequencer
:SHARE SQ1                  !SQ2 is to share SQ1's constants,
                            !but not switches or controls
?:MODIFY SQ2/X1 555.        !modify the first value in SQ2
SQ1/X1 555.                 !the program answers that SQ1 has
                            !been modified
?:MODIFY SQ2/A1 0.5         !modify the first control amplitude
SQ2/A1 0.5                  !in SQ2 - the program answers
                            !that SQ1 has been modified, since
                            !controls are not being shared
?:CREATE SEQ SQ3 5          !create a third sequencer which
:SHARE SQ1 S                !shares tables and switches with SQ1
?:MODIFY SQ3:S5 SWXX        !modify the fifth switch input in SQ3
SQ1:S5 SWXX                 !the program answers that SQ1 has
                            !been modified
```

N.B. a) This facility is available for FUNCTIon generators, SEQUENcers and QUANTIfiers only.
b) SHARE is an optional facility in WSP. It is available only if specified in COMPILE.

COPY

When creating a box that contains lists or tables of data (such as SEQUENcers, FUNCTIon generators, etc), it is possible to specify that it will contain exactly the same values as those currently assigned to a previously defined box of the same type. Format:

```
?:CREATE box newname [parameters]
:COPY oldname
?:
```

where

box	is the type of box being created
newname	is the name the new box is to be given
parameters	are the various parameter values required for this particular box type
oldname	is the name the box whose values are to be copied

The number of segments or cells in the new box must be either the same as the number of segments or cells in the copied box, or defined with the default sign %.

This facility is designed merely to speed up the definition of boxes; the new box can be used as a completely separate entity. For example:

```
?:CREATE SEQ SQ1 5          !create a sequencer with 5 cells
:200.  0.5  SW1             !define the 5 cells
:100.  0.1  SW2
:200.  0.3  SW3
:800.  0.8  SW4
:500.  0.7  SW5
?:CREATE SEQ SQ2 %          !create a second sequencer
:COPY SQ1                   !copy SQ1's tables into SQ2
```

N.B. COPY is an optional facility in WSP. It is available only if SHARE and COPY have been specified in COMPILE.

END

Used in the creation of FUNCTIon generators, SEQUENcers and QUANTIfiers to indicate that the segment just defined is to be the final one in this box. This is to allow the creation of boxes of whose size the user does not which to define immediately. For example:

```
?:CREATE FUNC F1 %  !create a function generator of indefinite size
FIRST BREAKPOINT: 0.8
DURATION CURVE BREAKPOINT TRIGGER
SEGMENT 1: 2.5  1.  0.9   !define first segment
SEGMENT 2: 1.5  4.5 0.4   !define second segment
SEGMENT 3: 0.5 -3.  0.7   !define third segment
SEGMENT 4: 1.5  2.  0.8   !define fourth segment
SEGMENT 5: 2.5  0.  0.0   !define fifth segment
SEGMENT 6: END            !now the size is defined: 5 segments
```

UND

Used in the CREATion and MODIFication of FUNCTIon generators, SEQUENcers and QUANTIfiers to indicate that the remaining segments are to be unchanged, i.e. they are to retain their current values, or receive default values if they have not yet been defined at all. For example:

```
?:MODIFY F1 %              !modify an existing function generator
FIRST BREAKPOINT: 0.8      !new value
DURATION CURVE BREAKPOINT TRIGGER
SEGMENT 1: 2.2  1.  0.88   !new values in first segment
SEGMENT 2: 1.5  3.5 0.43   !new values in second segment
SEGMENT 3: UND             !leave the rest unchanged
```

CLE

Used in the CREATion and MODIFication of FUNCTIon generators, SEQUENcers and QUANTIfiers to indicate that the remaining segments are to receive default values. For example:

```
?:MODIFY F1 %              !modify an existing function generator
FIRST BREAKPOINT: 0.8      !new value
DURATION CURVE BREAKPOINT TRIGGER
SEGMENT 1: 2.2  1.  0.88   !new values in first segment
SEGMENT 2: 1.5  3.5 0.43   !new values in second segment
SEGMENT 3: CLE             !set all the remaining segments to
                           !default values, including switches
                           !and triggers associated with each
                           !segment
```

ABBREVIATIONS

All words that are defined by the system (i.e. command words, boxtypes, and the parameters used in the commands SET and SHOW) may be written in shortened form by omitting letters from the end of the word. The user is required to write only as many letters as distinguish a system-word from all other system-words that might be used in the particular context.

For example: E, EX and EXI are interpreted as the same command as EXIT. The single letter C, on the other hand, would be ambiguous (CLEAR or CREATE); in this case at least two letters are required to distinguish the command (CR or CL, for instance).

Note, however, that user-defined names take precedence over the shortened command names. Suppose that the user creates a box call EX. When EX is later written at the beginning of a command line, it will be interpreted as referring to the user-defined box, and not to the system command EXIT. E and EXI will continue to function as EXIT.

Exceptions:

END
UND
CLE
OLD
SHARE
COPY
All SWITCH values (ON, OFF, LIN, EXP, etc - see under Boxes SWITCH)

RUNNING WSP

a) Start the program by typing (after the operating system's $):

```
RUN [WSP]BADA
```

b) Open a sound file, make connections between oscillators, and set channel output levels with:

```
function name: TAPE (to open a .DAC file)
file number: 1      (for example, if file is to be TAPE01.DAC)
function name: CDA  (to set amplitude on channel distributor)
CD no., channel, amplitude: 1 1 1.0
function name: FMCON (to make FM connections)
.
.
```

c) Enter WSP with:

```
function name: WSP
start from scratch (y/n): y  (to initialize all data fields
                              and clear boxes)
```

Now the default connection file [WSP]SYSCON.WSP is read in. This contains definitions of

```
10 DISPLAy boxes, with the SWITCHes and CONNECtion boxes
        necessary to control them
6  INFO boxes, with the SWITCHes and CONNECtion boxes
        necessary to control them
```

If you do not wish to use these, do CLEAR immediately. Note that this file also turns OFF the SYNTHE flag, which means that no calculations are made and nothing is written to the .DAC file until it is turned ON.

d) At the end of the run type

```
EXIT        (or CTRL Z)
```

followed by:

```
function name: ENDPLY    (to close the .DAC file)
```

****
* COMMANDS *
****

COMMANDS

```
CALL     - reads a user file from disk
CLEAR    - deletes all boxes
CREATE   - creates a box
EXIT     - exits from WSP
HELP     - displays information at the user's terminal
MODIFY   - modifies or displays a previously defined box
RENAME   - renames a user-defined box
SAVE     - creates a disk file containing current box data
SET      - sets various system parameters and flags
SHOW     - displays information at the user's terminal

boxtype  - same as CREATE or SHOW
boxname  - same as MODIFY or SHOW
```

Commands consist of a command word taken from the list above, followed by a series of parameters.

Commands are written at the terminal after the prompt' **left angle bracket** '. The left angle bracket is substituted by the characters' ?: ' througth this whole manual.
When a command must be written on more than one line, the second and subsequent lines are written after the prompt' : '.

Empty lines may be inserted at will.

FORMAT

Each word or parameter in the command line must be separated by :

EITHER one or more spaces
OR a comma (with optional spaces).

In the command line, the user may leave a parameter undefined:

EITHER by typing two commas together (with optional spaces)
OR, if the parameter would otherwise be the last parameter in the command line, by simply omitting it.

For example, suppose that a full command line would be:

```
      ?:CREATE THIS THAT AND THE OTHER
or    ?:CREATE   THIS   THAT   AND   THE   OTHER
or    ?:CREATE,THIS,THAT,AND,THE,OTHER
or    ?:CREATE, THIS, THAT, AND, THE, OTHER
```

or any other combination. Write:

```
?:CREATE THIS THAT AND THE      to leave out OTHER
?:CREATE THIS THAT,,THE OTHER   to leave out AND
?:CREATE THIS THAT AND          to leave out THE and OTHER
?:CREATE THIS THAT,,,OTHER      to leave out AND and THE
```

CALL

```
?:CALL [filename] [version]
```

filename — name of the file to be read from disk. If the name contains no directory specification, the specification defined in the SET DEFAUL command is used. If filename contains no file extension, the default extension .WSP is used. If 'filename' is not defined, the program displays the format required to CALL a file. When working with the HP, users may write directory specifications in either VAX format e.g. [USER.DIREC], or HP format e.g. /USER/DIREC/; the HP format ::USERDIREC may be used only if specified directly as part of '**filename**'; it will not work if defined thus in a SET DEFAUL command.

version — an integer giving the version number of the file to be read. If not defined, the most recent version of the named file will be read.

The files that can be read with the CALL command consist of WSP commands in exactly the same format as those written at the terminal. These files are created either by the user, or by the program's SAVE command.

The contents of the file are displayed at the terminal if the ECHO flag is ON. This flag is controlled by the SET ECHO/NOECHO command, and by 'CTRL N'.

Examples

```
?:SET DEFAUL [USER.DIREC]
?:CALL TUT.DAT 15       calls version 15 of 'TUT.DAT' on directory
                        [USER.DIREC]
?:SET ECHO              turn on the ECHO flag
?:CALL [MYFILES]PLING   calls the latest version of [MYFILES]PLING.WSP
                        and displays its contents at the terminal
```

CLEAR

```
?:CLEAR
```

Deletes all boxes.

CREATE

?:[CREATE]boxtype boxname [parameters]

boxtype	the name of one of the box types described under the heading 'Boxes'. The user need type only as much of the word as is necessary to distinguish it from other box types.
boxname	a user-defined name (max 6 characters) with which this box is to be associated. If there is already a box which has the specified name, then: a) if it is of the same boxtype as the one in this command, then it is modified by the parameter values given here and no new box is created b) if it is of a boxtype different from the one specified in this command, an error condition occurs and no new box is created
parameters	an optional list of names and values that define this particular box

This command creates a box of the specified type, with the specified name, and with the parameter values given here.
Note:

1) the word CREATE need not be written
2) parameters which are to have default values may be indicated either by commas (,,) or by the sign %
3) ?:CREATE boxtype (i.e. no user-name or parameter list) can be used to display information about the correct format for creating this particular boxtype

As soon as a box has been created, it starts to function, and continues to function until the end of the run; once created, a box cannot be destroyed, except when all boxes are destroyed with the command CLEAR. Boxes may, however, be modified with the command MODIFY.

Examples

```
    ?:CREATE TRIG T1                          creates a TRIG box called T1
    ?:CREATE RANDOM RA1,,,T3 543987321        the first two parameters get
                                              default values
    ?:TRIG T4                                 creates a TRIG box called T4
    ?:CREATE RANDOM                           display the required format
                                              for CREATing a RANDOM box, i.e.
  *** CREATE RANDOM boxname [speedswitch],[distswitch],[trigger],[seed]

    ?:CREATE TR ENVELOPECONTROL               creates a TRIG box called
                                              ENVELO
```

The last command is accepted by the program, though only the first six letters of the name (ENVELO) are in fact assigned to the TRIG box. Future references to this box will succeed only if the name ENVELO is used.

EXIT

?:EXIT

Exits from WSP to the system monitor or calling program.
'CTRL Z' performs exactly the same function.

HELP

?:HELP

Prints the message:

'HELP' NOT YET IMPLEMENTED: USE **'SHOW'**

but will eventually be available to provide the user with information about the use of WSP.

MODIFY

?:[MODIFY]boxname[spec] [parameters]

boxname	name of a box previously created by the user, with optional wild-cards (*)
spec	the specification of either a control input/output (e.g. /SPEED, /A3, etc) or a switch or trigger associated with a particular cell or segment (e.g. :S2, :T6, etc)
parameters	an optional list of words and values specifying the changes to be made to the box. The parameters are the same as those required when CREATing a box, unless **'spec'** is defined: if spec is a control input/output, the parameter must be the number to be assigned to it; if spec points to a segment switch/trigger, the parameter must be the name of a previously defined switch/trigger

A description of the specified box(es) is displayed at the user's terminal if no parameters are written after the boxname, or if parameters are written after the boxname and the system parameter VERIFY is in force. The description is in the form:

boxtype boxname parameters !other information

where **'other information'** might be the current values of the box's signal inputs and outputs, and the names of any connection boxes this box is associated with.

Note 1) parameters that are to remain unchanged can be indicated with commas (,,)
2) parameters can be returned to the default state with the sign %
3) the previous contents of the box in question are lost when new parameter values are given
4) the word MODIFY need not be written
5) a wild card (*) may be used to indicate that several boxes are to be modified and/or displayed. There are certain restrictions:
   - boxes that require two or more input lines for their creation and modification cannot be modified with the help of a wild card, though they can be displayed
   - error messages are not displayed when a wild card is used, unless no boxes are found with the specified group of letters; so, if there are boxes whose names match the wild-card specification but whose format precludes their being modified by the given parameters, no error message is displayed, but these particular boxes are not modified.

Examples

?:MODIFY CON1 ENV1,,,ADDER	modifies box CON1 such that its first parameter becomes ENV1, its second and third parameters are unchanged, and its fourth parameter is changed to ADDER. The fifth and subsequent parameters remain unchanged.
?:CON1,,BIP %	changes the second parameter of box CON1 to BIP and the third parameter to its default status. The remaining parameters are not changed.
?:VV*,,5.3	changes the second parameter of all boxes whose names begin with VV
?:*BR*,,,%	sets to their default states the third parameters of all boxes whose names contain the letters BR
?:CON1	displays a description of box CON1
?:XXP/A2 0.1	puts 0.1 into the /A2 control input of box XXP
?:*CON	displays descriptions of all boxes whose names end with the letters CON
?:*XX/S	displays the current values of the /S signal point of all boxes whose names end with XX
?:FUNC1:T5 TR2	puts trigger TR2 into the fifth segment of box FUNC1
?:*GG?:S1	displays the names of the switches associated with the first segments of all boxes whose names include 'GG'

RENAME

?:RENAME oldname newname

oldname the user-assigned name of an existing box (or a wild-card specification referring to a group of boxes)

newname the name to be assigned instead of 'oldname'; if a wild-card is used in 'oldname', the wild-card in 'newname' must be EITHER of the same type (though the number of letters need not be the same), OR of type 4 or 5, indicating that the group of letters specified in 'oldname' is to be deleted from every name.

Before executing a RENAME command with wild-cards, the program makes three checks:

1) Do the wild-card specification types in 'oldname' and 'newname' break the rules described above?
2) Will the renaming produce any names that are longer than six characters or shorter than one character?
3) Will the renaming result in the duplication of existing names?

If the answer to any of these questions is 'Yes', an error message is displayed at the terminal, and the command is not executed.

Examples

?:RENAME SEQ1 TAPS	the box called SEQ1 becomes TAPS
?:RENAME *D$* *DT1*	in every box whose name includes the group of characters D$, the first occurrence of

```
                                D$ is replaced by DT1
?:RENAME BB*  *                  in all names that begin with BB, BB is
                                deleted
```

SAVE

?:SAVE [filename] [box-spec]

filename — name to be given to the new file. If the name contains no directory specification, the specification defined in the SET DEFAUL command is used. If filename contains no file extension, the default extension .WSP is used. If 'filename' is not defined, the program displays the the format of the SAVE command. When working with the HP, users may write directory specifications in either VAX format e.g. [USER.DIREC], or HP format e.g. /USER/DIREC/; the HP format ::USERDIREC may be used only if specified directly as part of '**filename**'; it will not work if defined thus in a SET DEFAUL command.

box-spec — a list of the boxes, with or without wild-cards, to be saved on the output file. If no box-specification is given, all boxes created during the run (or since the last CLEAR) are saved.

SAVE writes a text file with the specified name containing information about some or all of the boxes created during the run. Files created with SAVE can later be read with CALL. For every box saved, the following information is written to the file:

a) a CREATE command of exactly the same type as would be needed to create the box if working interactively from the terminal
b) as many MODIFY commands as are required to describe the current values of the box's control inputs and outputs; however, if a control has its default value or is the output of any CONNEC box currently in the system, no MODIFY command is written for it

No box is written to the file more than once, even if referred to in more than one of the box-specifiers. The order in which the box-specifiers are written is of no significance.

The output file is displayed at the user's terminal if the ECHO flag is ON. This flag is controlled by the SET ECHO/NOECHO command, and by 'CTRL N'.

Example

```
?:SAVE ROBINHOOD  XX*  *YY*  LK3  *VBB
```

creates a disk file called ROBINHOOD.WSP, containing:

a) all boxes whose names begin with XX
b) all boxes whose names contain YY
c) the box called LK3
d) all boxes whose names end with VBB

SET

?:SET k [p]

k — one of the key-words

- VERIFY - enables display of boxes after MODIFY
- NOVERI - disenables display after MODIFY
- ECHO - enables display of text files while they are being written with SAVE or read with CALL
- NOECHO - disenables display of text files
- SYNTHE - enables transfer of data to synthesizer
- NOSYNT - disenables transfer of data
- DEFAUL - defines the default directory to be used in CALL, SAVE and SHOW FILE commands; the directory specification must be written in the operating system's normal format, e.g. for the VAX:

  ?:SET DEFAUL [USERDIREC]

  or for the HP:

  ?:SET DEFAUL /USERDIREC/

  DEFAUL can be'**undefined**' by typing:

  ?:SET DEFAUL

  The VAX format [] may be used on the HP as well.
- CLOCK - clears and starts an internal stop-watch, which can later be examined with SHOW CLOCK

possibly, in the future

- SRATE - sampling rate in Hz
- STIME - studio sampling time
- CHANS - number of output channels

p — parameter value, at present required only for SET DEFAUL

SHOW

?:[SHOW]p

p one of the following key-words (only as many letters need be typed as make the key-words unambiguous):

BOXTYP	- list names of all available box types
CLOCK	- displays the time in seconds since SET CLOCK was last done, or since the start of the run if SET CLOCK has not been done
COMMAN	- list names of all available commands
DATA	- display information about common blocks ADATA, IDATA, LDATA and CDATA: how many elements are in use, and how many are free
DEFAUL	- display the name of the current default directory, as defined by SET DEFAUL
ECHO	- current status: ECHO or NOECHO
FILE filename	- display the contents of the file called '**filename.WSP**'; see under CALL for a description of the rules for specifying directory names. Note that the ECHO flag is automatically turned ON when the SHOW FILE command is given; it is not returned to its original state after execution.
NAMES	- list names of all boxes created by the user
SYNTHE	- current status: SYNTHE or NOSYNT
TEXT	- displays everything that follows the word TEXT in the current input line, except for comments
TIME	- displays the current time in hours, minutes and seconds
VERIFY	- current status: VERIFY or NOVERI
boxtypes	- (i.e RANDOM, CONNEC, FUNCTI, etc) list the boxnames of all boxes of this type
boxname	- display the contents of the named box. Wild cards (*) may be used to display groups of boxes.
boxname/spec	- display the contents of the specified control input/output. Wild cards (*) may be used to display groups of boxes.
boxname:spec	- where spec is of the form Sn or Tn, and n is a number pointing to a cell or segment in a FUNCTIion generator, SEQUENcer or QUANTIfier - display the name of the switch or trigger associated with the specified box and cell. Wild cards (*) may be used to display groups of boxes.
command	- displays the format required to use the specified command, which should be one of the WSP command words CREATE, CLEAR, EXIT etc
STATUS	- displays the current system status in the following format:

BOXTYPE	IN USE	FREE	ADATA	IDATA	LDATA
OSCILL	n	max-n	2	0	0
TDELAY	n	max-n	nsaves+1	0	nsaves
RANDOM	n	max-n	2	0	0
.	.	.	.	.	.
.	.	.	.	.	.
TDIVID	n	max-n	0	outputs	0

ADATA n / max-n ; IDATA n / max-n ; LDATA n / max-n ; CDATA n /max-n

where 'n' is the number of boxes created within each box-type, and 'max' is the maximum number of boxes that are permissible. The columns on the right show how many elements of the COMMON data fields are required by each additional box. If a particular box-type is not being used at all, it is not included in the display.

Examples:

?:SHOW COM	print names of all commands
?:BOX	print names of all boxtypes
?:SHOW CONNEC	print names of all CONNECtion boxes
?:SHOW *PLO*	display all the boxes whose names include the letters PLO
?:SHOW X?:T4	display the names of the triggers associated with cell %4 of all boxes whose names begin with X
?:FILE DEF!	displays the contents file DEF!.WSP
?:SHOW TEXT HALLO!	prints HALLO! at the terminal

```
**********************************************************************
*                              BOXTYPES                              *
**********************************************************************
```

BOXTYPES

WSP contains the following boxtypes:

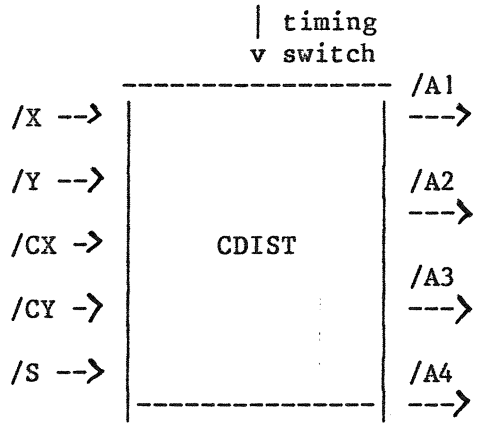
CDISTRibutor	quadraphonic channel distributor
CONNECtion	makes a connection between two signal points
DISPLAy	displays signals graphically
FUNCTIon	multi-segment function generator
IF	compares two input signals
INFO	displays information about specified signal points
LIMIT	adjusts an arbitrary number of input values so that their sum does not exceed a given limit value
MATH	performs mathematical functions on signals
MIX	mixes any number of signals to one output
OSCILLator	sound generator/oscillator
PFUNC	generates periodic functions (SINE, TRIANGLE and SQUARE)
QUANTIfier	quantifies a signal in the range 0 to 1 to one of a specified list of real-number quantities
RANDOM	random number generator
RECORD	stores a signal in successive cells of a SEQUENcer, FUNCTIon generator, QUANTIfier or USER box
SDELAY	sends an input signal to several outputs, each with its own delay time
SEQUENcer	multi-cell sequencer
SIGSWItch	signal-to-switch converter
SLIDE	single-segment function generator
STRING	executes a command string whenever a specified TRIGGEr is ON
SWITCH	multi-directional switch
TDELAY	delays a TRIG pulse for a specified time
TDIVIDe	sends a TRIG pulse to several TRIG boxes simultaneously
TRIGGEr	TRIG pulse generator
TSELECt	sends a TRIG pulse to one of several specified TRIG boxes
USER	an **'empty'** box which may be filled by the user with FORTRAN code
VALUE	single value generator

The characteristics of all box types except USER are system-defined (i.e. they have a fixed number of inputs and outputs, as well as a fixed algorithm for calculating their outputs). However:

* it is not necessary for the user to define every input and output, since the system can always substitute default signals or values

* some box types require the user to define the number of inputs and/or outputs; for example, the MIX box-type mixes any number of input signals to one output signal. But once the number of inputs has been defined for a box, that particular box cannot be altered to accommodate a larger number of inputs; it may, however, be altered to mix fewer inputs. Other MIX boxes, with different numbers of inputs, may of course be created.

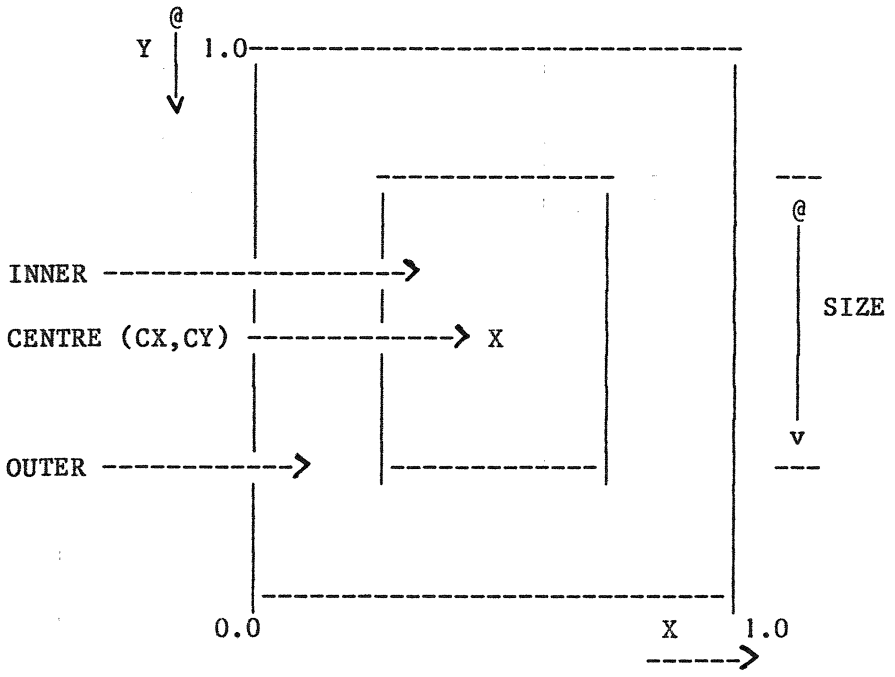
CDISTRibutor



?:CREATE CDIST boxname [timingswitch]

boxname: a unique user-defined name, max 6 characters
timingswitch: name of a previously defined switch that will determine how often this box is to calculate new levels. For example, if the switch has the value 10, the output will be updated every 10th studio sample; if it has the value 2, the output will be updated every 2nd studio sample. Default: updated every sample

Creates a quadraphonic channel distributor which simulates the distribution of sound in a quadratic room with variable room-size and room-centre. The program is based on the following room model:



Control inputs:

/X	position of sound on X-axis: 0 = left wall of outer room, 1 = right wall, default = 0.5
/Y	position of sound on Y-axis: 0 = back wall of outer room, 1 = front wall, default = 0.5
/CX	position of room-centre X-axis: default = 0.5
/CY	position of room-centre Y-axis: default = 0.5
/SIZE	length of the walls of the inner room in relation to the length of the walls of the outer room; e.g. 1.0 = inner room is same size as outer room 0.5 = inner room walls are 0.5 * length of outer room walls, etc

Control outputs (write-protected):

/A1	current amplitude on channel 1
/A2	current amplitude on channel 2
/A3	current amplitude on channel 3
/A4	current amplitude on channel 4

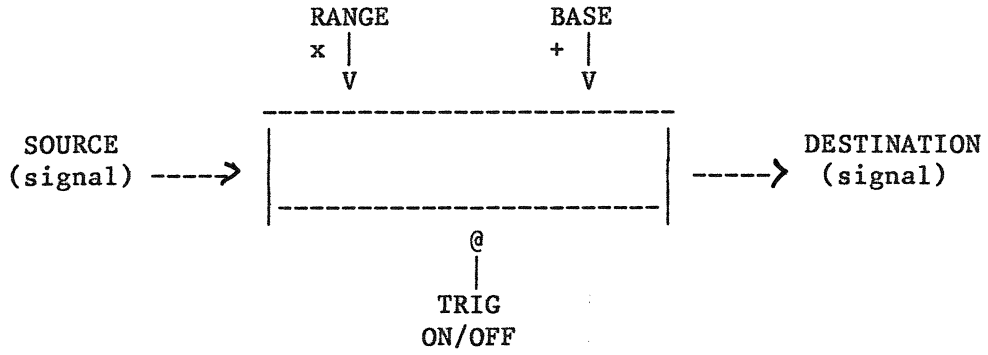
N.B. a) There are no theoretical limits for the values of any of the control inputs, except that /SIZE is automatically kept above 0.01.

b) In the inner room, amplitudes are calculated for all four channels. When (X,Y) lies outside the inner room, at least two channels have amplitude zero.

Example:

```
?:CREATE SWITCH TIMING 10  !create a CDIST box to be updated
?:CREATE CDIST CD1 TIMING  !every 10th studio sample
?:CONNEC CON21 BOX1 CD1/X  !BOX1 will control the X-axis
?:CONNEC CON22 BOX2 CD1/Y  !BOX2 will control the Y-axis
?:CD1/SIZE 0.8             !set the size of the inner room
```

CONNECtion



?:CREATE CONNEC boxname [source],[destination],[range],[base],[trig]

Algorithm:
IF (TRIGGER SET) DESTINATION = (SOURCE * RANGE) + BASE

boxname: a unique user-defined name, max 6 characters
source: either a user-defined name that refers to a signal point or a real-number constant (default = 0.)
destination: a user-defined name that refers to a signal point; some signal points are **'write-protected'** (i.e. they may not be specified as the destination of a connection box) - refer to box descriptions for information on this
range: SOURCE amplification: either a user-defined name that refers to a signal point, or a real-number constant; if not defined, SOURCE is not amplified (i.e. default = 1.)
base: a user-defined name or a real-number constant which will be added to the product of SOURCE and RANGE; if not defined, nothing is added to SOURCE (i.e. default = 0.)
trig: the name of a previously defined TRIGGEr which will determine whether or not the connection is to be made; if not defined, the box is in a permanent state of TRIGGEr ON

User-defined names that refer to boxes with more than one signal input/output are defined as **'boxname/spec'**. See the relevant box descriptions for details.

The output of every box can be directed simultaneously to any number of other boxes, and even back to itself. Box inputs, on the other hand, are unique: if several signals are connected to one input, only the last one has any effect and the others are lost. A special addition, or MIX, box must be used if several signals are to be added to the same input. SOURCE and DESTIN may refer to the same signal point.

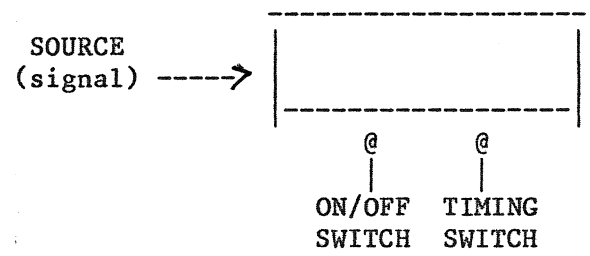
Connection boxes are processed in the order in which they are defined. Connecting several signals to the same DESTIN may under certain circumstances mean that the earlier-defined connections have no effect.

Example:
?:CREATE CONNEC CON1 RAND1 FUNC3/SPEED SEQ2 2.5 TR5

Every time TRIGGEr TR5 is ON, the output of RAND1 is multiplied by the output of box SEQ2, the result is added to the constant 2.5, and this result is put in the control input /SPEED of box FUNC3; when TRIGGEr TR5 is not ON, no data will be moved.

DISPLAy

Graphic display box



?:CREATE DISPLA boxname [on/offswitch] [timingswitch]

boxname: a unique user-defined name, max 6 characters
on/offswitch: name of a previously defined SWITCH. When ON, the input signal is displayed; when OFF, there is no display
default: ON
timingswitch: name of a previously defined SWITCH whose value will determine how often information is to be displayed:
.LT.2 = every studio sample
2 = every second sample
3 = every third sample, etc
default: 0

Control input:
/I input of the signal to be displayed. The value of this signal should lie in the range 0 to 1; values outside this range are plotted at the edge of the screen

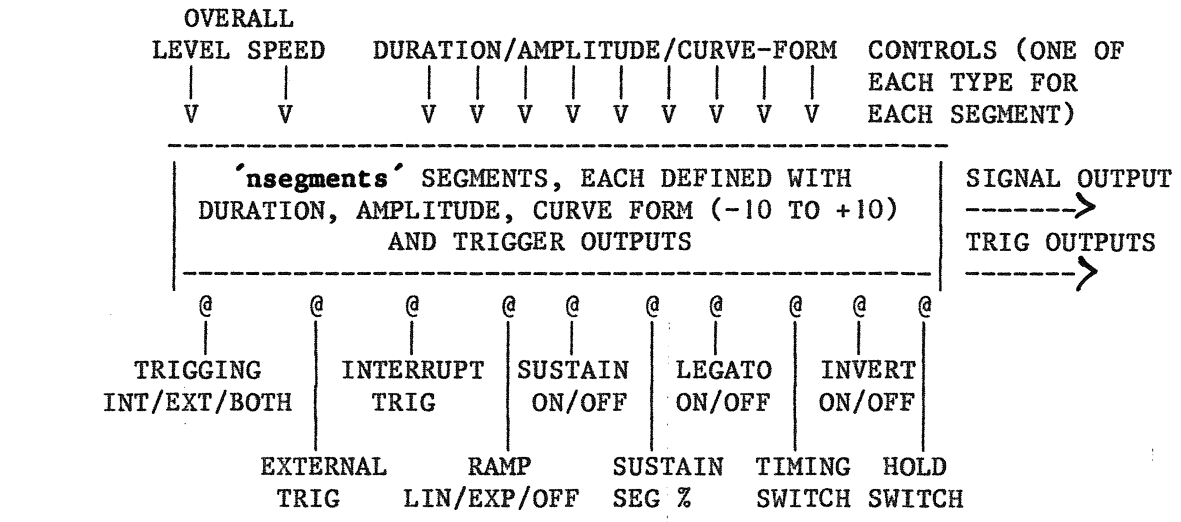
N.B. This boxtype is available only for work on the VAX-11, and should be used only to plot data on the Tektronix T4112 terminal.

Example:

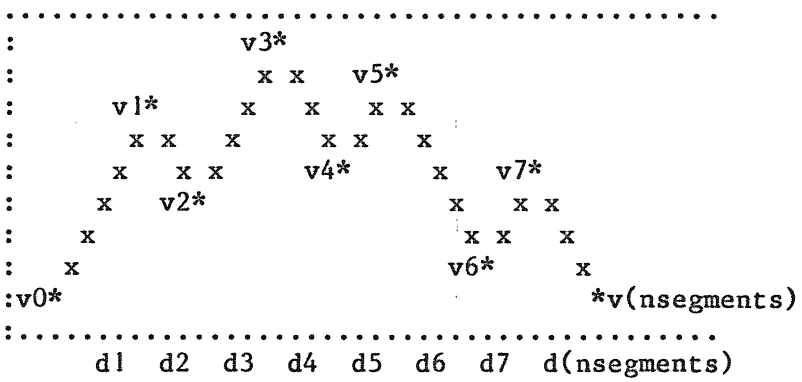
```
?:CREATE SWITCH SW5 5
?:CREATE DISPLA DISP1 SW3 SW5
?:CREATE CONNEC CON5 GEN33 DISP1/I
```

Here we create a DISPLAy box called DISP1, which will be controlled by a previously defined SWITCH called SW3. It will display a value every fifth studio sample (determined by SWITCH SW5). We then create a CONNECtion box called CON5, which connects a previously defined box called GEN33 to DISP1 - DISP1 will now display the signal at GEN33.

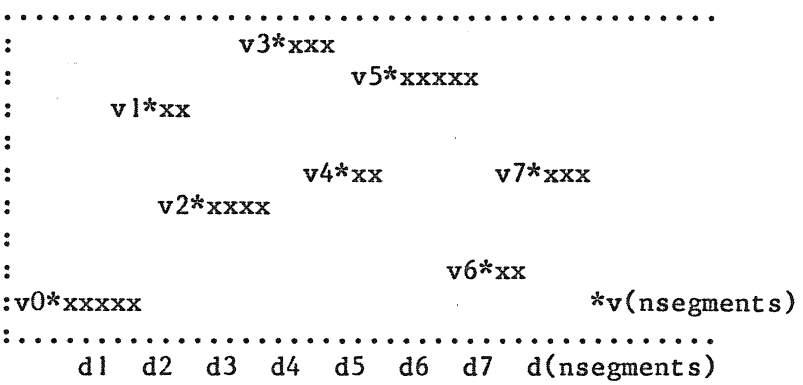
Function generator box



* either * (if RAMP SWITCH = LIN (linear) or EXP (exponential))



* or * (if RAMP SWITCH OFF)



```
?:CREATE FUNCTI boxname [nsegments],[sustainsegnr],[sustainswitch],
 [int/extswitch],[legatoswitch],[holdswitch],[invertswitch],[rampswitch],
 [timing],[trig],[interrupt]
FIRST BREAKPOINT: v(0)
DURATION CURVE BREAKPOINT TRIGGER
SEGMENT 1:[d(1)],[c(1)],[v(1)],[outtrig(1)]
SEGMENT 2:[d(2)],[c(2)],[v(2)],[outtrig(2)]
SEGMENT(n):[d(n)],[c(n)],[v(n)],[outtrig(n)]
```

boxname: a unique user-defined name, max 6 characters

nsegments: number of segments in generator: if not specified, the program reserves as much memory as possible for this box. Input continues until either the reserved space is full or the word END is typed in one of the first three fields of the input line

sustainsegnr: number of the segment which is to be sustained when SUSTAINSWITCH is in the ON position. When the generator reaches the end of the specified segment, the value at the end of the segment is maintained until SUSTAINSWITCH is put to the OFF position. SUSTAINSEGNR may not be greater than NSEGMENTS. If it is less than 1 or undefined, no segment will be sustained.

sustainswitch: name of a previously defined switch which will determine whether or not segment SUSTAINSEGNR will be sustained. default: OFF

int/extswitch: name of a previously defined SWITCH which will control TRIGging of this generator. Position of this switch:
1 (or EXT) external trigging only
2 (or INT) internal trigging only - generator starts automatically as soon as it reaches the end of the final segment or whenever it becomes inactive
3 (or BOTH) external and internal trigging; default: EXT

legatoswitch: name of a previously defined switch which affects the calculation of the generator's first segment ramp:
ON - the first segment ramp starts at the last value calculated for this generator's output
OFF - the first segment ramp starts at the value specified as start value in the function definition
default: ON

holdswitch: name of a previously defined switch:
ON - generator output is frozen
OFF - output calculated normally
default: ON

invertswitch: name of a previously defined switch:
ON - generator output is inverted (v = 1 - v)
OFF - normal output
default: OFF

rampswitch: name of a previously defined switch whose value means:
LIN (or 1) - linear ramps interpolated between the defined breakpoints
EXP (or .GT. 1) - exponential ramps with curve forms determined by the 'c values in the segment definitions
OFF (or zero) - no ramps between the breakpoints
default: LIN

timing: name of a previously defined switch that will determine how often the generator is to output a new value. For example, if the switch has the value 10, the generator's output will be updated every tenth studio sample; if it has the value 2, the output will be updated every second studio sample.
default: 1

trig: name of a previously defined TRIGGEr, which will perform external trigging when INT/EXTSWITCH has the value EXT or BOTH

interrupt: name of a previously defined TRIGGEr, which when turned ON will interrupt the current envelope, i.e. cause the generator to jump immediately to the function's final breakpoint value. If the INT/EXT switch is in the INTernal position, the generator will start immediately from the beginning of the function.

v(0-nsegments):breakpoint values, normally in the range 0 - 1; must include the decimal point; default 0

d(1-nsegments):segment durations in seconds; must include the decimal point; if undefined or less than zero, set to zero automatically by the program; default 0

c(1-nsegments):segment curve forms in the range -10.0 to +10.0; must include the decimal point; default 0

outtrig(1-nsegments): previously defined TRIGgers that will be set to ON when the end of each segment is reached. **'outtrig (1)'** is set at the end of segment 1, **'outtrig(2)'** is set at the end of segment 2, etc.

Control inputs:

/SPEED controls the overall speed of the generator, e.g. 1.0 = normal speed, 2.0 = double speed, 4.0 = four times speed, 0.0 = zero speed (i.e. generator stops - equivalent to HOLDSWITCH ON) default 1.0

/LEVEL controls the overall level of the generator, e.g. 1.0 = normal level, 2.0 = 2 * level, 0.0 = zero level

/An is multiplied by the amplitude of the function's breakpoints. n is the number, in the range 0 to nsegments, of the breakpoint to be controlled by this input.

/Dn controls the speed of individual segments, where n is the number, in the range 1 - nsegments, of the segment to be controlled by this input.

/Cn is multiplied by the curve forms defined for the function. n is the number, in the range 1 - nsegments, of the segment to be controlled by this input.

/Xn is the breakpoint value defined for segment n

/Yn is the curve-form value defined for segment n

/Zn is the duration defined for segment n

Write-protected output:

/O is the signal output of the generator

Segment triggers:

:Tn is the trigger associated with segment n

Keywords:

The following keywords may be written in answer to the question **'FIRST BREAKPOINT:'**

SHARE to share the tables of another FUNCTIon generator with optional qualifiers C (=controls) and T (=triggers)

COPY to copy the tables of another FUNCTIon generator

The following keywords may be written in answer to one of the subsequent questions **'SEGMENT n:'**

UND the remaining segments will be unaltered

CLE the remaining segments will receive default values

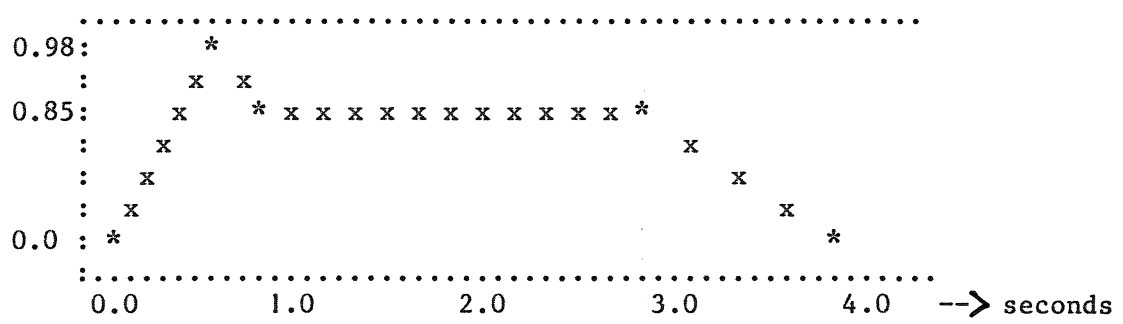
END the generator will contain only those segments that have already been defined

Example:

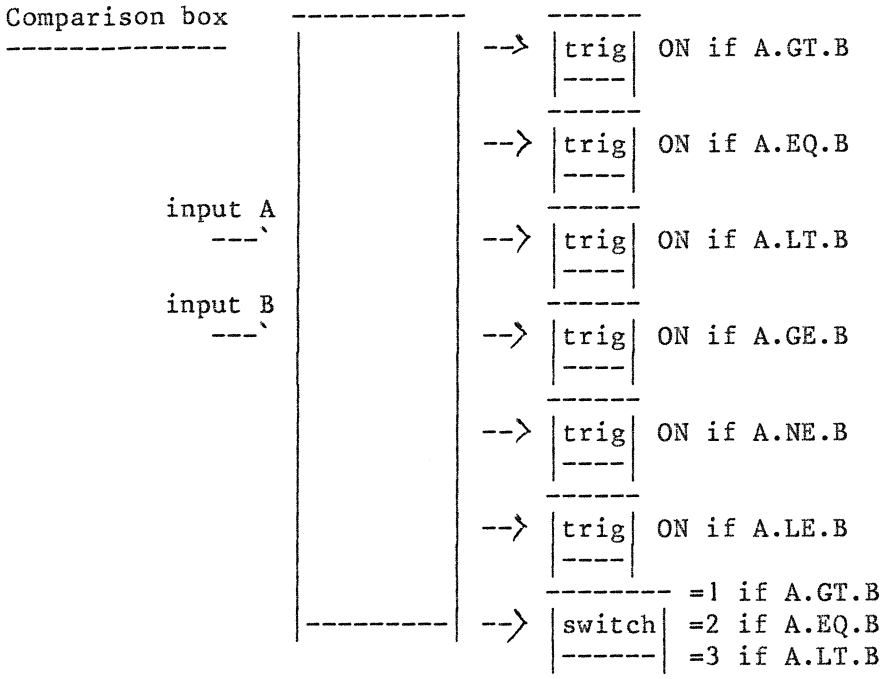
```
?:CREATE FUNCTI FUNC2 4 3 SW2 EXT5,,SW5,,,,TRIG1 BREAK
FIRST BREAKPOINT: 0.0
DURATION CURVE BREAKPOINT TRIGGER
SEGMENT 1: 0.5  1.  0.98  TRIG10
SEGMENT 2: 0.4  1.  0.85  TRIG11
SEGMENT 3: 2.0  1.  0.85  TRIG12
SEGMENT 4: 1.0  1.  0.0   TRIG13
?:CREATE CONNEC CON10 BOX5 FUNC2/SPEED     !control speed
?:CREATE CONNEC CON11 BOX44 FUNC2/A1       !control breakpoint 1
?:CREATE CONNEC CON12 BOX12 FUNC2/D1       !control duration 1
?:CREATE CONNEC CON13 BOX12 FUNC2/D2       !control duration 2
?:CREATE CONNEC CON14 BOX295 FUNC2/C1      !control curve form 1
```

First we create a function generator called FUNC2. It will consist of four segments, INT/EXT trigging is determined by a previously defined SWITCH call EXT5, external trigging comes from a previously defined TRIGGEr called TRIG1, the function can be interrupted by a previously defined switch called BREAK, sustain at the end of segment 3 will be controlled by a previously defined SWITCH called SW2, the ramp, legato, inversion and timing parameters will all have default values.

The breakpoints and durations of the generator are:



Then we create a series of CONNECtion boxes to determine which inputs are to be controlled by which signals: a box called BOX5 will control the SPEED input, BOX44 will control the /A1 input, BOX12 will control both /D1 and /D2 inputs, and BOX295 is connected to the C1 input.



?:CREATE IF boxname switch trig1 trig2 trig3 trig4 trig5 trig6

boxname: a unique user-defined name, max 6 characters
switch: name of a previously defined switch which will be set to 1 when input A is greater than input B, 2 when they have exactly the same value, and 3 when A is less than B
trig1: set to ON when A .GT. B
trig2: set to ON when A .EQ. B
trig3: set to ON when A .LT. B
trig4: set to ON when A .GE. B
trig5: set to ON when A .NE. B
trig6: set to ON when A .LE. B

These triggers are set to ON only when there is a change in the relationship between inputs A and B

Control inputs:
/A the 'A' input signal (default 0.0)
/B the 'B' input signal (default 0.0)

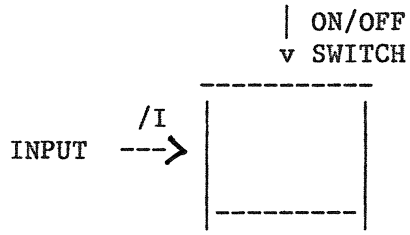
For example:

```
?:CREATE IF IF1 OUTSW T1 T2 T3,,T5
?:CREATE CONNEC CON33 BOX1 IF1/A
?:CREATE CONNEC CON34 BOX2 IF1/B
```

We create an IF box generator called IF1, and connect a previously defined box called BOX1 to its 'A' input, and another previously defined box called BOX2 to its 'B' input. A previously created SWITCH called OUTSW will receive the value 1, 2, or 3 (as described above); TRIGGEr T1 will be set to ON whenever the relationship between A and B changes from A.LE.B to A.GT.B; TRIGGEr T2 will be set to ON whenever the relationship between A and B changes from A.NE.B to A.EQ.B; TRIGGEr T3 will be set to ON whenever the relationship between A and B changes from A.GE.B to A.LT.B; and TRIGGEr T5 will be set to ON whenever the relationship between A and B changes from A.EQ.B to A.NE.B. The other trigger outputs will not be used.

INFO

Information box



?:CREATE INFO boxname on/offswitch timingswitch

boxname: a unique user-defined name, max 6 characters

on/offswitch: name of a previously defined switch: ON = display the current value of this box, OFF = do not display
default: OFF

timingswitch: name of a previously defined switch that will determine how often values are to be displayed. For example, if timing has the value 10, the display will be updated every tenth studio sample; if it has the value 2, the display will be updated every second studio sample.
default: 1

Display is in the following format:

```
11111.11111  22222.22222  33333.33333  44444.44444  55555.55555  66666.66666
```

where 11111.11111 represents the value of the first INFO box (i.e. the first one created), 22222.22222 represents the value of the second INFO box, and so on.

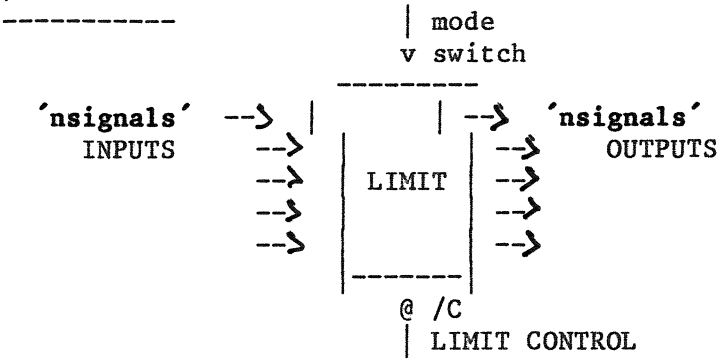
For example:

```
?:CREATE INFO INF1 SW1 SW50
?:CREATE CONNEC CON33 BOX1/S INF1
```

We create an INFO box called INF1, and connect the /S input of a previously defined box called BOX1 to it. A previously created SWITCH called SW1 will control the display (ON/OFF); when SW1 is ON, the value of SWITCH SW50 will determine how often the contents of BOX1/S are to be displayed.

LIMIT

Limiter/normalizer



?:CREATE LIMIT boxname nsignals modeswitch

boxname: a unique user-defined name, max 6 characters
nsignals: integer constant defining the number of input signals; **'nsignals'** outputs are automatically produced
default: 10
modeswitch: name of a previously defined SWITCH, whose value means:
0: OFF, output(n) = input(n)
1: ON, limiter mode -
if the sum of the inputs is greater than the value at the /C input and the sum is not equal to zero, all values are adjusted such that:
output(n) = input(n)*limitvalue/sum
which forces the sum of the outputs = limitvalue.
otherwise, output(n) = input(n).
.GT. 1: NORM, normalization mode -
values are adjusted as in limiter mode, so that the sum of the outputs is ALWAYS equal to the value at the /C input; except when sum = 0., in which case
output(n) = input(n)
default: OFF

Control inputs:
/C limit value (default 0.0)
/In input signal n (where n is in the range 1 to **'nsignals'**)

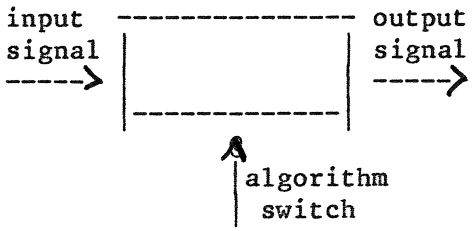
Write-protected control outputs:
/On output signal n (where n is in the range 1 to **'nsignals'**)

Example:

```
?:CREATE SWITCH MODESW 1                    !mode switch - limiter mode
?:CREATE LIMIT LIM1 4 MODESW                !LIMIT box with 4 inputs
?:CREATE CONNEC CON33 BOX1 LIM1/I1          !connect four inputs
?:CREATE CONNEC CON34 BOX2 LIM1/I2
?:CREATE CONNEC CON35 BOX3 LIM1/I3
?:CREATE CONNEC CON36 BOX4 LIM1/I4
?:CREATE CONNEC CON43 LIM1/O1 BOX11         !connected adjusted signals
?:CREATE CONNEC CON44 LIM1/O2 BOX12         !to wherever they are going
?:CREATE CONNEC CON45 LIM1/O3 BOX13
?:CREATE CONNEC CON46 LIM1/O4 BOX14
```

MATH

Mathematic function box



?:CREATE MATH boxname switch

boxname: a unique user-defined name, max 6 characters
switch: name of a previously defined SWITCH whose value will determine the algorithm to be used in calculating the output:

0	OFF	output = input
1	SIN	output = SIN (input)
2	FIX	output = AINT (input) (limited to range -32767 to +32767)
3	SQR	output = SQR (input)
4	EXP	output = EXP (input)
5	COS	output = COS (input)
6	LOG	output = LOG (input)
7	LOG10	output = LOG10 (input)
8	DBVOLT	output = DBVOLT (input) (converts dB intensities in the range 0 - 100 to linear amplitudes in the range 0 - 1)
9	VOLTDB	output = VOLTDB (input) (converts linear amplitudes between 0 and 1 to dB intensities in the range 0 - 100)
10	ABS	output = ABS (input)
11	NINT	output = NINT (input)

default: OFF

Controls:

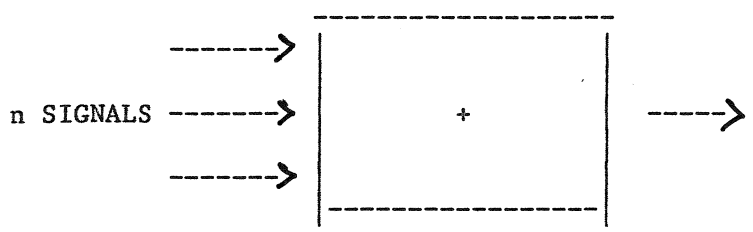
/IN	input signal	(default 0.0)
/OUT	output signal	(write-protected)

Example:

```
?:CREATE SWITCH MATSW DBVOLT
?:CREATE MATH MAT1 MATSW
?:CREATE CONNEC CON33 BOX1 MAT1/IN
?:CREATE CONNEC CON34 MAT1/OUT OSC1/AMP
```

We create a MATH BOX called MAT1, to be controlled by a SWITCH called MATSW, which has the value DBVOLT, or 8. BOX1 is connected to MAT1's input, while its output is connected to the /AMPlitude input of a box called OSC1. Thus the intensities (in dB) generated by BOX1 are converted to a linear amplitude scale before being sent to oscillator OSC1.

MIX



?:CREATE MIX boxname ninputs

boxname: a unique user-defined name, max 6 characters
ninputs: integer, number of input signals to be mixed

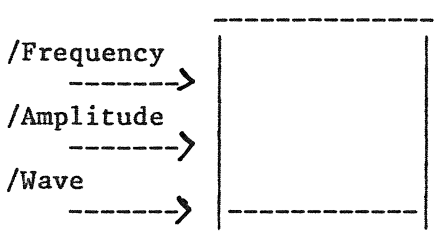
Controls:
/In the nth input signal (default 0.0)
/O signal output (write-protected)
Algorithm:
OUTPUT = I(1) + I(2) + ... + I(n)

For example:
```
?:CREATE MIX MIX3 3
?:CREATE CONNEC CON10 BOXA MIX3/I1
?:CREATE CONNEC CON12 BOXC MIX3/I2
?:CREATE CONNEC CON14 BOXE MIX3/I3
```

Here a MIX box called MIX3 with 3 inputs is created. Then various previously defined boxes are connected to MIX3's inputs. The resulting signal is BOXA + BOXC + BOXE

OSCILLator

Synthesizer oscillator box



?:CREATE OSCILLator boxname

boxname: a unique user-defined name, max 6 characters

Control inputs:

/F oscillator frequency (Hz) in range 0 to samplingrate/2
default 0.0

/A oscillator amplitude, normally in range 0 to 1
default 0.0

/W wave descriptor, in range 0 to 1
Intermediate values between the ones shown here result in interpolations between the wave forms. Thus, for example, a signal value of 0.55 results in a wave-form which is a mixture of wave-forms 4 and 5 in the proportion [0.66666-0.55:0.55-0.5] or about [11:5] . Default 0.0

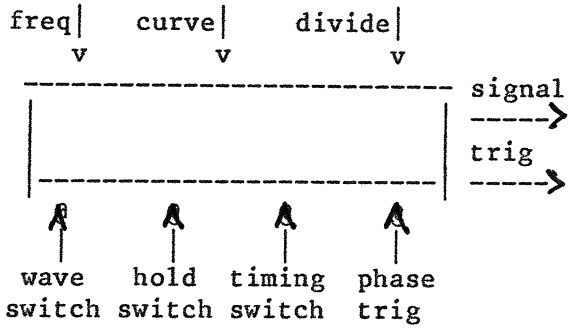
signal value	wave number
1.0	7
0.83333	6
0.66666	5
0.5	4
0.33333	3
0.16666	2
0.0	1

N.B. The wave-form input is not available in the HP version of WSP.
See BADA.HLP for a description of the assignment and use of wave-forms.

Example:

```
?:CREATE OSCILL OSC1                !create an oscillator OSC1
?:CREATE CONNEC CON33 BOX1 OSC1/F !control its frequency
?:CREATE CONNEC CON34 BOX2 OSC1/A !control its amplitude
?:CREATE CONNEC CON35 BOX3 OSC1/W !control its wave-form
```

Periodic function generator



?:CREATE PFUNC boxname waveswitch holdswitch timingswitch phasetrig outtrig

boxname: a unique user-defined name, max 6 characters
waveswitch: name of a previously defined switch which will determine which wave-form this box will generate: 1 SIN (cosine); 2 TRI (triangular); 3 SQR (square); default: SIN
holdswitch: when ON, halts the generator, and freezes its output; default OFF
timingswitch: (optional) name of a previously defined switch that will determine how often the generator is to output a new value. default: 1
phasetrig: name of a previously defined TRIGGEr; when it is ON, the PFUNC's phase will be reset to zero
outtrig: name of a previously defined TRIGGEr which will be set to ON every time the generator reaches the end of a cycle

Controls:

/FRE the generator's frequency in Hz; if the frequency is less than or equal to zero, the generator in effect stops default 0

/CURVE curve-form, or rate of change, for SIN and TRI wave-forms; in the range -10 to +10; Default 0

/DIV divide, or mid-point, for TRI and SQR wave-forms. When the signal at the /DIV input differs from 0.5, the mid-point of the wave-form is shifted away from its normal 180 degree position. At 0.0 or less, the mid-point is at phase 0; at 1.0 or more, the mid-point is at phase 360. A sawtooth wave can therefore be generated by selecting the TRI wave-form and putting the /DIV input to 0.0 or 1.0. default 0.5

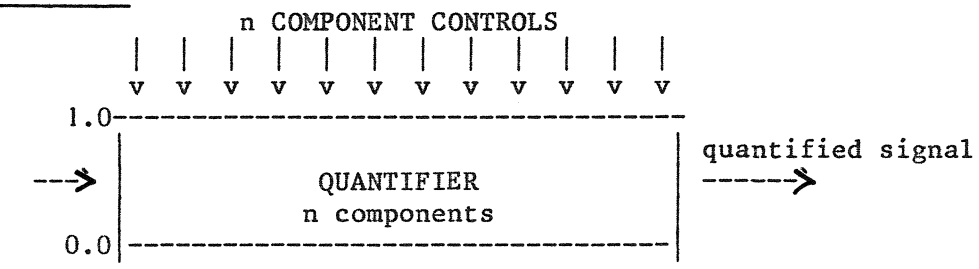
/O signal output

Example:

```
?:PFUNC PF1 PFWAVE PFHOLD PFTIM PFPHAS PFTRIG
?:CONNEC PFFREQ 1. PF1/F
?:CONNEC PFCURV 0. PF1/C
?:CONNEC PFDIV 0.5 PF1/D
```

We create a periodic function generator called PF1, and connect the constants 1.0, 0.0 and 0.5 to its FREquency, CURVE and DIV inputs respectively. The wave-form will be determined by a switch called PFWAVE, the hold and timing functions will be controlled by switches PFHOLD and PFTIM, the phase will be set to zero when TRIGGEr PFPHAS is ON, and a TRIG pulse will be sent to PFTRIG every time the generator reaches the end of a cycle.

Signal quantifier




```
?:CREATE QUANTI boxname [n]
:v(1)  v(2)  ... v(n)
```

boxname: a unique user-defined name, max 6 characters
n: number of components the input signal will be divided into; if not specified, the program reserves as much memory as possible for this box. Input continues until either the reserved space is full or the word END is typed
v: values to be output (default 1.0)

Controls:
/IN the source signal that is to determine which of the components is to be output; default 0.
/Cn component controls, one for each component; default 1.
/Xn component values - i.e. those defined as v(1), v(2), etc
/O signal output (write-protected)

Algorithm:
p = sourcesignal*n+1 [limited to the range 1 to n]
output = v(p)*control(p)

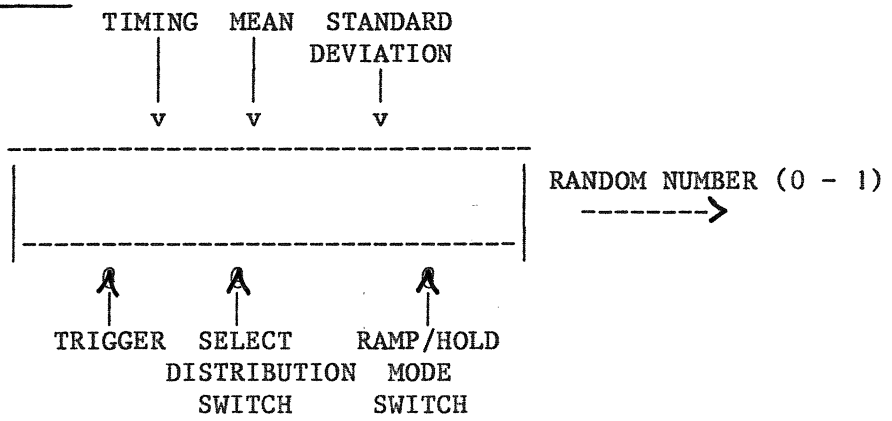
Keywords:
The following keywords may be written first in the second line, immediately after ' : '
SHARE to share the tables of another QUANTIfier with optional qualifier C (=controls)
COPY to copy the tables of another QUANTIfier
The following keywords may replace any component value:
UND the remaining components will be unaltered
CLE the remaining components will receive default values
END the generator will contain only those components that have already been defined (illegal in component %1)

Example :

```
?:CREATE QUANT AMAJOR 7
:220.  246.94  277.18  293.66  329.62  369.98
:415.29
?:CREATE CONNEC CAMAJ TUNER AMAJOR/C3
```

We create a QUANTIfier box called AMAJOR; it is to have 7 components, to which we give the values of the frequencies in the A major scale. When the input signal to AMAJOR is less that 1/7 (or 0.142857) the first component 220. will be output; when the input signal is between 1/7 and 2/7 the second component 246.94 will be output; when the output signal is greater that 6/7 (or 0.85714), the seventh component 415.29 will be output; and so on. We then create a connection called CAMAJ from a previously defined box called TUNER to AMAJOR's third
control input: the output of TUNER will amplify the third component of AMAJOR.

Random number generator



?:CREATE RANDOM boxname distswitch rampswitch trig seed

boxname: a unique user-defined name, max 6 characters
distswitch: name of a previously defined switch that determines which random distribution is to be used. At present:
1 rectangular (SWITCH value: RECT) in the range 0 to 1
2 cutgauss (SWITCH value: CUTG) gaussian distribution restricted to the range 0 to 1
3 binary (SWITCH value: BIN) outputs 0 or 1 ; default: 1
modeswitch: name of a previously defined switch: (default 0)
0 (or OFF): the /TIME input is ignored, and a new random number is calculated every studio sample
1 (or HOLD): a new random number is calculated every s seconds, where is s is the current value of the /TIME Input; meanwhile, the output remains unchanged
2 (or RAMP): a new random number is calculated every s seconds, where is s is the current value of the /TIME input; intermediate values are interpolated between the random numbers
trig: name of a previously defined TRIGGEr - if ON, the generator outputs a new number; otherwise nothing is output. If not defined, the generator outputs continuously
seed: a reel number in the range 0 to 1 which will determine the starting-point of the random number sequence. If no value is given, the program selects a starting-point based on the current time.

Output (write-protected):
/O

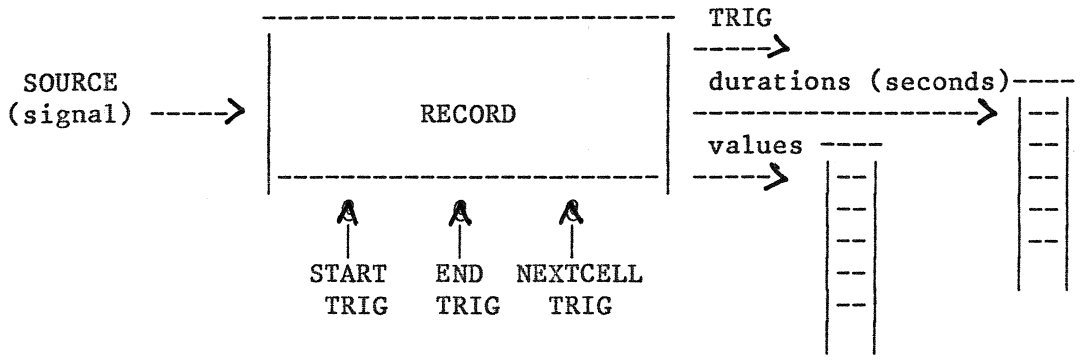
Control inputs:
/DEV determines the standard deviation for distribution type 2 (CUTGAUSS) (default 0)
/MEAN determines the mean value for distribution type 2 (default 0)
/TIME determines how often new random numbers are to be output. See the description of '**modeswitch**' above. If /TIME has the value zero, or if no signal is connected to /TIME input, a new random number is generated every studio sample.

Example :
```
?:CREATE RANDOM R1 SPSWI RDIST
?:CREATE CONNEC CON11 TIMER R1/TIMING
```

A random generator called R1 is created, controlled by two previously defined switches called SPSWI and RDIST, but with no TRIGGEr control. A connection is then made from a previously defined box called TIMER to R1's /TIMING input.

RECORD

Signal recording box



Takes the output of one box and stores the signal in successive cells of a SEQUENCER, QUANTIFIER, FUNCTION GENERATOR, or USER box. Optionally, the durations between changes in the value of the source signal may also be stored.

A RECORD box can be in three states:

HIBERNATING — This is the state the box is in at creation; it will do nothing until **'starttrig'** is set to ON

READY — When **'starttrig'** is set to ON, the box is READY to record. If **'endtrig'** is set to ON, it will return to the HIBERNATING state.

RECORDING — The box will go into the RECORDING state:

1) if **'nextcelltrig'** has not been defined, at once, and the current source value is copied immediately into the first output cell; then, whenever the source value changes, this new value is recorded into the next output cell, at the same time as the duration of the first value is recorded into the first duration cell.
2) if **'nextcelltrig'** has been defined, the box waits in the READY state until **'nextcelltrig'** is set to ON. Now we are in the RECORDING state. The box copies the current source value into the first output cell. The next time **'nextcelltrig'** is set to ON, the time between the trig pulses is recorded into the first duration cell, and the current source value is recorded into the second output cell. And so on.

The box returns to the HIBERNATING state if

1) it has recorded a value and duration in the final specified cell
2) **'endtrig'** is set to ON

Whenever the box returns to the HIBERNATING state, for any reason whatsoever, **'outtrig'** is set to ON.

```
?:CREATE RECORD boxname [source] [value/start] [value/end] [duration/start]
  [starttrig] [endtrig] [nextcelltrig] [outtrig]
```

boxname: a unique user-defined name, max 6 characters

source: either a user-defined name that refers to a signal point or a real-number constant (default = 0.)

value/start: the user-defined name of the signal point that is to receive the first of the source values

value/end: the user-defined name of the signal point that is to receive the last of the source values; it must lie within the same box and parameter as value/start; it may not refer to a lower cell or segment number than value/start.

duration/start: the user-defined name of the signal point that is to receive the first of the durations recorded from source. This signal point need not lie within the same box as value/start and value/end, though there must be sufficient room in the box containing duration/start to accommodate (value/end - value/start + 1) durations. If duration/start is not defined, no duration values are recorded

starttrig: the name of a previously defined TRIGGEr which, when set to ON, puts the RECORD box in the READY state

endtrig: the name of a previously defined TRIGGEr which, when set to ON, puts the box in the HIBERNATING state

nexttrig: the name of a previously defined TRIGGEr which, when set to ON, causes the current source value to be recorded into the next output cell

outtrig: the name of a previously defined TRIGGEr which is set to ON when the box returns to the HIBERNATING state

N.B. RECORD boxes have no inputs or outputs that can be connected with CONNEC boxes.

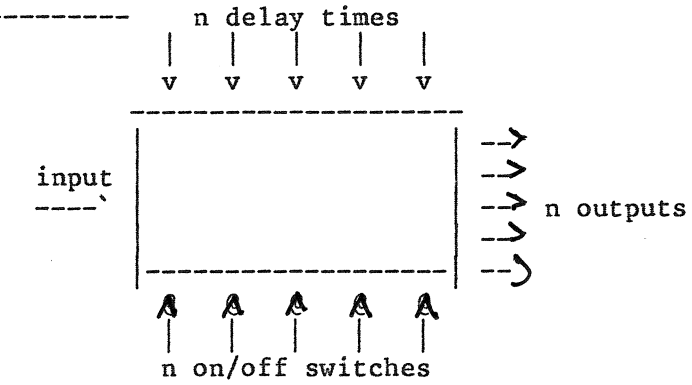
Example:

```
?:TRIG START
?:TRIG END
?:TRIG OUT
?:CREATE RECORD RCREC BOX1 RCQ/X1 RCQ/X10 RCQ/Y1 START END,,OUT
```

When TRIGGEr START is set to ON, the output of BOX1 will be recorded into cells /X1 to /X10 of box RCQ, while the durations (times between the changes in value) will be recorded into the 10 cells starting at RCQ/Y1. **'Nexttrig'** has not been defined, so it the changes in source values which control when new values are transferred. Recording can be interrupted by setting trig END to ON, and when recording is complete, trig OUT will automatically be set to ON.

SDELAY

Signal delay box



?:CREATE SDELAY boxname maxdelay noutputs
noutputs CONTROL SWITCHES:s(1) s(2) ... s(noutputs)

boxname: a unique user-defined name, max 6 characters
maxdelay: the maximum delay time (expressed as a floating-point number in seconds) that will be required; this time determines the amount of memory that will be allocated to this box, approx. 4 bytes per studio sample (0.01 secs); default 0.1 secs
noutputs: the number of signals to be output; default 10
s: names of the switches which will determine whether or not each of the outputs will be updated: only those outputs whose switches are ON are calculated. (default: ON)

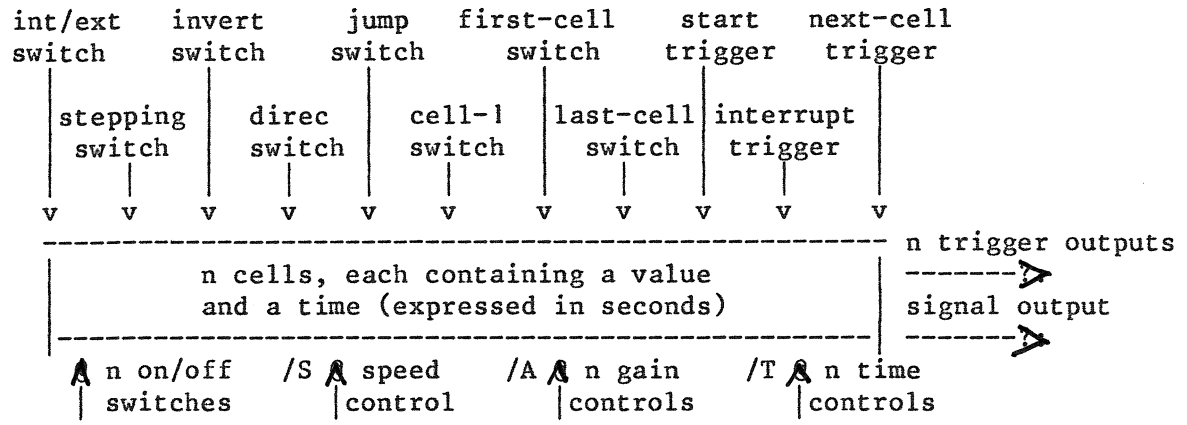
Control inputs:
/I input signal
/Dn the delay time, in seconds, for output n (default 0)
Write-protected output:
/On output signal n
Control switches:
:Sn the switch associated with output n

For example:

```
?:CREATE SDELAY SD1  0.1  4              !4 outputs, max delay 0.1 secs
4 CONTROL SWITCHES:SW1  SW2  SW3  SW4 !on/off switches
?:CREATE CONNEC CON11 BOX1 SD1/I        !input signal
?:CREATE CONNEC CON12 BOX2 SD1/D1       !boxes 2-5 are connected as
?:CREATE CONNEC CON13 BOX3 SD1/D2       !delay controls
?:CREATE CONNEC CON14 BOX4 SD1/D3
?:CREATE CONNEC CON15 BOX5 SD1/D4
?:CREATE CONNEC CON22 SD1/O1 BOX12      !the 4 outputs are connected
?:CREATE CONNEC CON23 SD1/O2 BOX13      !to boxes 22-25
?:CREATE CONNEC CON24 SD1/O3 BOX14
?:CREATE CONNEC CON25 SD1/O4 BOX15
```

See also the example SDELAY under the heading **'Examples'**.

Multi-cell sequencer




```
?:CREATE SEQUEN boxname [ncells],[int/ext],[stepmode],[invert],[direc],
   [jump],[cell-1],[first-cell],[last-cell],[starttrig],[intrpt],[next-cell]
VALUE TIME(SECS) TRIGGER
CELL 1: v(1)  t(1)  switch(1)  trigout(1)
CELL 2: v(2)  t(2)  switch(2)  trigout(2)
 .
 .
CELL ncells: v(ncells)  t(ncells)  switch(ncells)  trigout(ncells)
```

boxname: a unique user-defined name, max 6 characters

ncells: integer value defining the number of cells this box is to consist of: if not specified, the program reserves as much memory as possible. Input continues until either the reserved space is full or the word END is typed in one of the first two fields of the input line

int/ext: name of a previously defined SWITCH:
EXT (1) = external trigging only (default)
INT (2) = internal trigging only
BOTH (3) = internal and external trigging

stepmode: name of a previously defined SWITCH which will determine whether stepping from one cell to the next is to be
a) externally triggered (EXT or 1) - default
b) automatically triggered when the time associated with each cell has elapsed (INT or 2)
c) a combination of (a) and (b) (BOTH or 3)

invert: name of a previously defined SWITCH: when ON, the sequencer output is inverted (output = 1.0 - output) (default OFF)

direc: name of a previously defined SWITCH: when it has the value FORE (or 1), the direction of the sequencer is forward; when it has the value BACK (3 or greater), the direction is backwards. When it has the value 0 or 2, the sequencer halts (default 1)

jump: name of a previously defined SWITCH whose value is added to the sequencer's **'cell-pointer'** whenever a step to a new cell occurs:
nextcell = thiscell + jump (if direc = FORE)
nextcell = thiscell - jump (if direc = BACK) (default 1)

cell-1: name of a previously defined SWITCH whose value determines which cell the sequencer is to start at when TRIGGEr **'starttrig'** is ON; if cell-1 has the value OFF or 0, the sequencer will start at **'first-cell'** when direc = FORE, or **'last-cell'** when direc = BACK. default: 1

first-cell: name of a previously defined SWITCH whose value determines the **'left-hand end'** of the sequencer; with **'first-cell** and **'last-cell** it is possible to redefine continuously exactly which part of the sequencer is to be used ; default: 1

last-cell: name of a previously defined SWITCH whose value determines the **'right-hand end'** of the sequencer
default: ncells

starttrig: name of the external TRIGGEr which will start a new sequence when set to ON; the sequence starts at cell **'first-cell'** if **'direc'** is FORE, or at cell **'last-cell'** if **'direc'** is BACK.

intrpt: name of a previously defined TRIGGEr which will interrupt the current sequence when set to ON: if the int/ext SWITCH has the value INT or BOTH, a new sequence will start immediately; otherwise a new sequence will start only when **'starttrig'** is set to ON

next-cell: name of a previously defined TRIGGEr which, when **'stepmode'** is INT or BOTH, causes a jump to the next cell; the jump itself is defined by the **'jump'** SWITCH

v: real-number value in each cell (default 0)

t: cell duration in seconds (default 0)

switch: name of a previously defined SWITCH which will determine whether or not this particular cell is to be used. OFF (0) = not to be used; ON (non-zero) = to be used (default ON)

trigout: name of a previously defined TRIGGEr which will be set to ON when the **'cell-pointer'** jumps from the relevant cell to another cell

Control inputs:

- /S a signal which controls the sequencer's overall speed 1.0 = normal speed, 0.5 = half speed, 2.0 = double speed, etc
- /An a signal which is multiplied by the real-number value of cell n when the cell-pointer points to cell n
- /Tn a signal which is multiplied by the time value of cell n, the result being the actual duration in seconds
- /Xn the value defined for cell n
- /Yn the duration defined for cell n

Write-protected output:

- /O

Control switches:

- :Sn the switch associated with cell n

Triggers:

- :Tn the trigger associated with cell n

Keywords:

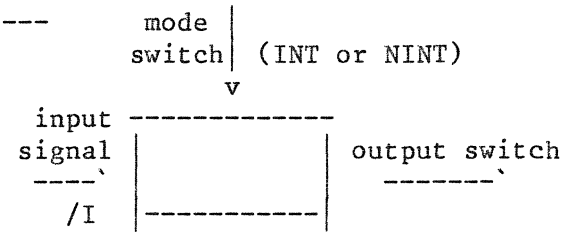
The following keywords may be written in answer to the first question **'CELL 1:'**

- SHARE to share the tables of another SEQUENcer with optional qualifiers C (=controls), S (=switches) and T (=triggers) .
- COPY to copy the tables of another SEQUENcer

The following keywords may be written in answer to any of the questions **'CELL n:'**

- UND the remaining cells will be unaltered
- CLE the remaining cells will receive default values
- END the sequencer will contain only those cells that have already been defined (illegal in cell %1)

Signal-to-switch converter



?:CREATE SIGSWI boxname [outswitch],[mode]

boxname: a unique user-defined name, max 6 characters
outswitch: name of a previously defined SWITCH which will receive a value equivalent to the value of the input signal; if the input signal is less than 0.0, outswitch receives the value 0.
mode: name of a previously defined SWITCH:
0 (or OFF) = the value of outswitch is not changed
1 or 2 (or INT) = outswitch receives the input value but with the part after the decimal point removed
?:2 (or NINT) = outswitch receives the input value rounded to the nearest integer
default: 1
Control inputs:
/I input signal (default 0)

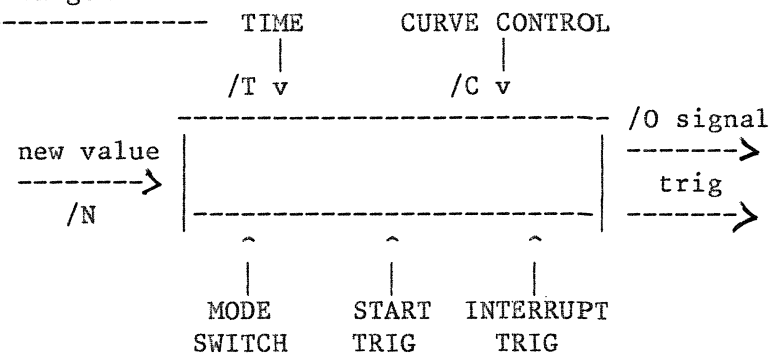
Example:

```
?:CREATE SWITCH MSW NINT
?:CREATE SIGSWI SS1 SWOUT MSW
?:CREATE CONNEC SCON SEQ5 SS1
```

We create a SIGSWI box called SS1, whose mode will be controlled by SWITCH MSW (which has the value NINT initially), and whose output is directed to switch MSW. A previously defined box called SEQ5 is connected to its input.

SLIDE

Single-segment function generator



When START TRIG is ON, a slide is started from the value currently stored at the box's output to the value at control input /N; the duration of the slide is controlled by control input /T (in seconds), and the type of slide (linear or exponential) is determined by MODE SWITCH; in exponential mode, the curve form is determined by control input /C.

?:CREATE SLIDE boxname [mode],[starttrig],[interrupt],[trigout]

boxname: a unique user-defined name, max 6 characters
mode: name of a previously defined SWITCH:
1 (or LIN) = linear interpolation (default)
.GT. 1 (or EXP) = exponential interpolation
starttrig: name of a previously defined TRIGGEr which will start a new slide when set to ON
interrupt: name of a previously defined TRIGGEr which will interrupt the current slide when set to ON: the value at the /NEW input is output immediately
trigout: name of a previously defined TRIGGEr which will be set to ON when the slide reaches its /NEW value

Control inputs:

/N	a signal which determines the ending-point of the slide default 0
/T	a signal which determines the duration in seconds default 0
/C	a signal in the range -10 to 10 which determines the curve form of the slide in exponential mode default 0

Example:

```
?:CREATE SLIDE SL1 MSW SLSTT SLBRK SLOUT
?:CREATE CONNEC CSL2 NOTE1 SL1/N
?:CREATE CONNEC CSL3 TIMER SL1/T
?:CREATE CONNEC CSL4 SHAPE SL1/C
```

We create a SLIDE box called SL1, to be started and interrupted by triggers SLSTT and SLBRK respectively, its mode to be determined by switch MSW, and its trigger output to be directed to trigger SLOUT. A previously defined box called NOTE1 will control the destination of the slide, TIMER will control its duration, and SHAPE will control the curve form.

Command string box

?:CREATE STRING boxname trigin trigout text

boxname: a unique user-defined name, max 6 characters

trigin: name of a previously defined TRIGGEr - when this is ON, the command string defined for this box is placed in the **'string queue'**, to be executed as soon as possible (when there is no input from file or terminal, and when all strings previously placed in the queue have been executed)

trigout: name of a previously defined TRIGGEr which will be set to ON when this box reaches the head of the queue

text: any legal WSP command; the user can make most efficient use of available memory by abbreviating commands as far as possible. For example, instead of CREATE CONNEC CX1, write CON CX1. All unnecessary spaces before and after the command text are automatically deleted

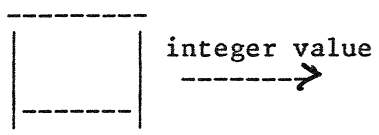
For example:

?:CREATE STRING ST1 TRIG10 TRIG11 SHOW TEXT ST1 ACTIVATED

ST1 contains the command SHOW TEXT ST1 ACTIVATED. Whenever TRIG10 is ON, this command is placed in the string queue, and when it reaches the head of the queue, the text ST1 ACTIVATED is displayed at the terminal, at the same time as TRIGGEr TRIG11 is set to ON.

SWITCH

Multi-directional switch



?:CREATE SWITCH boxname [value]

boxname: a unique user-defined name, max 6 characters
value: EITHER an integer
OR one of the following words, whose corresponding integer values are shown:

used in:		FUNC/SLIDE	MATH	PFUNC	FUNC	FUNC	RANDOM	SEQUEN
OFF	0							
ON	1	LIN	SIN	SIN	EXT	HOLD	RECT	FORE
	2		FIX	TRI	INT	RAMP	CUTG	
	3		SQR	SQR	BOTH		BINARY	BACK
	4	EXP	EXP					
	5		COS					
	6		LOG					
	7		LOG10					
	8		DBVOLT					
	9		VOLTDB					
	10		ABS					
	11		NINT					

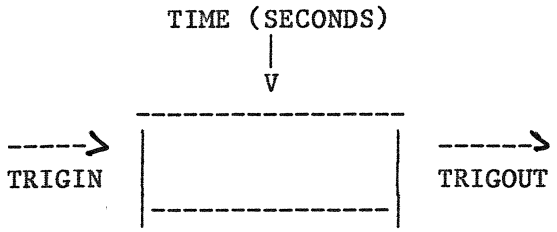
Default value: 0 (OFF)

Examples:

```
?:CREATE SWITCH SW5 SIN      creates a SWITCH called SW5 with the
                             value SIN, or +1
?:SWITCH INTEXT INT          creates a SWITCH called INTEXT with
                             the value INTernal, or +2
?:SWITCH BINARY 2            creates a SWITCH called BINARY with the
                             value 2
```

TDELAY

Trig delay box



?:CREATE TDELAY boxname trigin trigout n

boxname: a unique user-defined name, max 6 characters

trigin: user-defined name of an input TRIGGEr

trigout: user-defined name of an output TRIGGEr to be set a specified time after TRIGIN is set

n number of memory locations to be allocated to this box. This is the same as the number of TRIG signals that can be stored by the box at any one time. (default 1)

Control input: /TIME
defines the time in seconds between the setting of TRIGIN and the setting of TRIGOUT (default 0)

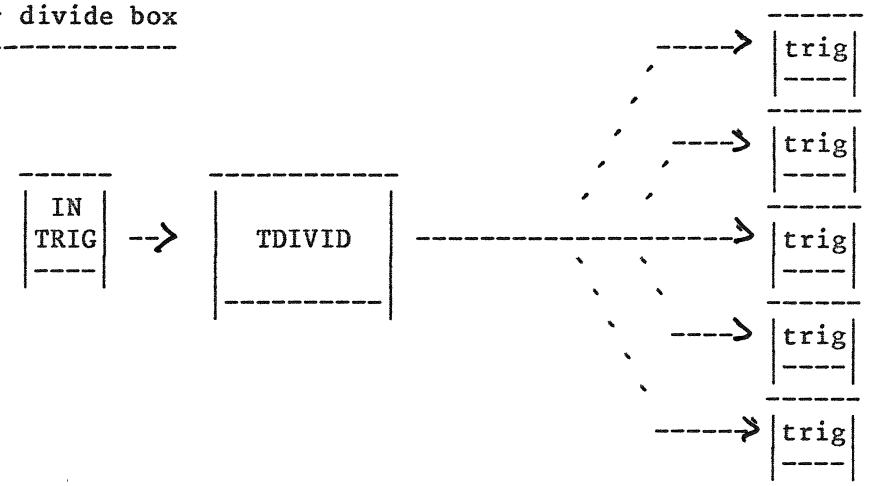
Example :

```
?:CREATE TDELAY DEL1 T3 T4 10
?:CREATE CONNEC CON5 TIMER DEL1/TIME
```

Here we create a TDELAY box called DEL1 which will set up a delay between TRIGGErs T3 and T4 (both previously created). We then create a connection box called CON5, which connects a box called TIMER to the /TIME input of DEL1: the output of TIMER will from now on determine the delay, in seconds.

TDIVIDe

Trigger divide box



?:CREATE TDIVID boxname intrig noutputs
:outtrig1 outtrig2 ... outtrig(noutputs)

boxname: a unique user-defined name, max 6 characters
intrig: name of a previously defined TRIGGEr
noutputs: number of trigger outputs (default 10)
outtrig: name(s) of previously defined TRIGGErs

Outputs:
:Tn is the nth output trigger

Whenever TRIGGEr **'intrig'** is ON, all the specified output triggers are set to ON.

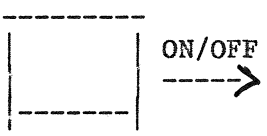
Example:

```
?:CREATE TDIVIDE TDIV1 TRIG10 5
:T1 T2 T3 T4 T5
```

A TDIVIDe box called TDIV1 is here created. TRIG10 is the input trigger, and there are five output triggers: T1, T2, T3, T4, and T5

TRIGGEr

Trig-pulse generator box



?:CREATE TRIGGE boxname [position]

boxname: a unique user-defined name, max 6 characters
position: one of the words ON or OFF
Default value: OFF

Example:

```
?:CREATE TRIGGE TR2
```

creates a TRIGGEr called TR2 in the default OFF position. See under the heading FUNCTION for an example of the use of a TRIGGEr.

A TRIGGEr is always turned off immediately by whatever box makes use of it. TRIGGErs should therefore not be used to control more than one box. TRIGGErs can be controlled from the terminal with the command:

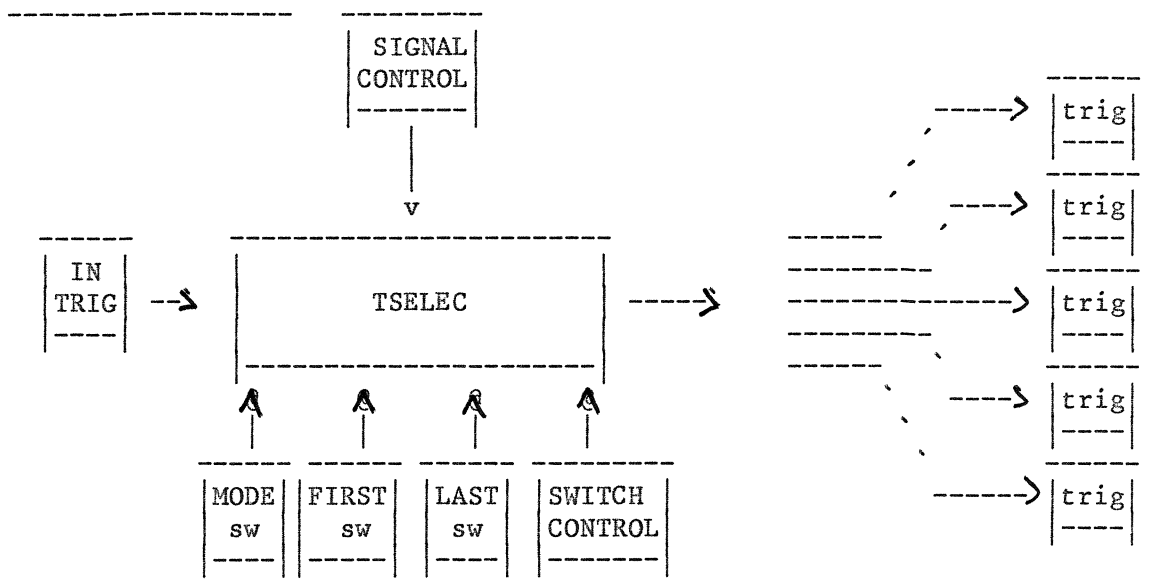
```
?:boxname position
```

For example:

```
?:TR2 ON
```

TSELECt

Trigger select box



?:CREATE TSELEC boxname noutputs trig mode first last select outswitch
:outtrig1 outtrig2 ... outtrig(noutputs)

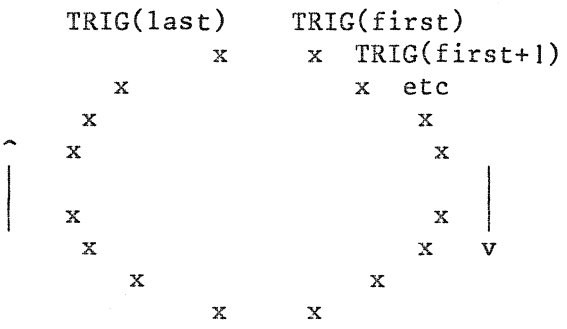
boxname: a unique user-defined name, max 6 characters
noutputs: total number of trigger outputs (default 10)
trig: name of a previously defined TRIGGEr
mode: name of a previously defined switch that will determine the mode of this box: (1) circular, (2) **'voltage'**-controlled, (3) switch-controlled (see below for details) default: circular
first: name of a previously defined SWITCH whose value, together with **'last'**, determines the range of trigger outputs which will actually be used. If, for example, SWITCH **'first'** has the value 3 and SWITCH **'last'** has the value 5, trig signals will be output only to the 3rd, 4th and 5th triggers in the list **'outtrigs'**
default: 1
last: name of a previously defined SWITCH whose value determines the end of the output trigger range
default: noutputs
select: name of a previously defined SWITCH that will control selection in mode 3 (default 0)
outswitch: name of a previously defined SWITCH that will receive a value describing which trigger has been set. E.g. when the 6th trigger is set to ON, outswitch is set to 6.
outtrig: name(s) of previously defined TRIGGErs

Control input: /CON for mode 2 control (default 0)
Triggers: :Tn refers to the nth trigger output

When a trig pulse is input (i.e. **'trig'** is ON), one of the output triggers **outtrig** is set to ON. The output trigger is chosen as follows:

mode 1: circular

Output triggers are set to ON in the same order in which they were written in the CREATE TSELEC command. The first trigger follows the final one.

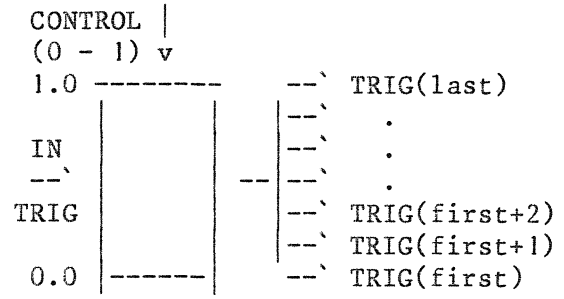


mode 2: voltage controlled

A signal in the range 0 to 1, connected to the /CON input, selects an output trigger according to the formula:

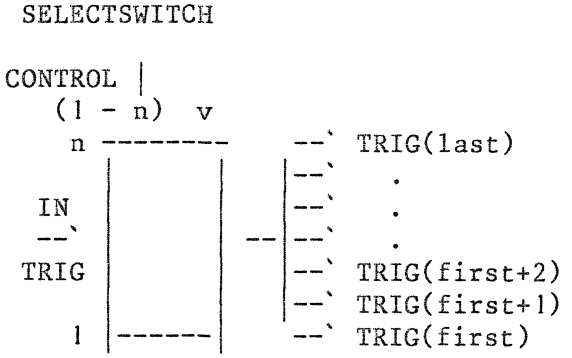
n = v * noutputs + 1

where v is value of the control signal, noutputs is the total number of trigger outputs, and the **'n'**th trigger in the CREATE list is set to ON.



mode 3: switch controlled

When **'selectswitch'** has the value n, the **'n'**th trigger in the CREATE list is set to ON.

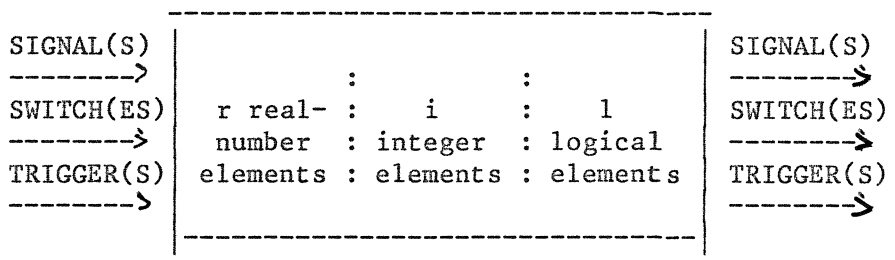


Example:

```
?:CREATE TSELEC TSEL5 4 TRIG20 MODE TFIRST TLAST SELEC
:ET1 ET22 ET13 ET45
?:CREATE CONNEC C55 BLEEP TSEL5/CON !connect BLEEP to control input
?:MODE 3                           !set mode switch to 3
?:SELEC 3                          !set select switch to point to
                                   !third output trigger (ET13)
?:TFIRST 1                         !set first and last switches to
?:TLAST 4                          !use ALL output triggers
```

A TSELEC box called TSEL5 is created: when TRIG20 is ON, a trig pulse will be sent to one of four previously defined triggers (ET1, ET22, ET13, and ET45); the mode of selection will be determined by the previously defined SWITCH called MODE; and a previously defined SWITCH called SELEC will control output when MODE has the value 1.

User-defined box



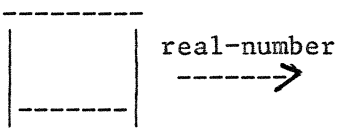
Performs one or more algorithms devised by the user in FORTRAN code. Detailed instructions on how to write USER boxes are contained in the file USER.FOR.

```
?:CREATE USER boxname [ndata],[ndati],[ndatl],[nsigin],[ntrigin],
     [nswitchin],[nsigout],[ntrigout],[nswitchout]
ndata FLOATING-POINT VALUES:v(1)  v(2)  ... v(ndata)
ndati INTEGER VALUES:n(1)  n(2)  ... n(ndati)
ntrigin INPUT TRIGS:t(1)  t(2)  ... t(ntrigin)
nswitchin INPUT SWITCHES:s(1)  s(2)  ... s(nswitchin)
ntrigout OUTPUT TRIGS:t(1)  t(2)  ... t(ntrigout)
nswitchout OUTPUT SWITCHES:s(1)  s(2)  ... s(nswitchout)

boxname:  a unique user-defined name, max 6 characters
ndata:      number of memory positions to be reserved for real numbers;
            default: 0
ndati:      number of memory positions to be reserved for integers;
            default: 0;
ndatl:      number of memory positions to be reserved for logical values;
            default: 0; the specified number of positions are put to .FALSE.
nsigin:     number of signal inputs; default: 0
ntrigin:    number of TRIGGEr inputs; default: 0
nswitchin: number of SWITCH inputs; default: 0
nsigout:    number of signal outputs; default: 0
ntrigout:   number of TRIGGEr outputs; default: 0
nswitchout:number of SWITCH outputs; default: 0
v:          values to be stored in the space reserved for real numbers;
            default: 0
n:          values to be stored in the space reserved for integers;
            default: 0
t:          names of the previously defined TRIGGErs to be used as input
            and output
s:          names of the previously defined SWITCHes to be used as input
            and output (input switches are by default OFF)
Control inputs:
        /Xn    the nth real-number constant - described as v(n) above
        /In    the nth input signal, where n must not be greater than 'nsigin'
        /On    the nth output signal, where n must not be greater than 'nsigout'
Switches:
        :Sn              the nth switch input
        :Sn+nswitchin    the nth switch output
Triggers:
        :Tn              the nth trigger input
        :Tn+ntrigin    the nth trigger output
```

VALUE

Real-number box



?:CREATE VALUE boxname [number]

boxname: a unique user-defined name, max 6 characters
number: the value to be assigned to the box (default 0)

The output of VALUE boxes is NOT write-protected; other signals may be connected to them, thus destroying their original contents.

Example:
?:CREATE VALUE V105 10.5

A VALUE box called V105 is created, which will output the value 10.5

EXAMPLES

The following demonstration files are available on directory [WSP.WSP]. They can be called with the WSP command:

?:CALL [WSP.WSP]name

CDISTR.WSP
FUNCTI.WSP
IF.WSP
LIMIT.WSP
MATH.WSP
MIX.WSP
PFUNCT.WSP
QUANTI.WSP
RECORD.WSP
SDELAY.WSP
SEQUEN.WSP
SIGSWI.WSP
SLIDE.WSP
STRING.WSP
TDIVID.WSP
TSELEC.WSP
USER.WSP