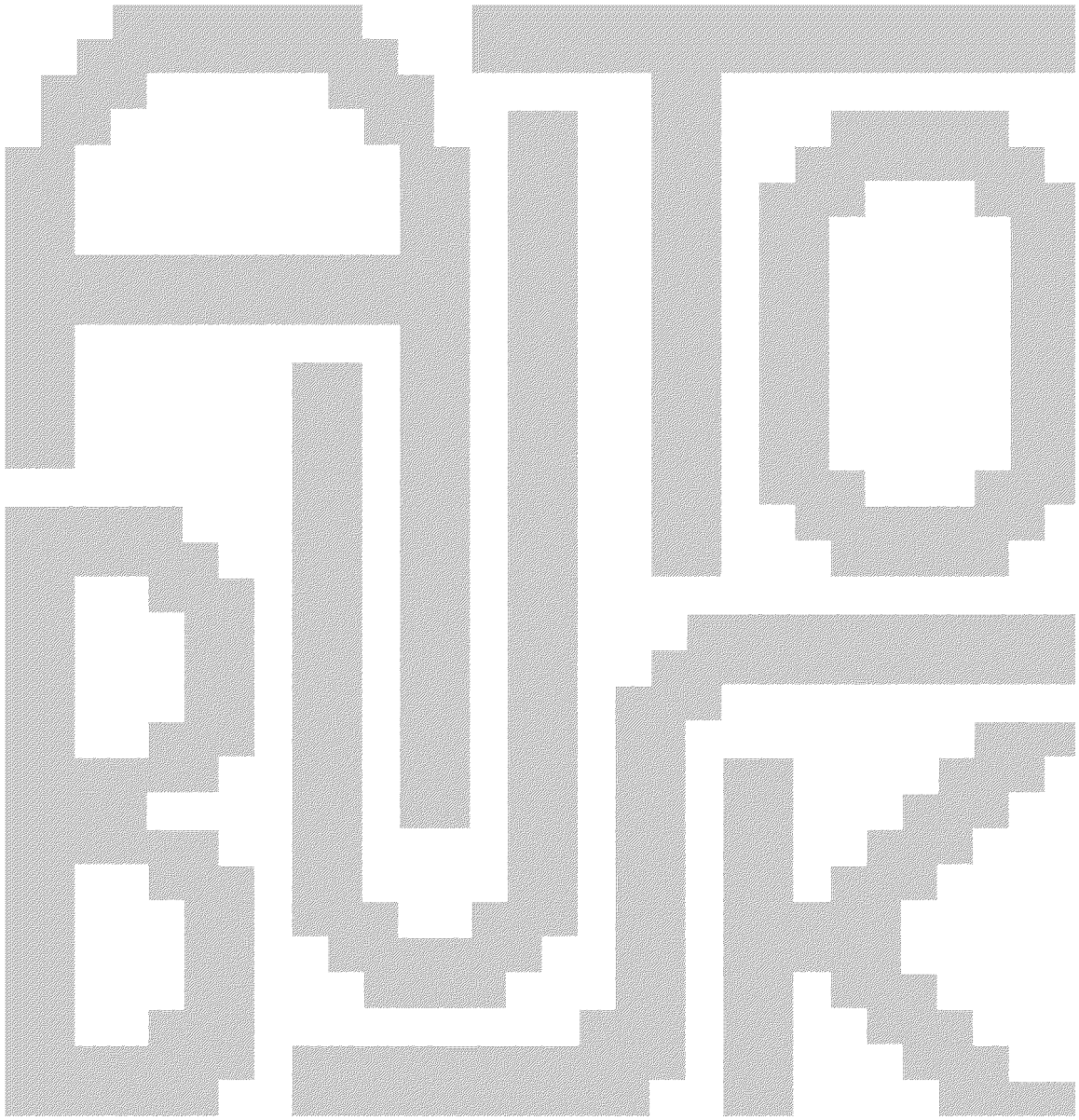


CLARENCE BARLOW

AUTOBUSK

A REAL-TIME PITCH & RHYTHM GENERATOR

October 2000

Contents

Chapter I - The Central Program
#1 Preliminary description 1
#2 Memory and Disk Requirements 1
#3 The AUTOBUSK package (to be read now or later) 1
#4 Starting AUTOBUSK 3
#5 The PRŒM (and Help) Boxes 5
#6 The Boxes at Bottom Left 9
#7 The Buttons at Top Left 10
#8 MIDI Input (somewhat complicated!) 13
#9 The Remap Box 16
#10 Final Remarks 17

Chapter II - The Pre-Processors
#1 **ASC** - a text editor 18
#2 **HRM** (Harmonic Rationalisation Measures) - a Scale Compiler 19
#3 **IDP** (Indispensibility Determination Program) - a Metre Compiler 25
#4 **JST** (Justified Systemic Tuning) - a Lister of Rational Intervals 26

Chapter III - The **.PRM**-Processors
#1 **EDIT** - a Text Editor 30
#2 **FILL** - the Flush Interpolator 30
#3 **FUSE** - the Synchronizer 32
#4 **JOIN** - the Concatenator 33
#5 **PACK** - the Compressor 34
#6 **PART** - the Separator 35
#7 **PICK** - the Selector 37
#8 **TIDY** - the Cleaner-Upper 38
#9 **VARY** - the Modifier 39

Chapter IV - File Formats
#1 **.CTS** - Raw Pitch Scales 42
#2 **.HRM** - Compiled Pitch Scales 42
#3 **.JST** - Interval Vocabulary 42
#4 **.MTR** - Raw Pulse Metres 43
#5 **.IDP** - Compiled Pulse Metres 43
#6 **.PRM** - Basic PRŒM Score 43
#7 **.PRK** - Compressed PRŒM Score 44
#8 **.α** - the 'Alformation' File 44
#9 **.Φ** - the Flash File 45
#10 **.N** - the AUTOBUSK Internal Font File 46
#11 **.MDK** - the MIDIDESK-format MIDI File 46

Glossary 47

Index 52

Bibliography 54

Chapter I - The Central Program

#1 Preliminary description

AUTOBUSK is a program for the real-time probabilistic generation of MIDI signals, primarily note-ons and -offs (remappable to other command types), in upto 3 parallel note streams (more than one computer used in parallel offers multiples of 3 streams). Pitch sets (i.e. scales e.g. major or minor) and pulse sets (i.e. metres e.g. 6/8 or 4/4) serve as the main input *Material*. A set of 12 variable *Parameters*, externally storable in time-tagged score form, can form MIDI and/or filed output; re-inputting the latter also sends MIDI events along 2 given *Routes*. 14 attendant programs (with, like AUTOBUSK, the extension **.PRG**) serve to enter and compile the Material and to process 'meta-scores'. AUTOBUSK itself took 272 days to write, spread between 18 August 1986 and 30 October 2000.

#2 Memory and Disk Requirements

AUTOBUSK requires an internal memory of 576 KB and a minimum disk space of 1.2 MB, which will increase during the use of the program. The total software package comprises the program **AUTOBUSK** and the 2 folders **PREPROC** (holding the 4 programs **ASC**, **HRM**, **IDP** and **JST** and the file **HRM.JST**) and **PRMPROC** (holding the 9 programs **EDIT**, **FILL**, **FUSE**, **JOIN**, **PACK**, **PART**, **PICK**, **TIDY** and **VARY**).

In principle, AUTOBUSK can run alone, without any of its attendant programs; on starting it, three more (empty) folders - **FLUSHES**, **PRKSCORE** and **PRMSCORE** - and three small data files - **AUTOBUSK.ℵ**, **AUTOBUSK.∝** and **AUTOBUSK.Φ** (the last two on demand) - will be created and kept for further use. The absence of the **PREPROC** and **PRMPROC** folders precludes the use of the programs therein.

#3 The AUTOBUSK package (to be read now or later)

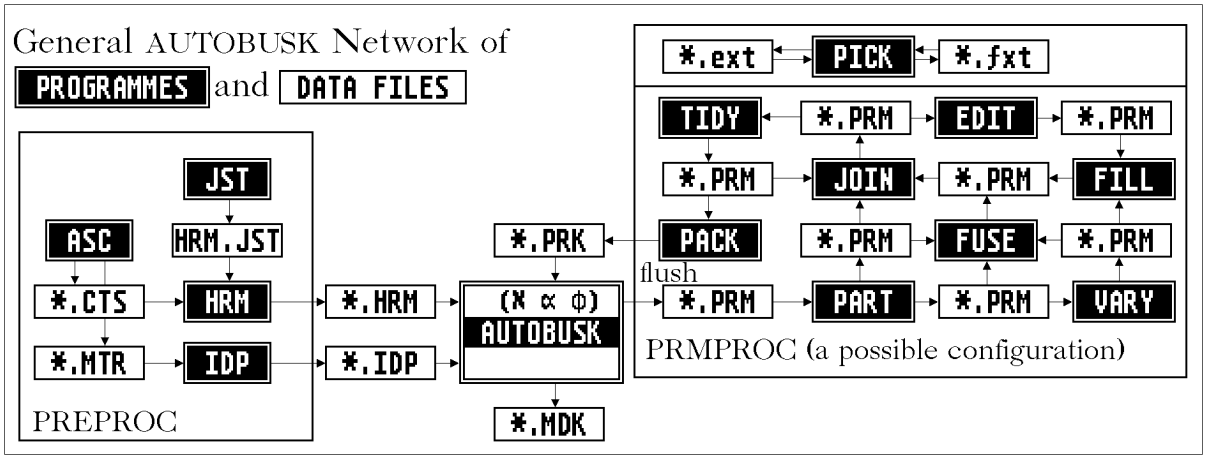
In diagram 1 one sees in the middle the program **AUTOBUSK**; above it are the 3 symbols **ℵ**, **∝** and **Φ**, representing the 3 data files with the corresponding extensions as mentioned at the end of #2 -

ℵ (a binary file created by AUTOBUSK for future reference and/or alteration) contains a special font used for displaying note-names and other entities,

∝ (a text file created by AUTOBUSK for future reference and/or alteration) contains special user-defined higher-level MIDI and other settings and is online-storable and -recallable on request,

Φ (a binary file) contains upto 99 independent stored user-defined configurations (called *flashes*) of Parameters, MIDI Routes and pitch/pulse Material, (collectively called *PR€Ms* - pronounced as in *premier* or *premiss*).

Diagram 1 - The AUTOBUSK Network



To the left of the diagram is a box entitled PREPROC (for 'pre-processing'): it shows 4 programs, all contained in the **PREPROC** folder, viz.

ASC, an imported ASCII-text-editor (e.g. **TEMPUS**), for the entering and editing of scales and metres in AUTOBUSK's own **.CTS** and **.MTR** formats, respectively,

HRM for the compilation of **.CTS** files into **.HRM** (binary) files,

IDP for the compilation of **.MTR** files into **.IDP** (text) files and

JST for the possible creation of a suitable pitch-interval vocabulary **HRM.JST** (a binary file, here already supplied) for use in the program **HRM**.

.CTS and **.MTR** files (and, correspondingly, **.HRM** and **.IDP** files) can contain several scales and metres, respectively, of which, however, only the first 6 are accepted by AUTOBUSK for use at one time - see Chapter IV for more.

AUTOBUSK also generates (see again the middle of the diagram)

.MDK files, MIDI event scores in a text-file format developed for use in the author's modular MIDI-file-processing package MIDIDESK (each line of text contains 5 or 6 elements described in Chapter IV) and

.PRM files in a text-file format, explicit individual PRCM *flushes* (related in content but different in format from the **AUTOBUSK.ϕ** collective binary flash information): each line of text contains 5 elements fully described in Chapters III#6 and IV#6). A **.PRM** file can comprise a *flush*, i.e. a list of identically time-tagged PRCM value settings; it can also form a *score* containing gradually increasing time-tags in seconds and milliseconds.

.PRK (compressed **.PRM**) files are read by AUTOBUSK - see the next paragraph.

.PRM flushes can be passed on to the PRMPROC ('PRƟM-processing') package on the right of the diagram: the nine programs here (all in the **PRMPROC** folder) are

EDIT, like **ASC** an imported ASCII-text-editor (e.g. **TEMPUS**), for the editing and/or manual entering of **.PRM** files,

FILL, for the interpolation of PRƟM values from one flush file to another at a fixable time-delay, thereby creating a time-tagged **.PRM** score,

FUSE, for making a number of **.PRM** scores simultaneous, i.e. for merging them into a single **.PRM** score, the length of which (i.e. the final time-tag) equals that of the longest input file,

JOIN, for concatenating a number of **.PRM** scores into a composite **.PRM** score, the length of which equals that of the sum of all the input score lengths,

PACK, for the compression of **.PRM** into binary **.PRK** files for on-line real-time reading by AUTOBUSK,

PART, for splitting a **.PRM** score into 2 separate ones according to exact constraints (these 2 can be merged again by **FUSE** to regain the original),

PICK, actually not solely a PRƟM processor, for selecting upto a maximum of 9 **.HRM**, **.IDP**, **.PRM** or **.PRK** files for use by AUTOBUSK's file selectors - or upto 20 **.CTS** or **.MTR** files for use by the **HRM** or **IDP** programs,

TIDY, to remove redundant information (generated e.g. by **JOIN**) in a **.PRM** score.

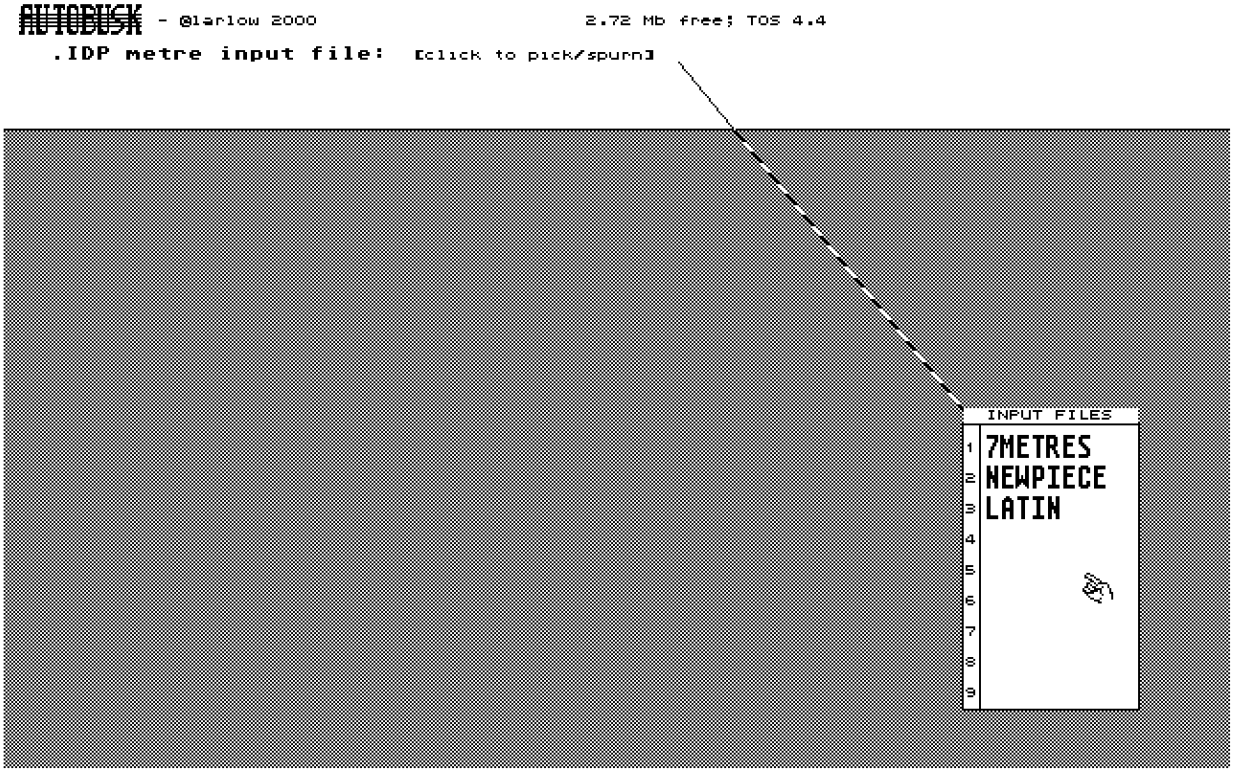
VARY, for the alteration of PRƟM attributes (including time-tags) in a **.PRM** score by *spreading*, *shifting* or *sliding*, as will be described later.

#4 Starting AUTOBUSK

Double-click the **AUTOBUSK.PRG** icon. A grey screen shoots up, leaving a white strip at the top. At once, the 8 letters **# # # # # # # #** begin to appear at top left, complete when the program has gone through its initialisations. If the **.N** file is absent, it is now created, its characters displayed on the screen at the letter **#**.

Then follows the first file request, asking for an **.IDP** compiled metre file (see diagram 2); the *Selector Box* it points to can offer upto 9 files, prechosen by **PICK** or being simply the only ones in the folder. A file can be chosen either by clicking its name with the left mouse button (hereafter referred to as "Ö_", the right button as "_Ö": on the screen the corresponding symbols are **ö_** and **_ö**, respectively) or by typing the number to the left of the file's name. Alternately, a Return chooses the first listed file. The chosen file is then read; upto 6 encountered metres are shown in a box at top right (see diagram 3) - metres 'known' to AUTOBUSK by name (e.g. **2/4**) or, if unknown, by *stratification* (e.g. **<2x2x2>**): in all cases, the lowest-level pulse is a ♪ (16th-note or semiquaver). If no files are present, or if all are 'spurned' by Öing outside the box, an internal **.IDP**-type default is used: a 16-pulse binarily stratified metre (2×2×2×2, i.e. a metre divided 4 times by 2).

Diagram 2 - the first File Selector Box



The second file request asks for a **.HRM** compiled scale file; as before, upto 9 files are shown, to be picked by mouse or explicit number or Return . . . or spurned as above. If picked, upto 6 scales from the file are seen at top right, 'known' ones by name (e.g. **major** - microtonal scales rounded to semitone-based ones, e.g. 7-tone equal-temperament given as **dorian**), 'unknown' by semitone steps (e.g. **<22122221>** for the major, were it unknown). A scale exceeding but not repeating at the octave, is shown limited to the same, followed by a **+**; intervals exceeding a major 6th are shown by **A** for 10, **B** for 11 semitones etc. For spurned or absent **.HRM** files a default internal **.HRM**-type setup is chosen: the chromatic scale.

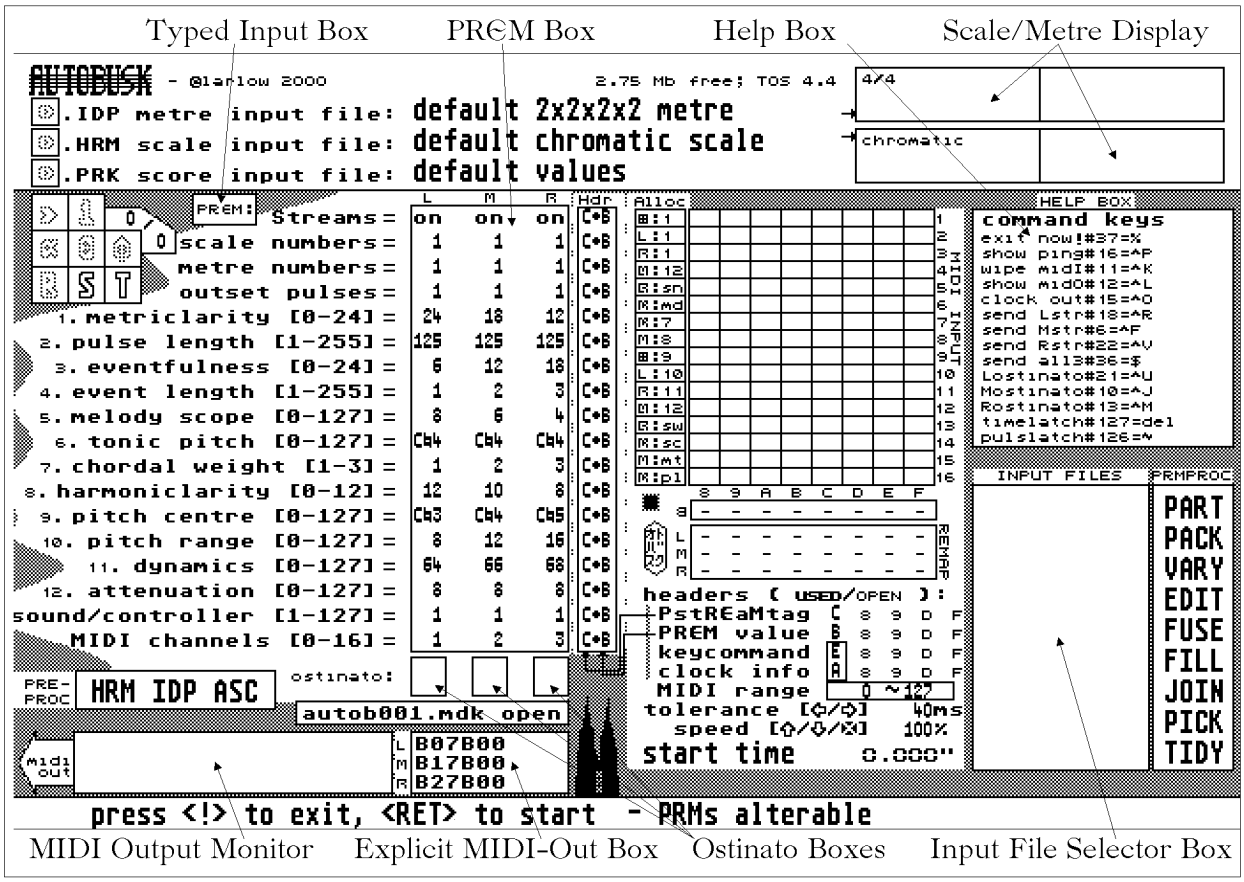
The third and final file request concerns **.PRK** (compressed **.PRM**) files: if one of the at most 9 is picked, it is at once is ready for execution; if spurned or absent, a default is chosen instead, being either the final flash in **AUTOBUSK.Φ**, or, if this file is also absent, a default PR©M setting, shown in the next section.

For each of the above selections, pressing Shift + Return (instead of only Return or Ö ing) causes the first file in the pertinent and subsequent Selector Boxes (or, if files are absent, the defaults/final flash) to be chosen without stopping.

#5 The PREM (and Help) Boxes

After all 3 file requests have been satisfied, AUTOBUSK fills up the screen with all its remaining buttons and boxes, the most prominent of which is the *PREM Box* in the middle with its several legends (e.g. **metriclarity**) on the left - diagram 3 displays a typical complete desktop, certain boxes being especially indicated.

Diagram 3 - The Main Autobusk Desktop (note the default PREM values)



But first look at the **HELP BOX** near the upper right-hand corner of the diagram. AUTOBUSK generally uses computer keys according to their position on the keyboard and not to the characters they represent. Since keyboards differ, depending whether they are configured for use in English, German, French etc., AUTOBUSK first scans the keyboard and lists some keys and their functions in the Help Box and in the Hexagon near the middle of the screen (described later): the Help Box seen here matches an English and a German keyboard. The ^ sign indicates the use of the Control key. The following will illustrate from time to time individual Help Box key functions.

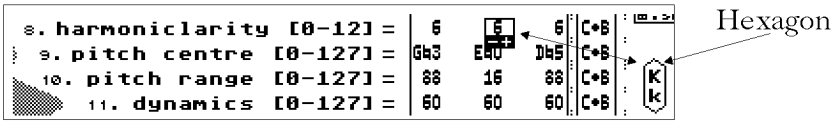
Back to the PRCM Box: the 3 vertical columns here correspond to the 3 independent MIDI streams the program generates; they are labelled L, M and R at the top, meaning 'Left', 'Middle' and 'Right', for lack of anything better. These 3 columns are crossed by 18 horizontal rows, each corresponding to a PRCM; note among them the 12 Parameters numbered 1 to 12. The *instruction line* at bottom says at this point **Press <!> to exit, <Return> to start; PRMs alterable**. Moving the mouse into this box, one notices it change its shape, which makes this a good place to mention the 6 different mouse types in AUTOBUSK:

1. the *inactive mouse* [▪], already seen, which does nothing but be visible,
2. the *file-pick mouse* [✎], a pointing hand, already seen, for selecting files,
3. the *plus-minus mouse*, [⊟], mainly used here in the PRCM Box, inviting the user to Ö to decrement and Ö to increment the value used (the general term 'crement' means either).
4. the *one-click mouse* ([▫]), used on a number of buttons such as the cluster at top left (more about this in #6), invites the user to Ö to activate the button - it thereupon at once becomes inactive until it has moved outside the button,
5. the *latch-on mouse* ([■]) is used in boxes where input is effected by typing a character: the mouse 'latches' onto the nearest spot where one can do this,
6. the *busy mouse* ([⌛]) appears ephemerally when AUTOBUSK is working something out and disappears thereafter.

There are in total 5 ways of altering the values in the PRCM Box (keep watching diagram 3):

1. As just seen, this may be done by mouse clicking.
2. Key-typing; if the mouse is poised within the PRCM Box, the Hexagon to the right of the box indicates the characters needed for the PRCM/stream where the mouse is currently poised (otherwise one sees the Japanese Katakana characters for *otobasuku*, the program's name). See also diagram 4: the keys for the 12 Parameters are shown as a character pair sharing the same key (with or without Shift, which if depressed causes the incrementatation, otherwise decrementation, of the values); all other PRCMs (except **streams**: - see here below) use control keys indicated by a preceding ^, where, too, the additional Shift key increments the values.

Diagram 4 - The Hexagon showing a typical key pair for the mouse-positioned PRCM



3. Explicit typed-in values: having poised the mouse at the desired PR∈M value, the keypad to the keyboard's right is used to type in the numerical value, the input being terminated by a keypad . ("full stop" or "period"). The input appears in the small box to the left of the word **streams** and disappears after completion; the display in the PR∈M Box is then updated.
4. Properly configured MIDI input (see #8).
5. Input from a **.PRK** file (see Chapter III).

Of the 18 horizontal rows in the PR∈M Box, the top 4 are Materials, dedicated as follows:

streams: this enables the 3 streams to be individually switched on or off; the corresponding display is **on** or --. The key combination usable here as a toggle
for **L** is Alternate + Tab (indicated in the Hexagon as ◘α),
for **M** Alternate + Control (¢α),
for **R** Alternate + Shift (♂α).

scale numbers: here one of the (maximum 6) scales can be entered by its number as shown in order in the Scale Display Box at top right. If the number exceeds that of the scales actually present, the value is accordingly set at the latter.

metre numbers: one of the (maximum 6) metres can be entered by its number, following the Metre Display Box at top right. Here, too, an excessive amount is curtailed.

outset pulses: Each metre can be phase-shifted so that the first pulse to be attacked on starting a run can be set to this value; e.g. if the metre has 8 pulses and this value is set to **5**, this is the pulse with which the music will start. The 3 streams can have different outset pulses.

The 12 Parameters follow, each attended in the square brackets at right by its minimum and maximum value:

1. **metriclarity**: a measure of the resultant clarity of the metre as played in the stream - at minimum all pulses are accorded equal probability and the metre cannot be musically detected, whereas at maximum the opposite is true.
2. **pulse length**: the length in milliseconds of the pulses in the stream. If different values are used here in the 3 streams and it is desired at a certain point that all attacks are set at once to the same pulse number or synchronized in time (e.g. if the pulse lengths are equated and the MIDI output should then be in phase), one can achieve this by pressing the keys shown for **timelatch** and **pulslatch** in the Help Box.
3. **eventfulness**: at minimum the stream is inactive, at maximum it is saturated with attacks, e.g. every available pulse in the metre accorded to this stream is played.

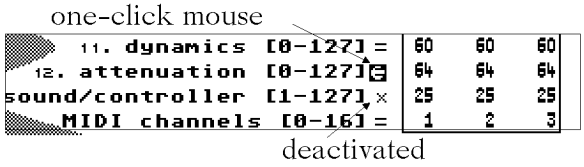
4. **event length**: the length in pulses of every event generated; a value of **1** is similar to 'staccato', a large value to 'legato'.
5. **melody scope**: the maximum number of semitones permitted between the MIDI number of any event and the one following it.
6. **tonic pitch**: the MIDI number of the current keynote; note that keynotes are non-octave-repeating - the octave number is included in the display (e.g. Middle C or MIDI number 60 is indicated as **C4**, one octave above standard MIDI notation, a common practice prior to and outside of standard MIDI).
7. **chordal weight**: upto a maximum of 3 notes can be played together in each stream.
8. **harmoniclarity**: analogous to Parameter 1 above, a minimum value accords all pitches in the scale an equal probability, thus weakening the key-feeling; at maximum the opposite is true. Note that this description particularly holds in the case of scales such as the chromatic; a major scale would introduce its own intrinsic key-feeling even at the minimum Parameter value.
9. **pitch centre**: the centre of the pitch range (Parameter 10 - see below).
10. **pitch range**: the width in semitones of one-half of the MIDI-note 'window' within which MIDI-notes are generated.
11. **dynamics**: the central (=average) MIDI-'velocity' of the stream.
12. **attenuation**: the extent to which the dynamics may deviate from the value of Parameter 11; at minimum 'metriclarity' (Parameter 1) this is decided at random, whereas at maximum 'metriclarity' the deviation is related to the 'indispensability' (see Chapter II#3) of the pulse - the metriclarity is then dynamically amplified; this may be useful in the case of maximum eventfulness (Parameter 3) where agogic elements are neutralised.

The 2 remaining PRGMs are Routes:
sound/controller: the program change number to access the desired sound in the attached audio device; in the case of remapping note-ons to controller commands (see #9), this number is that of the controller.
MIDI channels: each stream can be directed to a specific audio device (0-16); here a 0 sends the stream to the internal sound-chip of the computer, enabling a synthesizer/sampler to be dispensed with - this is only useful if the chordal weight (Parameter 7) is set to its minimum value (**1**). The values 1-16 indicate the MIDI channels of the audio device.

Note that only Parameter values are quantities; non-Parameter values - Material and Routes - are simply labels.

The = to the right of each PR©M legend can be converted by the one-click mouse into a × - the PR©M is now immune to change and can be made available to re-alteration only through the renewed application of the one-click mouse - see diagram *5*.

Diagram *5* - PR©M Deactivation



Press Return to start an execution run. The time indicator at bottom right starts counting the elapsed seconds (e.g. **time now 4"**); the instruction line now reads **Press <Esc> to stop, <Blank> to interrupt**.

Press Blank to interrupt the run. The time indicator shows the time of interruption e.g. as **stopped at 6.025"**; the instruction line reads **INTERRUPTED; press <Esc> to stop, <Return> to resume**.

Press Return to resume. The time indicator resumes the counting of seconds; the instruction line again reads **Press <Esc> to stop, <Blank> to interrupt**.

#6 The Boxes at Bottom Left

Make sure the computer is connected via a MIDI-cable to a synthesizer, sampler or other sound source and that - if necessary - the latter is connected via audio cables to an amplification system. If AUTOBUSK is still running (watch the time indicator), if Parameter 3 (eventfulness) is not at minimum, if the MIDI channels shown at the bottom of the PR©M Box are meaningful and available on the sound source and if the amplification is on, a result should be heard.

All PR©Ms can be altered in the manner described above (mouse, special keys shown in the hexagon, explicit key-pad numerical input, MIDI input and/or a **.PRK** file); try the first 3 - a change should be audible. Press the key given in the Help Box for **show ping**: the **on** display in the **streams** line at the top of the PR©M Box starts blinking on every first pulse of the corresponding stream's metric cycle. Re-pressing the key switches this function off.

Press the key given in the Help Box for **show midO**: the MIDI Output Monitor at bottom left will show the MIDI values being sent out in hexadecimal code, stream by stream. Re-pressing the key makes the MIDI Output Monitor inactive; pressing the key yet again wipes its contents and reactivates the Monitor.

Ostinato Boxes: the 3 little boxes just below the PR©M Box marked `ostinato:` - use the one-click mouse on one of them - at once the symbol :fTf: appears in the box and the rhythm of the corresponding stream is frozen, i.e. it is now repeated, though the pitches change. This *rhythm ostinato* function is switched off by re-using the one-click mouse. Now do it again while pressing Shift: the symbol :fTf: re-appears, joined by the symbol ≡ above - this is the *pitch+rhythm ostinato* function, which freezes both pitch and rhythm. Repeating the same action switches it off. The one-click mouse can be replaced by typing the keys given in the Help Box for `Lostinato`, `Mostinato` and `Rostinato` (for the `L`-, `M`- and `R`-streams, respectively). Adding Shift has the same result as with the mouse. One can go from rhythm ostinato to pitch+rhythm ostinato (Shift+one-click mouse) and back (one-click mouse); however, the rhythm ostinato will change each time.

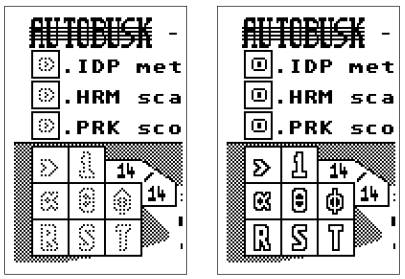
The `PREPROC` Box just above the MIDI Output Monitor names the three programs **ASC**, **HRM** and **IDP** in the **PREPROC** folder; these can be called when AUTOBUSK is not yet doing or has interrupted an execution run (see Chapter II).

The remark **autob001.mdk open** below the Ostinato Boxes will be explained in the next section. Below this is the Explicit MIDI-Out Box; here the latch-on mouse is active; use it to type in hexadecimal code (maximum 10 characters). The Help Box keys given for `send Lstr`, `send Mstr` and `send Rstr` send out this code on the MIDI channels allocated in the PR©M Box to the `L`-, `M`- and `R`-streams, respectively, as can be seen in the MIDI Output Monitor, if it is switched on. Press the Help Box key given for `send a113` to send out all three.

#7 The Buttons at Top Left

In diagram 6 one sees 3 vertically aligned buttons for the choice of **.HRM**, **.IDP** and **.PRK** files, respectively. Using the one-click mouse on these draws up the file menu in the corresponding Selector Box, which one can use to pick or spurn files as at the start of the program.

Diagram 6 - The 11 Top Left Buttons



(unset, set)

Below these buttons there is a cluster of 8 more, which will be explained from left to right and bottom to top. In the diagram these buttons have been shown twice, once in their 'unset', once in their 'set' position. All buttons may be set and unset by the one-click mouse.

The [R] button, when set, allows the recording of the MIDI output directly to disk, to the file **autob001.MDK**, reported open as mentioned above. Assuming AUTOBUSK to be still running, press [Esc] to stop the execution. The instruction line now reads **press <!> to exit, <Return> to go on**; the record file report shows **autob002.MDK** as being open (its predecessor was closed on stopping the execution). As long as this button is set, MIDI output will be recorded, the record file number being incremented on each new execution run. The text format **.MDK** is decribed in Chapter IV; it can be converted to standard MIDI by other software written by the author.

The [S] button, normally set, allows MIDI to be passed outwards; unsetting it may be preferred e.g. if one is recording an execution to disk but does not need to hear the results as yet.

The [T] button, normally set, generates MIDI 'in time', i.e. according to the clock, which is running normally. Unsetting it eliminates all waiting: events are composed as fast as possible - this may also be useful when recording while unsetting [S] as mentioned above.

The [α] button, when set, reminds AUTOBUSK on exiting (after pressing [!] - the final instruction line will include the information **filed new alfa;**) to store all special user-defined higher-level MIDI and other settings (e.g. the contents of the Explicit MIDI-Out Box) in the file **AUTOBUSK.α**, mentioned above. One can also store the settings at any time by Öing the button while simultaneously pressing [Shift]. Similarly, the disk-stored settings can be called by Öing while pressing [Alternate]. More will be said about the **.α** file at the end of this chapter.

The [0] button activates the 'thru' mode: all MIDI input is passed on outwards as well, but only when AUTOBUSK isn't executing PRGMs (i.e. it is stopped or interrupted) - if the MIDI channel is one of those allocated to the 3 streams, the signals will be displayed if desired in the MIDI Output Monitor according to the stream in question; otherwise the **R** stream is used.

The [Φ] button stores the current PRGM configuration to internal memory; upto a maximum of 99 flashes may be stored in this way. Irrespective of whether AUTOBUSK is executing or not, Öing here: the flash numbers displayed at the upper right corner of the button cluster are at once both incremented - if they had been standing at e.g. 7/7 (meaning "this is the 7th of 7 flashes"), they will have changed to 8/8. These 8 flashes will be stored in the binary file **AUTOBUSK.Φ** on exiting the program; the final instruction line will indicate the number of flashes stored. This way all previous configurations are loaded the next time AUTOBUSK is started. More about the flash indicators at the end of this section.

The [»] button sets the start time in the current **.PRK** input score file. For this, type a new start time if desired at the blinking cursor at bottom right (a [Return] sets it to zero). More will be said about the **.PRK** file type in Chapter III.

The [1] button, when set, will place a marker in the **AUTOBUSK.∝** file (provided the [∝] button is also set), telling AUTOBUSK to choose all first-listed input files (**.HRM**, **.IDP** and **.PRK**) the next time it is started - this has the same effect as pressing [Shift]+[Return] on the first appearance of the **.HRM** File Selector Box; AUTOBUSK, on being started, goes straight to the 'ready for execution' point - this could be useful for stand-alone applications or for when the same input files are always being used and are alone or first in their lists (or are absent and one is using the defaults/final flash).

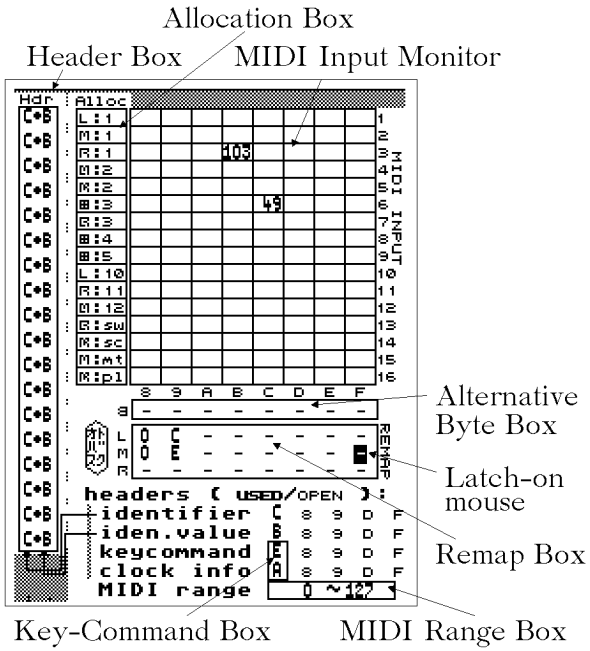
Now back to the flash indicators. As mentioned and shown in diagram 6, this is a pair of numbers indicating at upper left the number of the current flash as originally loaded, at lower right the length of the flash list. Also as mentioned, any current PRCM configuration can be 'flashed' to internal memory by the [0] button. A flash can be recalled in two ways (in all cases by first poising the mouse at the current flash number - the mouse converts to its latch-on form, latching onto the number): one can type in the number or alter it by the [+] and [−] keys. In all cases, this number will not exceed the flash list length. Finally Ö to recall the flash. Öing (with the right mouse button) while latched onto this number causes the current flash to be flushed as a **.PRM** flush file to disk - it will be stored in the **FLUSHES** folder as **#***nn*.**PRM**, where *nn* is the flash (here also flush) number.

As known, Öing the [0] button raises the flash list length; this can be lowered (with the mouse latched onto it) by the [−] key and raised back to its original .0-given length (not beyond) by the [+] key. Furthermore, latching onto the current flash number and pressing [Delete] causes this flash to be erased and all following flashes (and consequently the flash list length) to be decremented by 1 (to recall the new - formerly 'next' - current flash, Ö on its number - careful: there is no 'undo'). Pressing [Insert] in the same way causes the current PRCM configuration to be 'flashed' to internal memory, just as by the [0] button, except that this can now happen at the current position in the flash list, incrementing all subsequent flash numbers by 1 (unless the list length is already 99). In any case, only the flashes listed upto the shown list length will be stored in the .0-file on exiting AUTOBUSK.

#8 MIDI Input (somewhat complicated!)

In diagram 7 one sees all the boxes and other information relating to MIDI input.

Diagram 7 - Midi Input Boxes (and the Remap Box)



Rightmost is the MIDI Input Monitor: any MIDI signals entering AUTOBUSK are shown here, provided the status nibble or half-byte (displayed as 8-F from left to right) is indicated at lower right as being **USED** in bold type. One of the incoming data bytes, termed the 'relevant' byte, is displayed in the Input Monitor in the space against the status nibble on the horizontal axis and the MIDI channel number on the vertical axis. The relevant data byte for the 8 status nibbles is here regarded to be:

nibbles 8 and 9 (notes off and on) - 1st byte (MIDI pitch)
nibble A (after-touch) - 1st byte (MIDI pitch);
nibble B (controller) - 2nd byte (controller value)
nibble C (program change) - only byte (sound number)
nibble D (channel pressure) - 1st byte (pressure value)
nibble E (pitch bend) - 1st byte
nibble F (system command) - 1st byte

The MIDI Input Monitor (in which only **USED** input commands are displayed!) can be wiped clean by typing the key given in the Help Box for **wipe midI**.

If the incoming MIDI does not belong to the **USED** but to the **OPEN** type (non-bold), one gets to include it in the former by moving the mouse into the box marked **Hdr** (the Header Box), latching onto a MIDI nibble in the right column and by typing in the new desired status nibble. The list of **USED** and **OPEN** nibbles will change, the new nibble appearing bold on the left. Conversely, replacing a uniquely used nibble by another one already used elsewhere will cause the first to be moved from the **USED** into the **OPEN** list. The left column in

the Header Box contains and can receive status nibbles needed to allow AUTOBUSK to make changes in the box marked Alloc by MIDI input instead of by mouse - this box, the Allocation Box gives the incoming MIDI signals a meaningful function in altering the PROM values.

The left column indicates the stream combination:
type L for L (Left), M for M (Middle), N for N (Left&Middle),
O for O (Left&Right), P for P (Middle&Right), Q for Q (all three)
and R for R (Right).

The right column specifies the PROM to be altered by MIDI input signals according to their MIDI channel:
type W for streams (on/off-switch) or flashes (see below), shown on screen as sw,
S for scale, M for metre, P for outset pulse (on screen: sc mt pl),
1-9 for Parameters 1-9, 0, ´ and ^ for Parameters 10-12 (on screen: 01-12),
I for the sound/controller and H for the MIDI channels (on screen: sn md).

The setting engendered by W - streams or flashes - uses MIDI input databyte 0 to switch off and 127 to switch on the stream; between these two values, the input data byte values 1-99 are used on a settable MIDI channel (given in the .a file; if this is absent, channel 16 is chosen by default) to recall the flashes of the same number.

As mentioned above, the Header Box indicates in its right column the MIDI input command type (expressed by its status nibble) required to effect a PROM change; the left column indicates the MIDI command type for external allocation via MIDI input (e.g. in the case of stand-alone applications) - the first data byte following the status byte determines the stream combination or the PROM to be changed:
bytes 0-11 (0-B in hexadecimal code) set Parameters 1 to 12,
19-25 (13-19 h) set the stream combinations L, M, N, O, P, Q and R,
27 (1B h) sets **sound/controller**,
28 (1C h) **streams** (on/off-switch),
29 (1D h) **scale**, 30 (1E h) **metre** and 31 (1F h) **outset pulse**.
The MIDI channel used for the allocation is that currently set in the input device, e.g. a MIDI keyboard. The stream combination should be set first, then the PROM to be changed.

For instance, the situation in diagram 7 is as follows:
MIDI channel 1 is set to change PREM 1 in L
MIDI channel 2 is set to change PREM 1 in M
MIDI channel 3 is set to change PREM 1 in R
MIDI channel 4 is set to change PREM 2 in combination LM : left & middle
MIDI channel 5 is set to change PREM 2 in combination MR : middle & right
MIDI channel 6 is set to change PREM 3 in all 3 streams (⊞)
MIDI channel 7 is set to change PREM 3 in combination LR : left & right
MIDI channel 8 is set to change PREM 4 in all 3 streams (⊞)
MIDI channel 9 is set to change PREM 5 in all 3 streams (⊞) etc.

The above 9 settings would also be achieved by the following MIDI input code (hex - note that the status nibble is here always C, as in the left column of the Header Box; other OPEN status bytes would of course be possible here, too):

1. C0 13 C0 00 (program changes 20 and 1 in MIDI channel 1)
2. C1 14 C1 00 (program changes 21 and 1 in MIDI channel 2)
3. C2 19 C2 00 (program changes 26 and 1 in MIDI channel 3)
4. C3 15 C3 01 (program changes 22 and 2 in MIDI channel 4)
5. C4 17 C4 01 (program changes 24 and 2 in MIDI channel 5)
6. C5 18 C5 02 (program changes 25 and 3 in MIDI channel 6)
7. C6 16 C6 02 (program changes 23 and 3 in MIDI channel 7)
8. C7 18 C7 03 (program changes 25 and 4 in MIDI channel 8)
9. C8 18 C8 04 (program changes 25 and 5 in MIDI channel 9) etc.

The MIDI input code changes PREMs thus: the **MIDI range** shown in diagram 7 - alterable using the plus-minus mouse (⊟) - parallels the PREM range: the lowest MIDI value engenders the lowest PREM value, the highest the highest. Between these two the values are linearly scaled. With e.g. a MIDI keyboard as input device, it might be useful to set the range to its usual **36 ~ 96**: the lowest C (36) will set the PREM to its minimum, the highest C (96) to its maximum value.

The keys Return, Blank and Esc, used for starting (or resuming), interrupting and stopping an execution run are also exported as MIDI signals on MIDI channel 16 (F); these can be received by AUTOBUSK on another computer - if they are set equally in both computers as a **keycommand**, using these keys on the first computer would have the same effect on the second. For instance, in diagram 7, the key command status nibble is E: pressing Return as mentioned at the end of #5 would result in the export of MIDI code EF 09 00 (in hex - 9 is the ASCII code for Tab, in this case for technical reasons an internal subsitute for Return; test this on the MIDI Output Monitor): on another receiving computer set the same way, the effect would be that of one having pressed Return there.

The time shown by AUTOBUSK under **time now** when running and **stopped at** when stopped is also exported as MIDI on channel 16 (F), either explicitly at any time by pressing the key shown in the Help Box for **clock out** or (when running) implicitly at every full minute. If another computer running AUTOBUSK is identically set in this respect, it can be synchronized - 'indefinitely' - with the first, because its clock will be set to the incoming clock value. Given that the clock out status nibble is A, as shown against **clock info** in the diagram, the following code will be sent out after one minute: AF 00 01 AF 00 00 (the 2nd and 3rd bytes form the seconds value, the 5th and 6th the milliseconds - test this on the MIDI Output Monitor).

The last remaining box concerning MIDI input is the Alternative Byte Box, marked with a reversed B: **B**. Here the latch-on mouse (■) can be used to type in a PR©M number, for which the 'alternative' byte, i.e. that not regarded here as the relevant byte in the MIDI input (e.g. for a note-on: 'velocity'), can be used to change the PR©M in the manner described above. Type the same keys as for the Allocation Box right-hand column, e.g. [/] for Parameter 11 (dynamics), as shown (**11**) in the diagram. In this case, the velocity of input note-ons will be used to alter the dynamics: the higher the input velocity, the higher the dynamics. Note: status nibble C ('program change') has no 'alternative' byte.

#9 The Remap Box

Just below the Alternative Byte Box is the Remap Box, unconnected with MIDI input but included in diagram 7 for reasons of clarity. Here, latch-on typed-in nibbles are taken to mean a re-routing of MIDI command types. In the diagram, one sees note-ons (status nibble originally 9) converted in the Left stream into program changes (**C**), in the Middle stream into pitch bends (**E**). Also, note-offs in both these streams have been deactivated by typing a zero (**0**). By setting the MIDI channel numbers in all three streams to the same value, one would hear only the Right stream, subjected to program changes and pitch bends by the other two streams. Typing the original nibble or [-] at the place where it was changed restores the Remap Box at that place to its normal inactive state.

Remapping can also be done to incoming MIDI, i.e. if the [0] button is set, the re-routed command will be sent out, as can be seen if desired in the MIDI Output Monitor. As mentioned before, this is only done when AUTOBUSK is not executing PR©Ms (stopped or interrupted). Also remember, command types currently **OPEN** are not shown in the Input Monitor.

#10 Final Remarks

Independently of Parameter 2 (pulse length), the speed of the computer as a whole can be changed by using the up and down cursor keys (↑,↓) as well as the ClrHome key: the former alter the speed to values between 77 and 300% and the latter restores the original speed of 100%. Exiting AUTOBUSK also restores this original speed.

The temporal performance tolerance (default 40ms) can be altered by the left and right cursor keys (←,→) - this means that if the computer fails to compose and perform any event within the given time value, it is dropped, so as to prevent a possibly resulting sluggishness. If, however, a mass of events is desired, to which a higher priority is given than that of punctuality, the tolerance value can be raised.

AUTOBUSK can be quickly exited at any time though the key given in the HelpBox for exit now!: the ⌫ button is respected and all flashes are stored in the .0 file.

The 9 programmes seen in the PRMPROC box at the bottom right of diagram 3 will be treated in Chapter III.

#1 **ASC** - a text editor

This name stands for any good ASCII text editor, needed for entering **.CTS** and **.MTR** files for input to the programs **HRM** and **IDP** (described below in #2 and #3); it was not found necessary to write a new program for the purpose. **TEMPUS**, written by Manfred Schuelein (© 1987 Creative Computer Design Eltville) is strongly recommended. In lieu of a detailed description of this program, here is a list of command keys as supplied by its author:

Diagram 8 - **TEMPUS** Key Commands

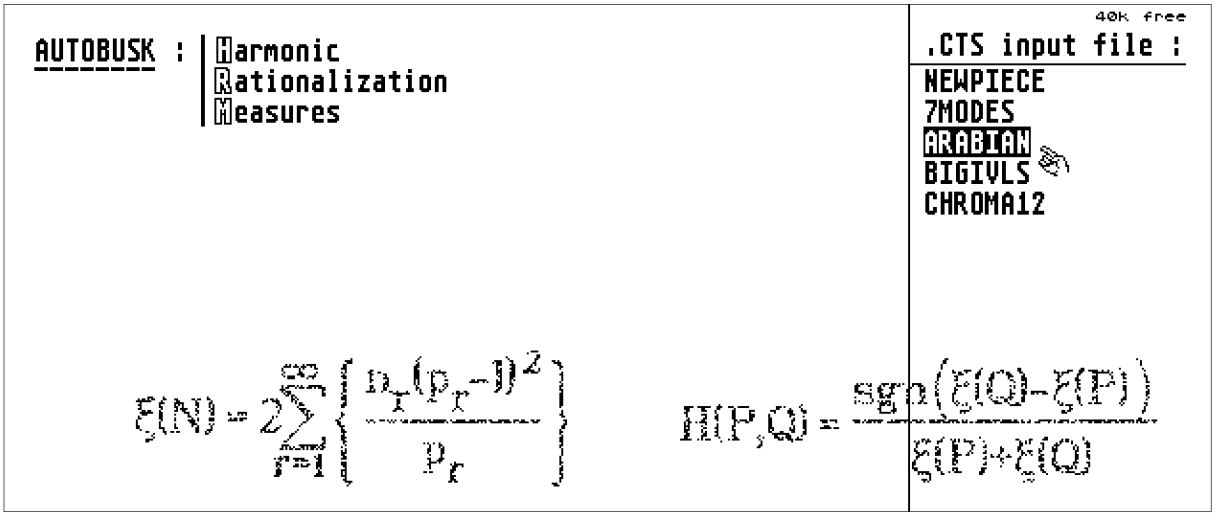
```
{ ****** Keys for the TEMPUS editor ******* }

-------------------------------------------------------------------------
| DESK  |   TEXT     |    MARKEN   |   SUCHEN     |   BLOCK    |   DRUCKEN   |
-------------------------------------------------------------------------
Info:%V  Load     : %X    goto mark:  Find     : ^L  Start : ^B  Print text:^P
         Overload : ^U   %1..%5       Refind   : %L  End   : ^K  Print bl. :%P
         Load in  : %R   set mark  :  F & Rep  : ^E  Inv. M: %H  Take inst.:^J
         Save+Bak : ^S   ^1..^5       -> Line  : %Z  Hide B: ^H  Save inst.:%J
         Save-Bak : %S                -> Page  : ^D  Move  : ^V
         Sava as  : ^W                -> Start : %T  Copy  : %C
         Save Bl. : %W                -> End   : %R  Delete: %Y
         Quit     : ^X                -> Bl.St : %B
                                      -> Bl.End: %K
                                      -> last p: ^Z

-------------------------------------------------------------------------
| SPEZIELL      |   MODUS      |      FKEYS           |     OTHER KEYS      |
-------------------------------------------------------------------------
 Text exp. : ^?  Insert   : %I    F1,^R : Page up         INSERT: Insert space
 Text comp.: %?  Overwrite: ^ü    F2,^C : Page down       UNDO  : Restore line
 Load 8x16 : %M  Autotab. : %A    F3    : Start of line   TAB,^I: Move to tab.
 Load 8x8  : %O  8x16 font: ^M    F4    : End of line     CLR H : Top of page
 Syst. font: %D  8x8  font: ^O    F5    : Word left       ^N    : Insert line
 Wind. over: ^;  Inverse  : %Q    F6    : Word right      ^T    : Del. word R.
 Wind. vert: ^:  Line len.: ^'    F7    : Refind          %'    : Clock On-Off
 Wind. hor.: ^-  Page len.: ^#    F8,^Y : Delete line     ^*    : ASCII table
 Calculator:     Page n   : %U    F9,^Z : Goto last pos.  %1(right):Save+Quit
 Funct.keys: %F  Tab len. : ^G    F10   : Hide block      ^1(r) : Save+Bak&Qt
                 Tab 5    : ^G                           SH+BS , SH+DEL...
                 Tab 8    : %G


(^:Control, %:Alternate)
```

If it is not there already, put the program of choice into the **PREPROC** folder and rename it **ASC.PRG**. It can be started by clicking on the word **ASC** in the **PREPROC** box near the bottom left of the AUTOBUSK screen, mentioned above in Chapter I#6.

#2 **HRM** (Harmonic Rationalisation Measures) - a Scale Compiler

This program - see diagram 9 - receives scales in **.CTS** text format and evaluates their *harmonicity* for AUTOBUSK; for more information on the process, please refer to the bibliography - the decorative formulæ seen in the diagram (provided the small file **HRM.FOG** is in the **PREPROC** folder) are central to the method.

Diagram 9 - The **HRM** Program's Input Menu



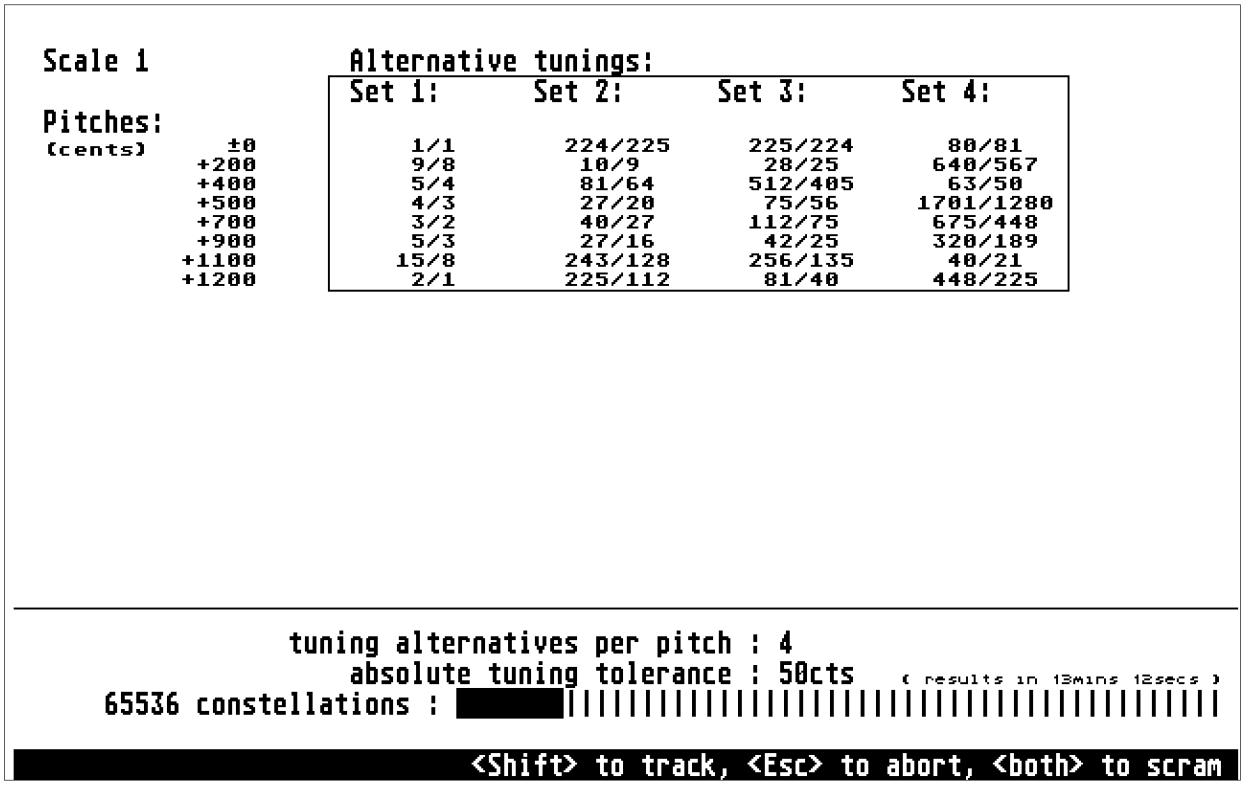
The input format **.CTS** consists of a number of lines of text containing the cent values of the scale degrees; here is for example one octave of the major scale:

0 200 400 500 700 900 1100 1200.

AUTOBUSK does not work on the principle of octavic equivalence; scales can in principle have any range, less or more than an octave. **HRM** can handle upto 24 scale-degrees. However, the more degrees, the longer the program takes; an estimate of the duration is offered at the beginning of the computation of each scale - see diagram 10, bottom right.

Start **HRM** by Ö-ing on its name in the **PREPROC** box near the bottom left of the AUTOBUSK desktop, as said in Chapter I#6. At once the screen in diagram 9 appears. If **HRM.JST** is not in the **PREPROC** folder, **HRM** suggests either creating this file (an interval list) by the program **JST** (see #4) or exiting (**<J> for JST, <Esc> to quit**). If **HRM.JST** is present, Ö- on one of the input files in the menu at right (a Return chooses the first of them; Ö-ing outside the menu moves the program to the previous option: J or Esc). If a file is chosen, the intervals listed in **HRM.JST** are then read, its header (see Chapter IV#3) shown at screen centre. When the **.JST** file has been fully read, press Return for the first pitch set in the **.CTS** file.

Diagram 10 - First scale in the file **7MODES.CTS** - the major scale



Thereupon the screen changes (cf. diagram 11): displayed at top left are the cent values, vertically listed, of the first scale in the chosen **.CTS** file. The cursor blinks below, requesting *tuning alternatives*; a number (upto **4**) is suggested according to the amount of calculations estimated; the larger the scale, the smaller the number of alternatives imposed as the upper limit for calculation (however, upto 4 alternatives can always be viewed: see also below). Here again, the more the alternatives, the longer the computation; mostly 2 alternatives suffice. Any change in the suggested number is effected by typing the desired value.

The other request, named in the line below, is for the *absolute tuning tolerance*. **HRM** works on the principle that a scale given in cents is not necessarily identical with its harmonically workable form - that scale degrees may need in theory to be 'bent' to a nearby harmonic interval in order to make musical sense. Take for example the one-octave major scale seen in diagram 10: a possible rationalisation offered by **HRM** is 1:1 8:9 4:5 3:4 2:3 3:5 8:15 1:2, all of which are relatively harmonic intervals (according to the author's formulæ, shown at the start of **HRM**, these 8 intervals have a harmonicity of ∞, 12.0%, 11.9%, -21.4%, 27.3%, 11.0%, 8.3% and 100%, which is the working basis for AUTOBUSK). The maximum deviation of the above rationalisations from the original cent input is that for the 3rd degree of the scale - it is bent by 14 cents from 400 down to 386. Therefore the higher the tolerance, the more harmonic the result, finally reaching a degree of implausibility.

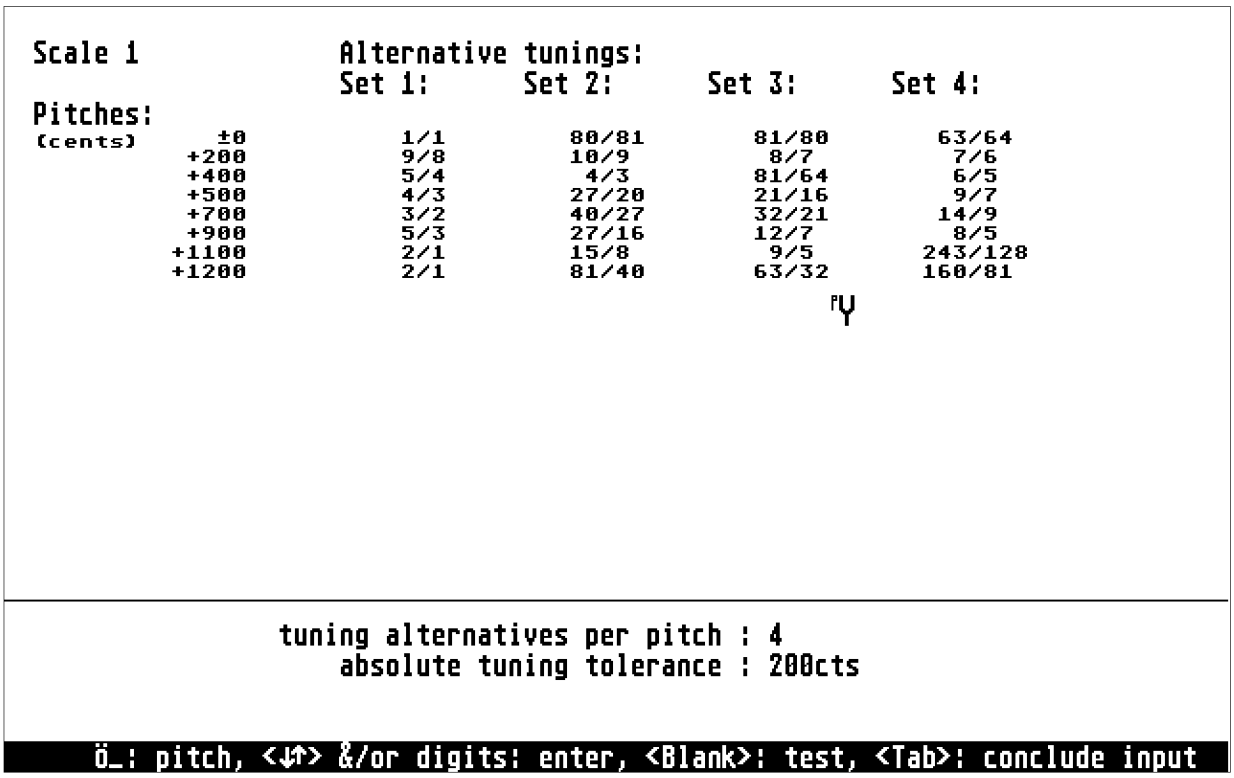
On the other hand, zero tolerance is not borne out by musical experience. The desired tolerance has to be worked out by experiment; **HRM** suggests by default a value half of the smallest interval in the scale. This tolerance is not a sharp boundary but a measure of the Gaussian-type compression used here. If a change in the displayed tolerance is desired, one can move to the line in question by the down cursor key and type in the new value, followed hereby a Return (or a Blank or a Tab - see below). The up and down cursor keys (↑,↓)can be used ad libitum to enter new values for the 2 Parameters.

When satisfied with the 2 values, press either Blank to view the tuning alternatives - this can be done repeatedly, re-changing the values if desired - or finally Tab to go ahead with the rationalisation. As mentioned, the scale shown in diagram 10 is the major, with **4** alternatives and a tolerance of **50** cents; diagram 11 shows the same scale, also with **4** alternatives but a tolerance of **200** cents (!) - note the shift in the choices.

Upto 4 candidates for the rationalisation can be viewed: even if this is done, only the expressly entered number of alternatives for rationalisation (boxed in) will really be considered.

If tuning alternatives are being viewed via Blank, the mouse reappears in the guise of a tuning fork as seen in diagram 11 - click it on any one of the cent values at left or on a ratio in one of the alternative sets to hear its pitch sound on the monitor loudspeaker (**±0** cents and the ratio **1/1** are set at Middle C).

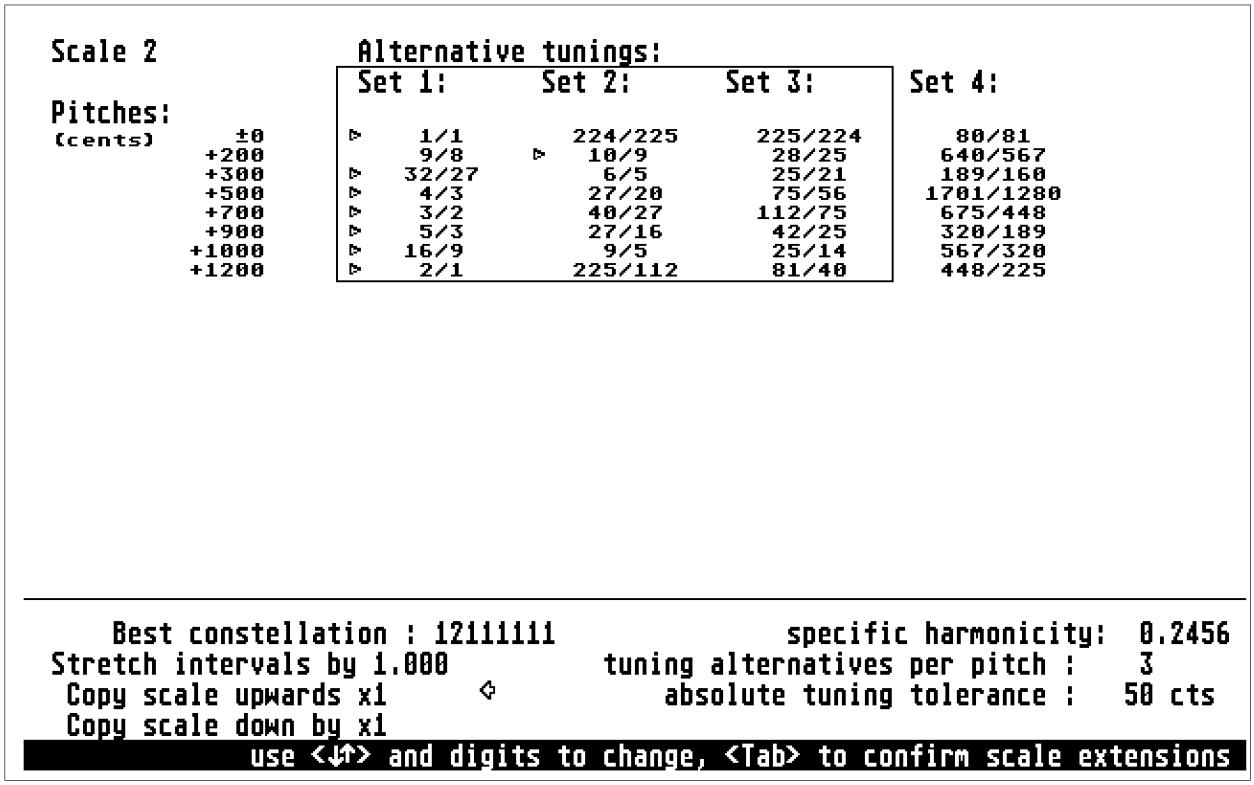
Diagram 11 - The major scale again, with 200 cents tolerance



If Tab has been pressed, computation begins. This is shown as a lengthening bar (see diagram 10 again) accompanied by an occasionally revised estimate of the computational time. As a rule of thumb, this is roughly proportional to the number of alternatives raised to the power of the number of scale degrees, in the given case with the major scale and 4 alternatives $4^8 = 65536$, as shown in diagram 10. Tests reveal that - depending on the computer used - this could take a maximum of 37', whereas 3 alternatives would take 4', 2 alternatives 9" and 1 alternative practically no time at all. One can interrupt the computation at any time by pressing Esc; Shift tracks the computation, showing arrow-heads at the alternatives being currently considered along with a line at top left the length of which reflects the *specific harmonicity* (the reciprocal of the sum of all intra-scalar inharmonicities, themselves reciprocals of the individual harmonicities) of the resultant individual tuning. Pressing Shift+Esc causes the scale to be abandoned altogether, with Left-Shift leaving the user the option of choosing another scale (Return then returns one to the **.CTS** menu) or of leaving **HRM** by pressing Esc, with Right-Shift quitting totally and at once, leaving no trace of an **.HRM** output file.

On completion of computation, arrow-heads point to the 'best tuning', i.e. that with the highest specific harmonicity, also shown as a series of digits meaning the alternative chosen for each scale degree (cf. **12111111** in diagram 12). This row of digits can be combed for an aural check with the tuning fork mouse; when done, pressing any key moves the program to a setup like that actually seen in diagram 12.

Diagram 12 - Completed harmonic rationalisation of the Dorian mode



Now 3 further options are presented (the entry of values is as before, using the ↑ and ↓ cursor keys, digits, . and Return):

1. Stretch intervals by a value, normally per default **1.000** (= no stretch); if a stretch is desired, it could be for microintervallic reasons - suppose the sound generator is so tunable, that a normal chromatic scale played via MIDI into it would sound in successive quarter-tones: in this case enter a value of 2.0, because every semitone in the scale which has been compiled would then be 'disguised' as a whole-tone in the **.HRM** output file, sounding through the sound generator's altered tuning as a semitone. In this situation, the degrees of a compiled quarter-tone scale would be disguised as semitones and would

sound as quarter-tones. AUTOBUSK has been essentially designed for a output pitch grid of semitones. If, say, a scale involving 17-tone equal temperament is being compiled, enter here the value **1.4167** (= 17/12), so that the microinterval between successive scale degrees comes out as semitones for AUTOBUSK's use. Irrespective of stretching, the harmonicities computed for the unstretched intervals are preserved; it is only the intervals' size that is disguised.

2. Copy the scale upward a number of times, normally per default **1** (= no copy). Because **HRM** cannot handle scales with more than 24 pitches and indeed a scale of this length would take ages (here two alternatives could take as much as two weeks!), it was found expedient to compute smaller units and copy them as many times upwards as desired; for instance

(a) compute a one-octave major scale and copy it **8**-fold, so as to cover 8 octaves.
(b) Or compute a chromatic scale ending at the perfect 5th, and copy this **14**-fold to cover just over 8 octaves.
(c) Or truncate the same scale at the tritone and copy it **16**-fold. There are some interesting differences between these 3 approaches - the following is a table of one-and-a-half octaves rationalised in this way at a tolerance of **50** cents:

scale degree cents	scale range: (a) tritone	(b) fifth	(c) octave
0	1/1	1/1	1/1
100	16/15	16/15	16/15
200	10/9	9/8	10/9
300	32/27	6/5	6/5
400	5/4	5/4	5/4
500	4/3	4/3	4/3
600	64/45	45/32	64/45
700	1024/675	3/2	3/2
800	128/81	8/5	8/5
900	2048/1215	27/16	5/3
1000	16/9	9/5	16/9
1100	256/135	15/8	15/8
1200	4096/2025	2/1	2/1
1300	16384/7593	135/64	32/15
1400	8192/3645	9/4	20/9
1500	16384/6834	12/5	12/5
1600	1024/405	81/32	5/2
1700	16384/6075	27/10	8/3
1800	16384/5695	45/16	128/45

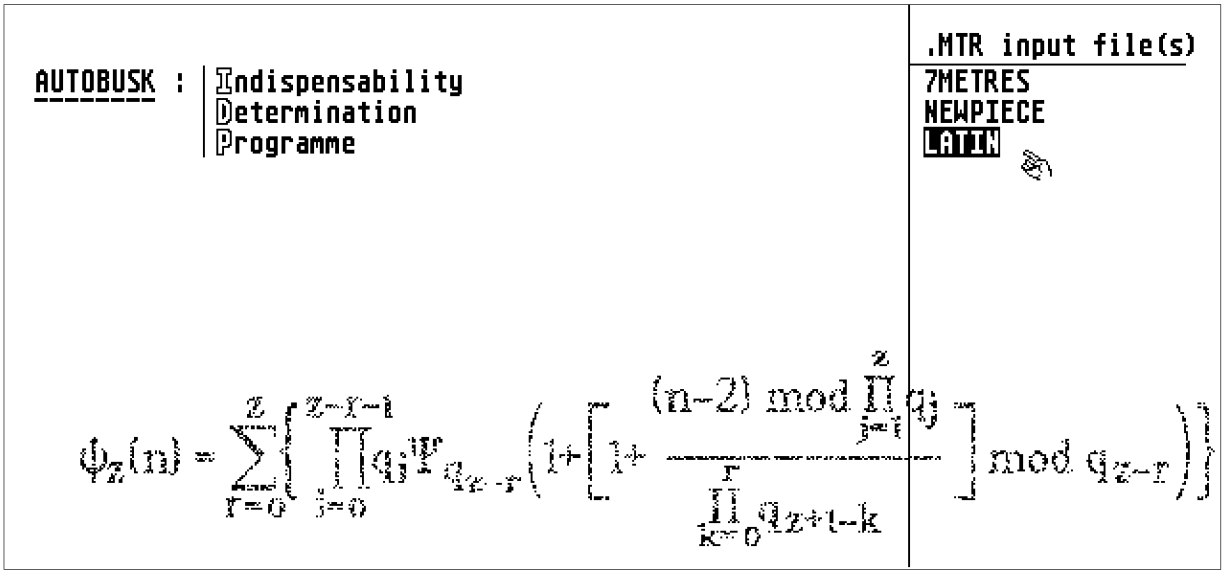
Notice the double-tritone (64/45 × 64/45 = 4096/2025) in column 'a' at 1200 cents as against the 5th+4th and 8ve (2/1) in the other 2 columns, also - at 900 cents - the tritone+minor 3rd (64/45 × 32/27 = 2048/1215) in 'a', the 5th+major 2nd (3/2 × 9/8 = 27/16) in 'b' and the 5-limit major 6th (5/3) in 'c'. Note also that the 'a' interval at 1300 cents, which should be a double-tritone+minor 2nd (4096/2025 × 16/15 = 65536/30375 = 2.15756) has been 'simplified' to 16384/7593 (= 2.15778) in order to get the unwieldy original numerator-denominator figures down to below the arbitrary limit 30000.

3. Copy the scale downward a number of times, normally per default **1** (= one copy). Analogous to the preceding option, this one works downwards. Thus an octave copied 8-fold up and down gives a range of **-9600** to **+9600** cents.

The maximum values for interval stretch and scale copy are interdependent - lowering the one raises the other: the product of these two - multiplied by the scale's highest cent value - may not exceed 12700.

The user signals satisfaction with all 3 option settings by a Tab; after the intervals are stretched and/or copied, **HRM** offers three possibilities - either redo the current scale by pressing Backspace or go on - by a Return - to the next one (or, if this was the last scale in the **.CTS** file, to the program's end - in the latter case one sees that the **.CTS** file has been compiled to a **.HRM** file of the same name and that a rather self-explanatory **.LOG** file has also been stored, recording all results of the **HRM** work session). A Shift added to this Return causes the program to go on all the way to the end without stopping. Finally, typing Esc at

Diagram 13 - The **IDP** Program's Input Menu



Very quickly all the metres in the file are gone through and one is told that an **.IDP** (text) file of the same name as the **.MTR** input is done. For example, the given 12/16 file is compiled thus: 11 0 4 8 2 6 10 1 5 9 3 7, reflecting the computed relevant importance or indispensability of the 12 pulses in the metre. By way of comparison, a 6/8 metre on the ♪-level, input as 2 3 2, is computed to be 11 0 6 2 8 4 10 1 7 3 9 5, as seen in the **.IDP** file.

#4 **JST** (Justified Systemic Tuning) - a Lister of Rational Intervals

The numbers forming interval ratios are products of powers of prime numbers. Time was, when most musicologists admitted only the first two primes, 2 and 3, to these deliberations to form what is called '3-limit' or Pythagorean tuning. Later the next prime, 5, was added - this was the 5-limit system. Other primes e.g. 7, 11 and 13 are conceivable, if ever less plausible, as intervallic building blocks - it seems the brain's ability to assimilate numbers decreases as the therein contained primes increase. As an attempt to model this, the author developed a formula for the *indigestibility* of numbers in 1978, seen decoratively displayed at the start of the program **HRM**; according to it, the indigestibility of the numbers 1 to 16 are

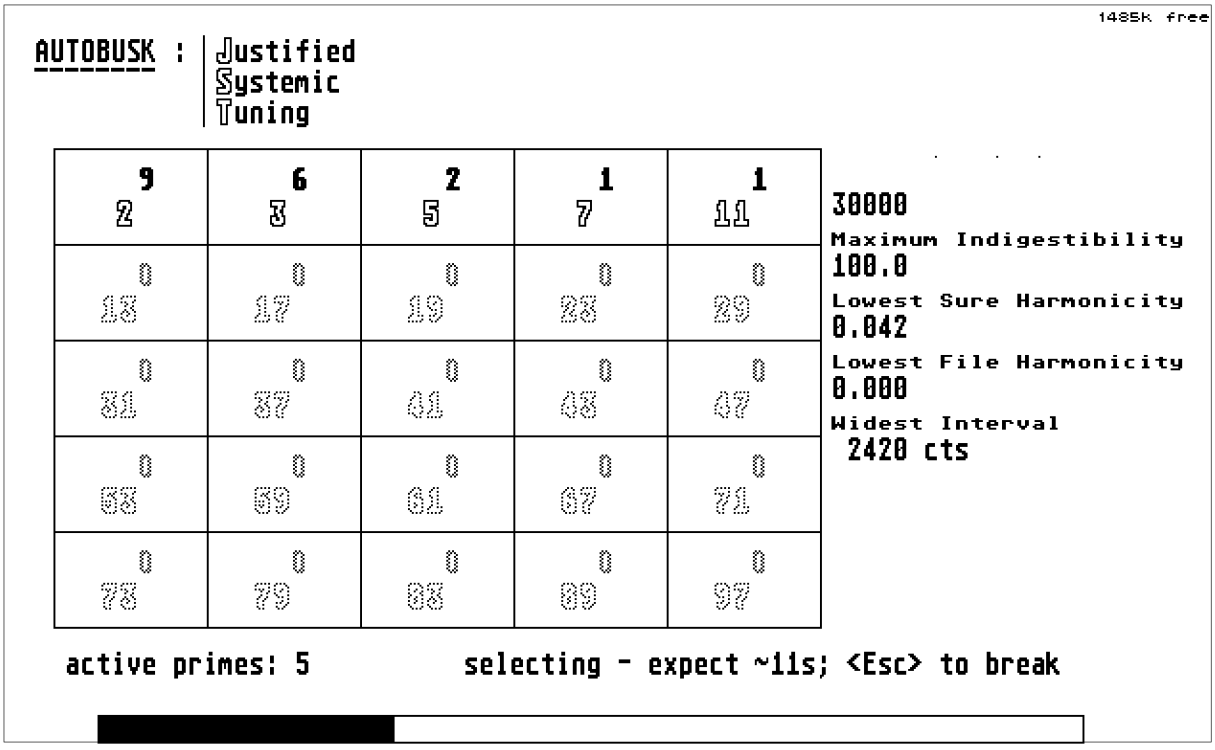
0.0, 1.0, 2.7, 2.0, 6.4, 3.7, 10.3, 3.0, 5.3, 7.4, 18.2, 4.7, 22.2, 11.3, 9.1 and 4.0,

respectively; notice the increasing progression of prime indigestibilities (1.0, 2.7, 6.4, 10.3, 18.2 and 22.2) and that the indigestibility of a product of numbers is the sum of the individual indigestibilities, e.g. the values for 2 (1.0) and 3 (2.7) added together give the value for 2×3 = 6 (1.0 + 2.7 = 3.7).

Based on the above, the author also came up with a formula for the *harmonicity* of an interval, given its ratio as two component numbers mutually prime to each other (also seen in **HRM**): it is the reciprocal of the sum of the 2 indigestibilities - so the harmonicity of the fifth 2:3 is the reciprocal of 1.0 + 2.7, i.e. 1/3.7 = 0.273. For more information on this, please consult the bibliography again.
The program **HRM**, rationalising scales defined in cents to a set of plausible ratios, bases its decisions on a sufficiently exhaustive list of intervals expressed as ratios (numerator/denominator, the ratios' *components*) and stored in the file **HRM.JST**, itself made by the program **JST**; the 6 constraints imposed on this list are

1. the maximum power of the prime numbers in the ratio components
2. the largest number allowable as ratio component
3. the maximum indigestibility the ratio components may have
4. the lowest harmonicity value to be considered
5. the lowest harmonicity value to be filed
6. the pitch range within which the intervals are investigated

Diagram 14 - The **JST** Program's Starting Desktop



Look at diagram 14 - though more than the first 5 prime numbers (2, 3, 5, 7 and 11) appear unnecessary to the author as intervallic building blocks the first 25 prime numbers are shown, with attendant powers (zero powers and primes in grey imply absence of the prime). However, only 8 primes are allowed at one time.

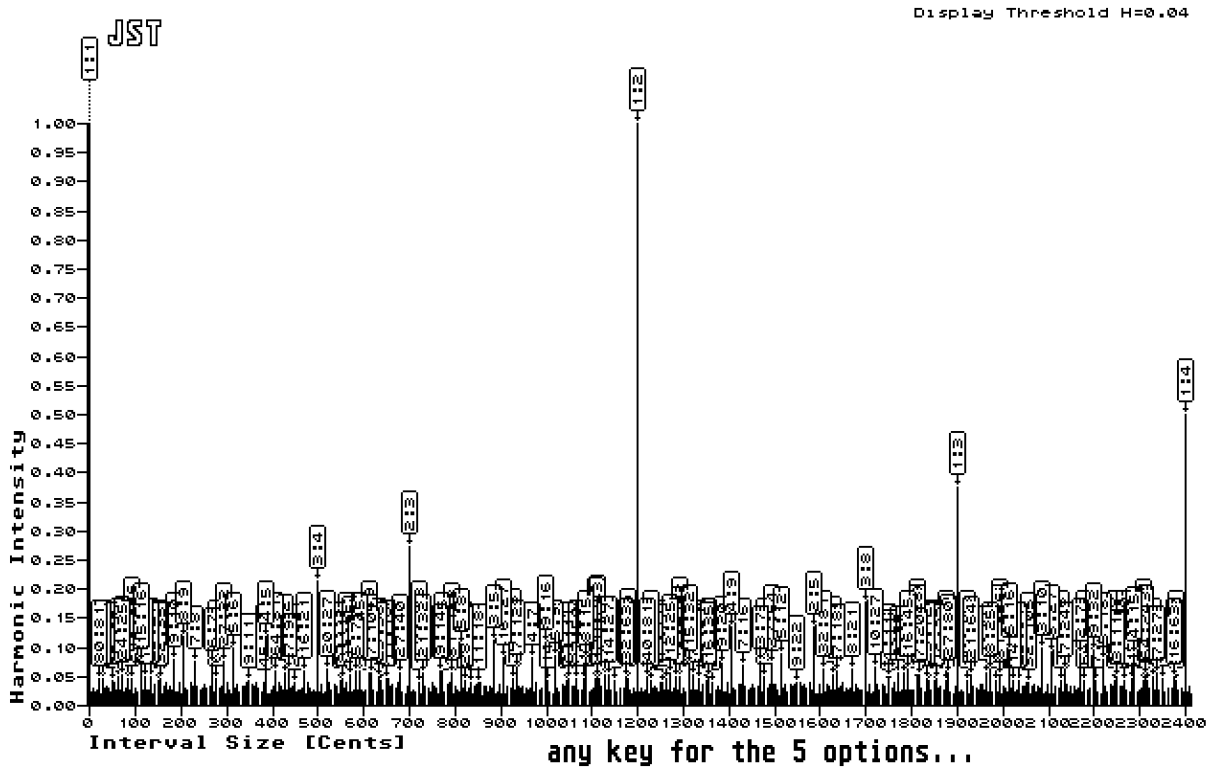
As a starting default situation, the first 5 primes are accorded the powers **9**, **6**, **2**, **1** and **1**, respectively; this series corresponds, as the author elsewhere explains (see the bibliography), to a sure harmonicity threshold of **0.042**. If however these powers are arbitrarily changed - they can be increased and decreased by Ö_ and _Ö, respectively - no harmonicity threshold can be assumed per se and the word **void** is displayed on the right in lieu of a value.

JST's other settings (points 2-6 above) can also be changed by means of Ö/_Ö or by pointing the mouse at the value to be changed and typing in the new value - this appears further to the right and is installed on pressing Return. For points 2, 3 and 6 permissible values range from 1 to 30000; the range for point 4 is from 0.03 to 1 and for point 5 from 0 to 1. Note that changing the values at points 4 and 6 the prime powers and the estimated computation time are altered (for example setting point 4 at minimum and 6 at maximum activates the first 8 primes and could take over 3 hours) - the latter does not include the time taken for sorting the intervals prior to listing.

If satisfied with the settings, computation can be started by pressing Shift + Return. A lengthening band shows the progress made. This process and the subsequent sorting can be interrupted any time by pressing Esc: in this case 2 options follow, as expressed in the instruction line - one can press R to restart or E to exit. Otherwise, if and when computation is complete, the number of intervals checked, kept and stored is indicated; in addition to this, 5 options are presented, each to be effected by typing one of the following 5 letters:

D for 'draw': here the resulting ratios are graphically plotted in terms and in order of their *harmonic intensity* (the absolute harmonicity value) against their pitch in cents. A harmonicity threshold is requested, above which the ratios are expressly named: first they are only plotted as vertical lines - when the harmonicities transcend the requested threshold, 'banners' showing the ratios of the pertinent intervals as fractions are additionally displayed, thereby sometimes partially obscuring earlier banners of lesser harmonicity - see diagram 15.

Diagram 15 - Graphic Display of the Ratios evaluated by **JST**



T for 'tabulate': the contents of the completed evaluations are shown in tabular form, 40 intervals at a time, in terms of size (cents), harmonicity×10000 and ratio; to see more, press Return - the additional use of the Control, Left-Shift or Alternate keys causes a jump of 10, 100 or 1000 windows forwards of 40 intervals each. Backspace moves 10, 100 or 1000 windows back in a similar way. Esc returns to the main 5-option menu.

S for 'store': the same contents are stored to disk as **HRM.JST**

R for 'restart': all evaluations are abandoned - the program returns to the start

E for 'exit': everything is abandoned and **JST** is exited.

Each of the following programs can be started by Öing on its name in the **PRMPROC** box at the bottom right (see again diagram 3). When a program is done, the bottom instruction line says **..any key to quit**, which is the only choice. Quitting returns the user to the AUTOBUSK main screen.

#1 **EDIT** - a Text Editor

Please (re-)read Chapter II#1: what was said about **ASC** also holds here.

#2 **FILL** - the Flush Interpolator

This program makes a **.PRM** score file by interpolating the PREM values between 2 flush files set a given time apart. See diagram 16 - a record of all files currently present in the **FLUSHES** folder is shown in a squat square at screen centre: flushes numbered 00 to 99 which are absent are displayed small and grey, those present large and black.

Diagram 16 - **FILL**'s Starting Desktop

If there are none, the message **NO FLUSHES!...** is shown in a strip above the square; in this case, the bottom instruction line says **any key to quit**, which is the only choice. If there are flushes, the instruction line suggests the selection of 2 (time-tag-alterable) flushes, followed by Return to confirm (or Esc to quit); the strip above the square indicates Ö_ for the start flush and _Ö for the final one.

Assuming flushes to be present, move the mouse over to any one of the large black numbers in the square - the mouse's arrow-head shape changes to a pointing hand. Ö_ on the number: a strip appears under the square listing the clicked number as a start flush. Now change the allocated time in the right of the strip, shown by its default value **0.000"** - to do this, move the mouse there: the mouse shape changes to that of a text cursor (ɪ). _Öing raises the digit held as it were by the 'pincer' at the right edge of the cursor and Ö_ing lowers the digit.

Now _Ö on another flush number: a second strip appears bearing this number as the final flush; the default time allocated is **60.000"** - change this and/or the start flush time if so desired. When satisfied, press Return: the upper strip requests an output file name - enter this and close with a Return, or, if any name will do, simply press Return without entering a name. The chosen name, or, in the latter case, the name **test.prm** preceded by the disk and folders containing the file is shown in the upper strip. At the same time, the progress of the interpolation is graphically displayed in the central square.

Note - since (as mentioned above) only Parameter values are quantities, it is not meaningful unless expressly desired to interpolate non-Parameter values, which are simply labels; thus in normal cases, both start and final flushes have identical non-Parameter values.

#3 **FUSE** - the Synchronizer

This program 'merges' or 'synchronizes' two or more **.PRM** scores to form one, by interlacing the input files' command lines according to their time-tags. On starting, a menu appears on the left, showing a maximum of 20 **.PRM** files. If there are none, or only one, the instruction line says **NO FILES** (or **LESS THAN TWO FILES) FOUND! ..any key to quit**, the only choice. Otherwise, as the instruction line suggests, use Ö to select, Ö to deselect some of these (a triangular pointer jutting out to the right of the file name indicate selections), then press Return: a box opens up at screen centre, to which arrows leading from the selected files point. It requests an output file name - enter this and close with a Return, or, if any name will do, simply press Return without entering a name. The chosen name, or, in the latter case, the name **test.prm** preceded by the disk and folders containing the file is shown in the box. At the same time, a time counter box appears on each of the arrows leading from the input files, showing the progress made with the interlacing process - see diagram 17.

Diagram 17 - **FUSE** in progress

When done, the output file name is ticked, the overall length is indicated and the instruction line reads **..any key to quit**.

#4 **JOIN** - the Concatenator

This program concatenates two or more **.PRM** scores to form one, by copying the first file verbatim, then cumulating subsequent time-tags. First a menu appears at left, showing maximally 20 **.PRM** files. If there are under two, the instruction reads **NO** (or **LESS THAN TWO**) **FILES FOUND! ..any key to quit**, the only choice. Otherwise Ö/_Ö to (de)select some of these (shown by a triangular pointer). A count of a file's entries is noted in a little box to the right of its name, clearly **1** at first; if a file is picked a second time, **2** appears in the box. When satisfied, press Return: the order of (upto 30) input files (and thus of concatenation) is noted sequentially by letters of the alphabet. At screen centre a box opens requesting an output file name - enter it and close with a Return, or, if any name will do, simply press Return with no name. The name (in the latter case, '**test.prm**') preceded by the disk and folders holding the file is shown in the box. Also, an arrow to the outfile box from the file being currently processed appears with a time counter on it, showing the progress. The file's sequential letter turns bold - see diagram 18.

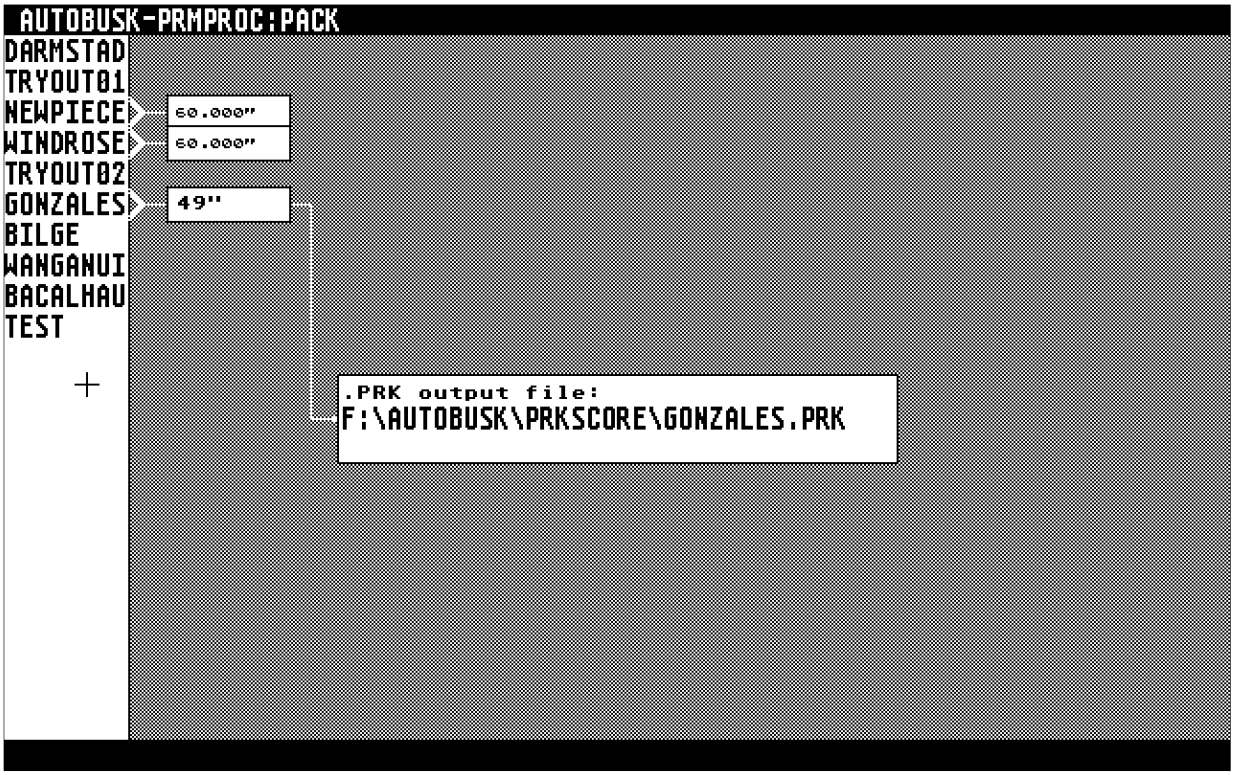
Diagram 18 - **JOIN** in progress

When done, the output file name is ticked, the overall length is indicated and the instruction line reads **..any key to quit**.

#5 **PACK** - the Compressor

This program compresses a **.PRM** file by five-ninths its size into **.PRK** binary format, which can be read by AUTOBUSK in real time. On starting it, a menu appears on the left, showing a maximum of 20 **.PRM** files. If there are none, the instruction line says **NO FILES FOUND! ...any key to quit**, which is all one can do then. Otherwise, as the instruction line suggests, use Ö_ to select, _Ö to deselect some of these (a triangular pointer jutting out to the right of the file name indicates selection). When all files to be packed are chosen, press Return: a box opens up at screen centre, containing the name of the output file concerned, being the same as that of the input with the extension swapped from **.PRM** to **.PRK**. At the same time, an arrow leading from the file being currently processed to the outfile box appears with a time counter box on it, showing the progress made with the compression - see diagram 19.

Diagram 19 - **PACK** in progress



#6 **PART** - the Separator

This program splits a **.PRM** file into two, according to time-sections as well as PRGM elements and range-segments selected by the user; **.PRM** command lines containing and/or corresponding to these selections are extracted and written into a file (called the '**X**' file), the remainder being written into another (the '**R**' file). Logically, putting these 2 into the program **FUSE** would cause **PART**'s input to be reconstructed.

On starting **PART**, a menu appears on the left, showing a maximum of 20 **.PRM** files. If there are none, the instruction line says **NO FILES FOUND! ...any key to quit**, which is all one can do then. Otherwise, as the instruction line suggests, use Ö to select one of these (a triangular pointer jutting out to the right of the file name indicates selection). When the file to be parted is chosen, press Return: a large chart opens up with 3 main columns, corresponding to the 3 streams - **L**(eft), **M**(iddle) and **R**(ight) of AUTOBUSK. Over on the left, a vertical strip denotes PRGMs by their **.PRM** symbols - subsets of the non-Parameters and the range of the Parameters can be selected or deselected by Ö-/Öing the symbols in the 3 main columns - they then turn from grey to black):

~ (streams, selectable as a whole, i.e. Öing selects the whole column),

***** (8 PRGMs in one, each selectable as a complete entity -

P for outset pulse, **R** for time-latch, **C** for stream switch

S for seconds reset, **M** for milliseconds reset,

@ for explicit play [=note-on], **#** for explicit damp [or note-off],

! for explicit force [or MIDI-velocity] - see Chapter IV#6 for more),

S (scales, 6 in all per stream, individually selectable),

M (metres, 6 in all per stream, individually selectable), followed by the numbers

1 to **9** (Parameters 1 to 9),

0, **"** and **^** (Parameter 10, 11 and 12),

I (sound/controller),

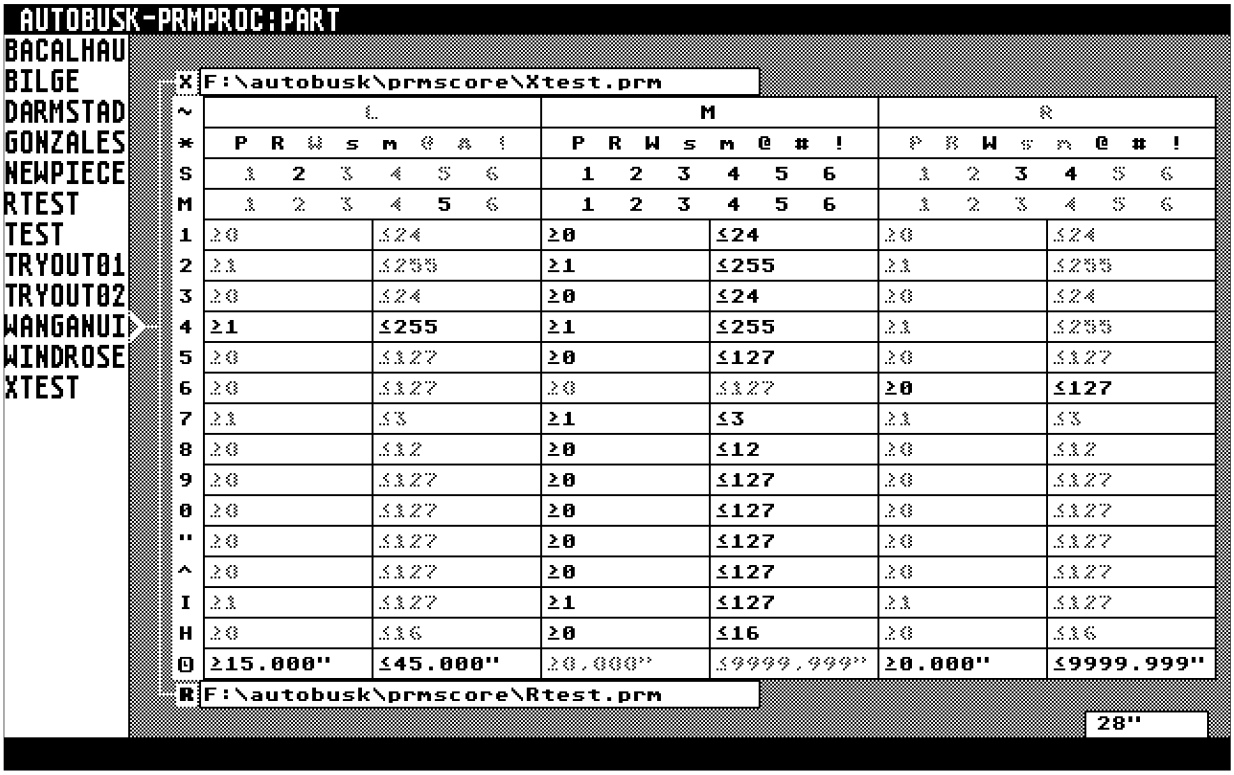
H (MIDI channel) and

@ time - in all 12 Parameters as well as in sound/controller and MIDI channel, range-segments can be marked out by entering minimum and maximum values (initially indicated by default, e.g. for Parameter 6 in all 3 streams as **≥0** and **≤127**). For changing the minimum value of one of these PRGMs in one of the streams, move the mouse into the left half of the stream's column, Ö to select it, keep it poised there and type in the new value (echoed in a strip under the chart); end with a Shift+Return - the value (curtailed if excessive) then appears

in its appropriate place in the chart. The bottom row of the chart offers a time-select option - if desired, enter in a similar fashion an initial and a final time here to select for the stream in question a time-section during which all events will be considered for extraction according to the other constraints higher up in the chart. When all choices have been made, move the mouse out of the chart and press Return: a longish strip appears just above the chart, asking for an output **X**-file name. Enter this and close with a Return, or, if any name will do, simply press Return without entering a name. The chosen name, or, in the latter case, the name `Xtest.prm` preceded by the disk and folders containing the file is shown in the strip. Thereupon the same happens just below the chart, asking for an **R**-file name. Do the same - pressing Return without entering a name results in `Rtest.prm` being chosen and shown in the strip.

Now the program begins to run - all black-displayed elements and range-segments within the selected time-sections are extracted from the input file and written into the **X**-file, the remainder into the **R**-file. Progress is shown in a little time counter box at the bottom right of the screen and by the alternate emboldening of the **X** and **R** symbols at left - see diagram 20.

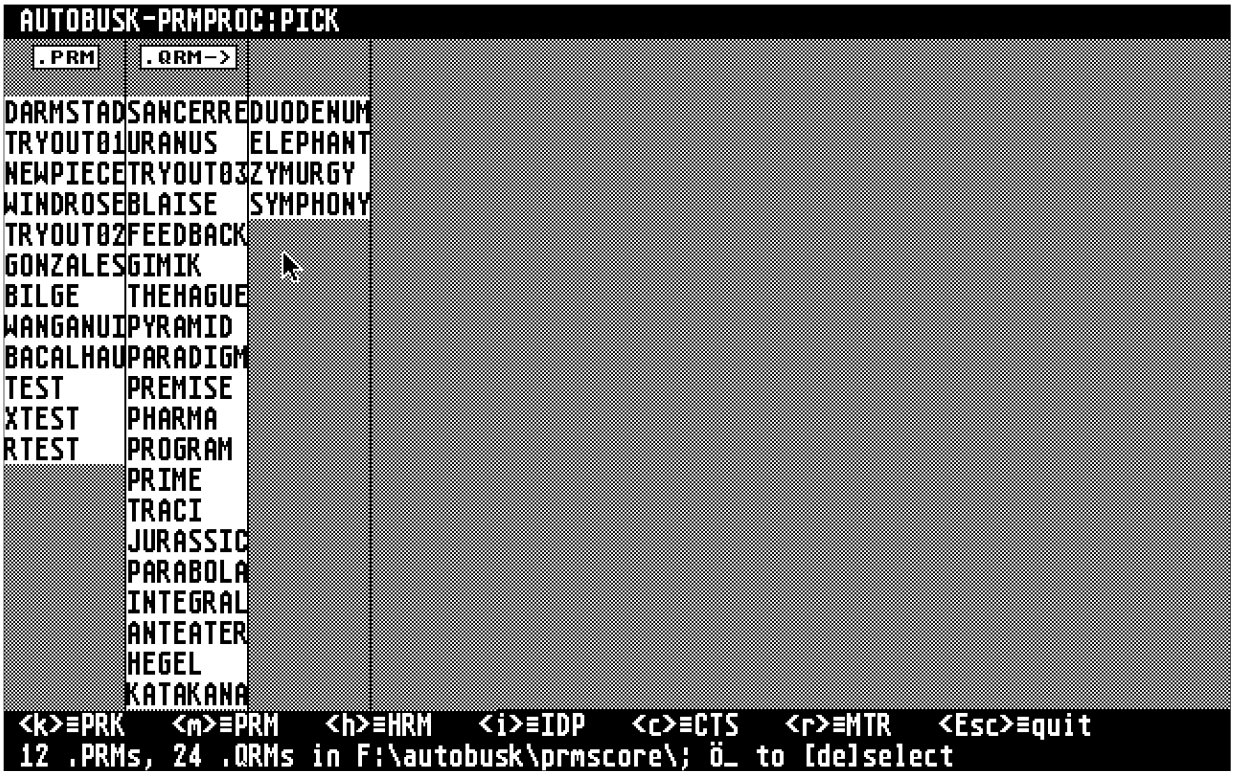
Diagram 20 - **PART** in progress



#7 **PICK** - the Selector

AUTOBUSK does not like long file selector menus; it limits simultaneously visible files in its **PREPROC/PRMPROC** program groups to 20, in the central program to 9. If there are more files in the pertinent folders, a pre-selection has to be made, a purpose served by this program. First, the instruction line lists 6 data file formats (see Chapter IV for details) with the 6 keys [K], [M], [H], [I], [O] and [R] for the appropriate menu and [Esc] to quit - see diagram 21.
Type one of these keys for a menu: if **NO FILES FOUND!**, type another menu key or [Esc] to quit; otherwise the left column shows upto 9 files of type **.PRK/.HRM/.IDP** or 20 files of the other types. To the right are listed files of **.QRM/.QRK/.IRM/.JDP/.DTS/.NTR** format, all simply hidden files of the above-mentioned 6 formats, with the first letters alphabetically one ahead, e.g. **PRK→QRK**). This program swaps files between the chosen and the hidden lists.

Diagram 21 - **PICK**ing, as exemplified by **.PRM/.QRM** files

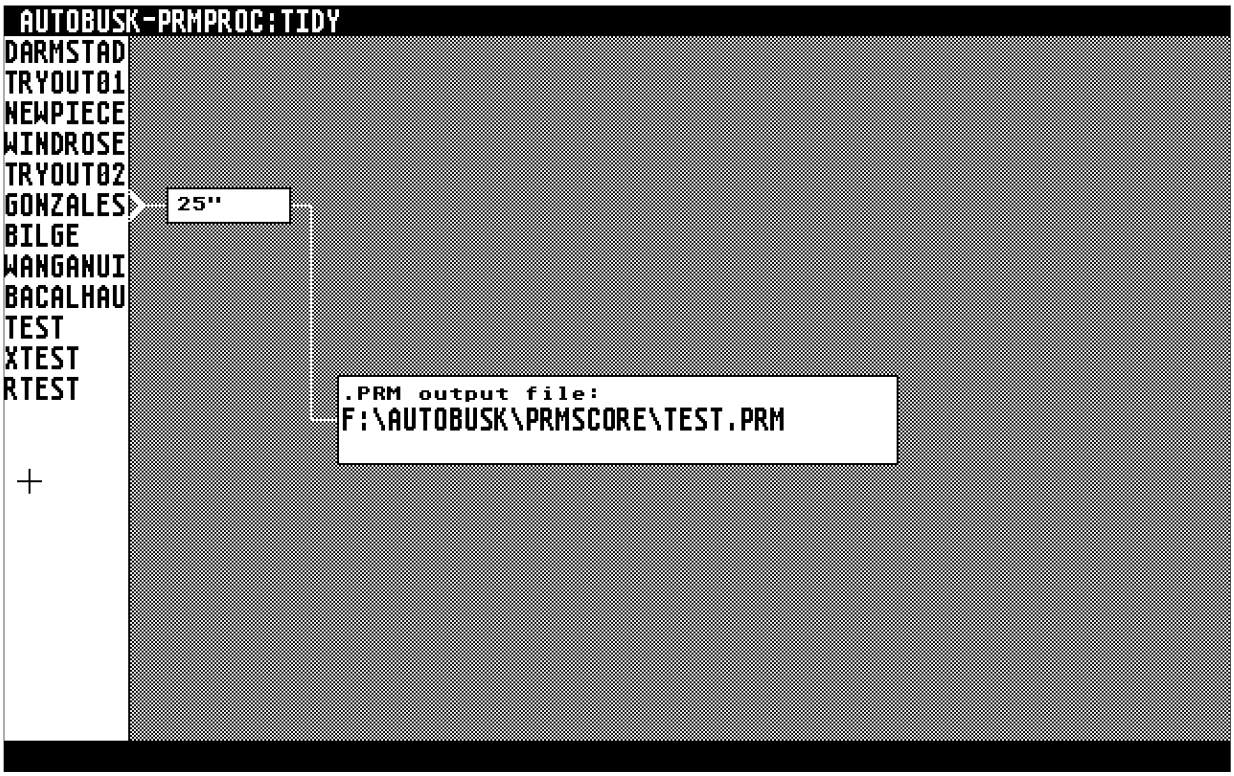


To hide a (left-listed) chosen file, Ö_ on it - it will disappear and reappear in the hidden list in the columns on the right; to choose a hidden file, Ö_ on it - it will move to the left column. If the chosen list is full (9 or 20 files), no more will be admitted to it. When done, press [Esc] to quit.

#8 **TIDY** - the Cleaner-Upper

This program removes redundancies in a **.PRM** file; they may have been caused e.g. by concatenating files containing identical command lines near to the 'seams'. First a menu appears on the left, showing a maximum of 20 **.PRM** files. If there are none, the instruction line says **NO FILES FOUND! ...any key to quit**, the only choice. Otherwise Ö to select one of these (indicated by a triangular pointer to the right of the file name). When the file to be tidied is chosen, press Return: a box opens up at screen centre, to which an arrow leading from the selected file points. It requests an output file name - enter this and close with a Return, or, if any name will do, simply press Return without entering a name. The chosen name, or, in the latter case, the name **test.prm** preceded by the disk and folders containing the file is shown in the box. At the same time, a time counter box appears on the arrow leading from the input file, showing the progress made - see diagram 22.

Diagram 22 - **TIDY** in progress



When done, the output file name is ticked and the instruction line reads **..any key to quit**.

#9 **VARY** - the Modifier

This program alters attributes of PR©Ms (including their time-tags) in 3 modes:

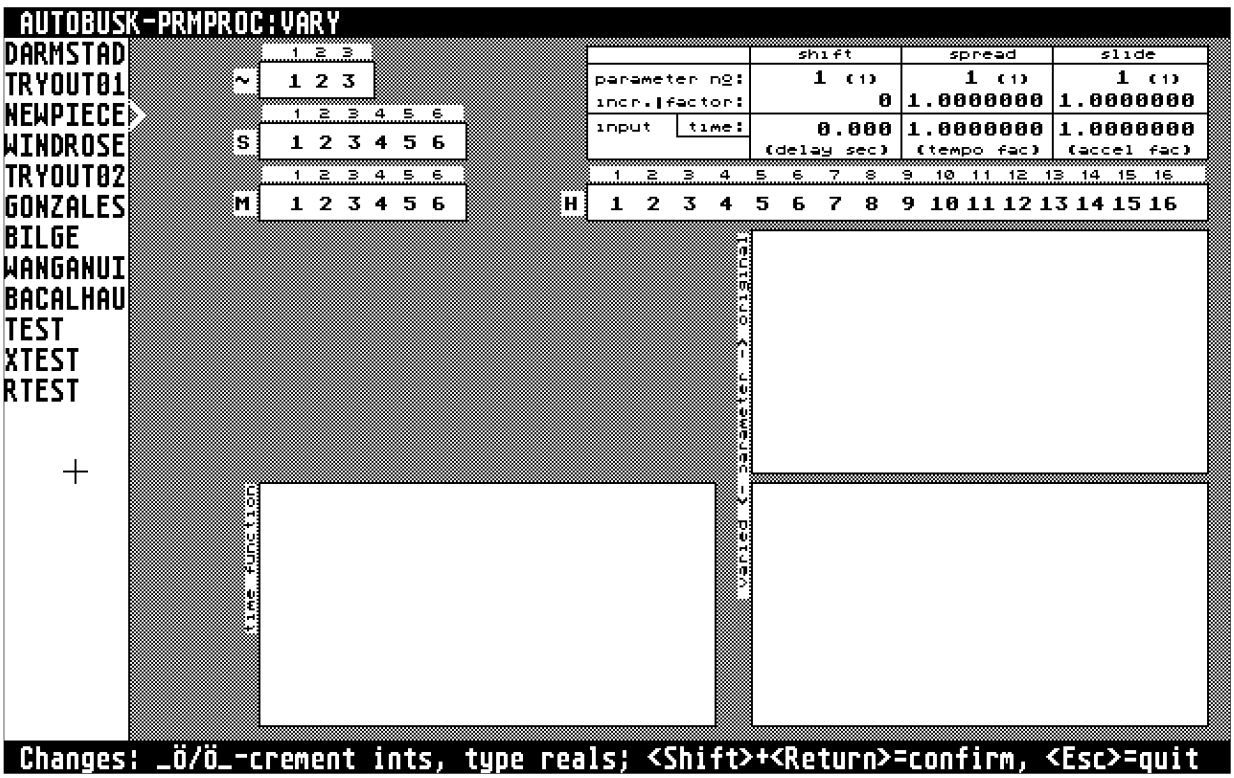
shift: all values (including time-tags) are cremented by a given constant amount.

spread: the distances of all values from a central 'pivot' are increased or decreased by a given constant factor; in the case of time-tags, their pace is altered to be faster or slower.

slide: the PR©Ms start at their original values but gradually move to a given final value, which they reach at file end; in the case of time-tags, their pace is accelerated or decelerated from the original to a given final speed.

At first, a menu appears on the left, showing a maximum of 20 **.PRM** files. If there are none, the instruction line says **NO FILES FOUND! ...any key to quit**, the only choice. Otherwise Ö_ to select one of these (a triangular pointer to the right of the file name indicates selection). When the file to be varied is chosen, press Return: several boxes appear on the screen - see diagram 23 - they are treated here from top to bottom.

Diagram 23 - **VARY**'s Starting Desktop (after picking an input file)



Four Material Replacement Boxes, marked at left by ~ for streams, S for scales, M for metres and at right by H for MIDI channel: here one sees 3 streams, 6 scales, 6 metres and 16 MIDI channels, listed in increasing order from left to right. Any one of these 3+6+6+16 elements can be replaced by another in its PRCM by _Ö- or Ö_ing to respectively increase or decrease it.

The Parameter Transformation Box at top right: its 3 columns correspond to the 3 variation modes, shift, spread and slide, as labelled in the top row.

The next lower row accepts Parameter numbers, one per mode, for simultaneous transformation; _Ö or Ö_ to respectively increase or decrease them - the small symbol in brackets reflects the .PRM code for the 12 Parameters (1234567890"^) - see also #6 above and Chapter IV#6.

To enter values for the next row (crements for shift, factors for spread/slide):
for shift, move the mouse to the displayed value (the preset default is 0) and _Ö or Ö_ to respectively increase or decrease it.
for spread/slide, poise the mouse at the pertinent current value (the preset defaults are 1.0 each) and type in the desired new value - permissible characters are +, -, . and 0 to 9. If properly entered, this value is shown in the bottom left of the box just under the word input. Press Return to enter it - it jumps to its proper place (where the mouse is), replacing the previously shown value.

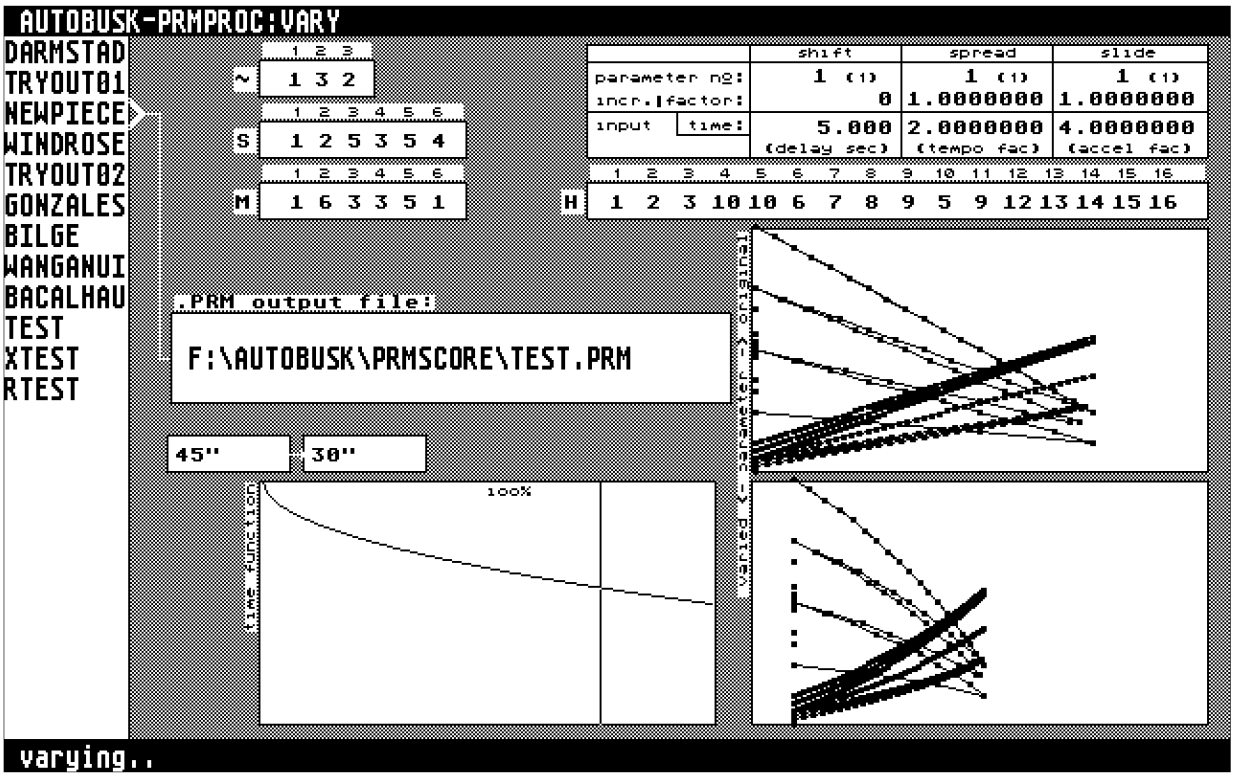
The bottom row is for time-tag transformations, the 3 columns having the same functions shift (delay in seconds), spread (average tempo change factor) and slide (acceleration factor). Enter values as above by poising the mouse at the pertinent current value (the respective preset defaults are 0.0, 1.0 and 1.0) and type in the new value - it, too, is shown under input. Close with Return. Note that if the tempo increases, acceleration must exceed tempo change and vice versa (i.e. if tempo change>1 then acceleration>tempo change; if tempo change<1 then acceleration<tempo change): logically, the new average tempo must be closer to the old than to the final tempo; whichever of the two is entered first, the other is automatically corrected to the nearest acceptable value - try it.

On being entered, the values jump to their proper place at the mouse, replacing the previous ones. At the same time the time function box at bottom left shows a straight line or curve, reflecting the way the time-tags are altered with time: see diagram 24, in which the average tempo doubles and the acceleration factor rises 4fold - the time function box curve starts on the left at 100% (the original tempo) and descends exponentially to 50% at right. The combination of factors tempo change and acceleration determines the curvature of the time function.

If satisfied, press Shift+Return (the Shift finalises this last Return). The Alternate key may also be used to suppress the variation display (see below); if it is not or if sliding and/or spreading, the input file's time-length is ascertained - all its time-tags upto the end are displayed in a time-counter box near the Time Function Box. Then a box opens at centre, to which a line from the file-name leads, asking for an output file name - enter this (or don't) and close with Return. The file name (in the latter case **test.prm**) preceded by its disk and folders is shown in the box. At the same time, the process starts, its progress shown

1. by another counter box, the two time boxes displaying original/varied time (different only if time transformations were entered),
2. by a moving line in the Time-Function Box and
3. (only if not suppressed by Alternate as said above) the original and varied Parameter values in two boxes at right: see diagram 24.

Diagram 24 - **VARY** in progress



If varied Parameter values exceed their normal limits (e.g. by bad spreads), they are stored in **VARY.LOG** in the **PRMSCORE** folder and truncated in the output file; this is announced both when it happens and at program end. When done, the output file is ticked and the instruction line reads **..any key to quit**.

#1 **.CTS**

Raw Pitch Scales for input to program **HRM**: a text file with an (in principle) "unlimited" number of lines, each containing a scale expressed in cents, e.g. `0 200 400 500 700 900 1100 1200` for the 8 degrees of one octave of the major scale. The maximum number of scale degrees is 24, though this results in an exceedingly long computation. The maximum cent value is `12500` (over 10 octaves, the range of human hearing).

#2 **.HRM**

Compiled Pitch Scales, output from the program **HRM** for input to AUTOBUSK's central program: a binary file with a 2-byte header followed by an "unlimited" number of 4-integer (i.e. 4×2-byte) packets, of which only the first 6 are accepted by AUTOBUSK . The header is the number `8192` (`2000` hex); if this is incorrect, then AUTOBUSK assumes the **.HRM** file to be obsolete or faulty and imposes a chromatic scale as default pitch material instead. The 4 integers in each following packet are the input interval **size** in cents, the **numerator** and **denominator** of the rationalised interval and finally the latter's estimated **harmonicity×1000** (s. the bibliography and Chapter II#2), e.g. `700`, `3`, `2` and `273` for the rationalised tempered *5*th.

#3 **.JST**

Interval Vocabulary, output from the program **JST** for input to the program **HRM**: a binary file with a header of 7, 9 or more (odd-numbered) integers followed by an "unlimited" number of 4-integer (i.e. 4×2-byte) packets. The header: the number of primes used followed by each prime and its attendant maximum absolute power (in the supplied **HRM.JST** file, this is 6 primes, followed by `2`, `11`, `3`, `7`, `5`, `3`, `7`, `2`, `11`, `1`, `13` and `1`, meaning 2^{11}, 3^{7}, 5^{3}, 7^{2}, 11^{1} and 13^{1}); then follow 4 more integers, viz.

- the maximum numerator and denominator in the interval ratios,
- the maximum indigestibility ×100 (s. Chapter II#4 - e.g. `10000` means 100.0),
- the range of the interval vocabulary in cents and
- the number of intervals in the file.

The 4 integers in each subsequent package are (cf.#2) the interval **size** in cents, the **numerator** and **denominator** of the rationalised interval and finally the latter's estimated **harmonicity×1000** (s. the bibliography and Chapter II#2), e.g. `700`, `3`, `2` and `273` for the rationalised tempered *5*th.

#4 **.MTR**

Raw Pulse Metres for input to program **IDP**: a text file with an "unlimited" number of lines, each containing a metre expressed as a succession of prime-number divisors, e.g. 2 2 3 for a 12-pulse metre structured like 12/16. There can be upto 9 divisors and the the number of pulses in the metre should not exceed 1000.

#5 **.IDP**

Compiled Pulse Metres, output from the program **IDP** for input to AUTOBUSK's central program: a text file containing an "unlimited" number of metres, each expressed as a succession of prime-number factors (e.g. 2×2×3: for a 12-pulse metre structured like 12/16), followed by the indispensabilities (s. Chapters I#5 and II#3) of the individual pulses, spilling over into several lines if necessary.

#6 **.PRM**

Basic PR©M Score for indirect input to AUTOBUSK's central program: a text file comprising an "unlimited" number of lines, each containing a command expressed as the following 5 elements -

seconds and **milliseconds**, as two integers forming the time-tag of the command, the pertinent **stream** - the characters L, M or R (for 'left', 'middle' or 'right'), and the PR©M **tag** and **value** applying to the stream - one of the following 25 characters for the former and a byte for the latter, indicated here in brackets:

- H the MIDI channel allocated to the stream (values 0 for monitor sound chip, 1-16 for real external MIDI channels)
- I the sound, or in case of a remap of pitch to controller, the controller number allocated to the stream (values 0-127)
- M the number of the applicable metre (values 1 to maximally 6)
- O for switching pitch/rhythm ostinato on or off (values 0 for both off, 1 for rhythm only, 2 for both on)
- P for outset pulse (values 1 to number of pulses in currently incumbent metre)
- R for time-latching attack times to the nearest of another stream (value 1-3 for the other stream to be latched onto)
- S the number of the applicable scale (values 1 to maximally 6)
- W for switching the stream off or on (values 0 or 1, respectively)
- 0 Parameter 10 - pitch range in semitones (values 0-127)
- 1 Parameter 1 - metriclarity (values 0-24)
- 2 Parameter 2 - pulse length in milliseconds (values 1-255)

3 Parameter 3 - eventfulness (values 0-24)
4 Parameter 4 - event length in pulses (values 0-255)
5 Parameter 5 - melody scope in semitones (values 0-127)
6 Parameter 6 - tonic pitch in MIDI pitch-numbers (values 0-127)
7 Parameter 7 - chordal weight (values 1-3)
8 Parameter 8 - harmoniclarity (values 0-12)
9 Parameter 9 - pitch centre in MIDI pitch-numbers (values 0-127)
¨ Parameter 11 - dynamics, the central force (values 0-127)
^ Parameter 12 - attenuation, maximum force deviation (values 0-127)
s seconds reset - AUTOBUSK's clock will be reset to this seconds value (0-32767)
m milliseconds reset - the clock will be reset to this milliseconds value (0-999)
@ explicit play - AUTOBUSK will expressly play this pitch (value 0-127)
! explicit force - explicit play is henceforth with this force (value 0-127)
explicit damp - AUTOBUSK will expressly damp this pitch (value 0-127)

These last 3 PR©M types make AUTOBUSK a sequencer! - try the supplied file **INVENTIO.PRK** - deliberately different pulse lengths here yield interestingly staggered rhythmic results (the pulse length of **20** ms given here will require the temporal performance tolerance to be increased to somewhat above **40ms** by [→]).

#7 **.PRK**
Compressed PR©M Score for direct input to AUTOBUSK's central program: a binary file made by putting a **.PRM** file into the **PRMPROC** program **PACK**. The file consists of an "unlimited" number of 4-integer (8-byte) packets containing
1. seconds,
2. milliseconds,
3. (high byte) stream number and (low byte) PR©M-tag,
4. PR©M value

#8 **.α**
'Alformation': a small text file containing special user-defined higher-level MIDI and other settings, installed when AUTOBUSK starts; it contains 10 lines with the following initialising information (except in line 1, all bytes are in hex; please also [re-]read Chapter I#7-8):
1. Four bytes - the channel for flush recall (decimal; in case of no **.α** file being present, the default setting is 16) the [1] button setting: value 0 means 'off', 1 means 'on' (the default is 'off') the bottom and top values for the MIDI input range (decimal; the defaults are 0 and 127)

2. The 18 status bytes (hex) used for MIDI input allocation change for the 18 PR€Ms (the default is CCCCCCCCCCCCCCCCCC).
3. The 18 status bytes (hex) allocated to MIDI input for PR€M value change (the default is '88888888898898888B' - note that Parameters 6 and 9 have been set to 'note-on' [9], all others to 'controller' [B]).
4. The 2 status bytes (hex) used for key-command and clock-out MIDI export (the default is EA); furthermore, the presence of the characters 0, R, S and/or T to the right mean correspondingly switching the 0, R, S and T 'on', 'on', 'off' and 'off', respectively (the default settings are to the contrary).

5-7. 8 to 18 bytes each line (hex): the first 8 indicate the remap setting in order of the 8 status bytes 8-F (the default setting 89ABCDEF thus means 'no change'); lines 5-7 represent the left, middle and right streams. The remaining bytes indicate the byte strings to be exported from the Explicit MIDI-Out Box (the default is B*n*7B00, where *n* is the MIDI channel of the stream).

8. 8 characters indicate the setting of the Alternative Byte Box in order of the 8 status bytes 8-F: '-' means 'no choice', else a Parameter tag (0-9/"/^) would be found here, e.g. '-^------' means "use status byte 9's alternative byte (force) to control Parameter 12" (the default is '--------').
9. 16 characters indicating the stream combinations controllable by each of the 16 MIDI channels in turn - the letters L to Q are available for this, L for left, M for middle, N for left+middle, O for left+right, P for middle+right, Q for all 3 streams and R for right (the default setting is 'QQQQQQQQQQQQQQQQ', i.e. all streams everywhere).
10. 16 characters indicating the PR€M to be controlled by each of the 16 MIDI channels in turn. The 12 Parameters are given by their tags '0-9/"/^'. Furthermore, the 4 non-Parameters are given by W for **switch**, S for **scale**, M for **metre** and P for **outset pulse** (the default setting is 1234567890"^SMPW).

#9 .Φ

Flash File: a binary file consisting of upto 99 packets of 54 bytes each; every packet reflects a flash, containing for each of the 3 streams in turn the values of all 18 PR€Ms. If all flashes in AUTOBUSK have been removed, this file is removed; if even one PR€M configuration is flashed to memory by the Φ button, the file is reopened and written on exiting the program.

#10 **.K**

Fonts: a binary file consisting of 756 packets of 8 integers each. Each packet reflects a symbol used in AUTOBUSK's various displays - see diagram 25; each integer contains the 16 horizontal pixels in one of the 8 graphic lines of the symbol in descending order. The symbols range from **0** to **300**, **A** to **G**, **Cb1** to **Gb9**, **8•8** to **F•F** (used in the Header Box) and **00** to **FF**; by accessing this file in a properly designed interface, the user could redesign the symbols.

Diagram 25 - AUTOBUSK's special font, as stored in the **.K** file

0	27	54	81	108	135	162	189	216	243	270	297	Eb0	Gb2	Bb4	C#7	Eb9	A•F	E•A	0D	28	43	5E	79	94	AF	CA	E5
1	28	55	82	109	136	163	190	217	244	271	298	Fb0	Ab2	Bb4	Db7	Fb9	B•8	E•B	0E	29	44	5F	7A	95	B0	CB	E6
2	29	56	83	110	137	164	191	218	245	272	299	F#0	Ab2	Cb5	Eb7	F#9	B•9	E•C	0F	2A	45	60	7B	96	B1	CC	E7
3	30	57	84	111	138	165	192	219	246	273	300	Gb0	Bb2	C#5	Eb7	Gb9	B•A	E•D	10	2B	46	61	7C	97	B2	CD	E8
4	31	58	85	112	139	166	193	220	247	274	A	Ab0	Bb2	Db5	Fb7	8•8	B•B	E•E	11	2C	47	62	7D	98	B3	CE	E9
5	32	59	86	113	140	167	194	221	248	275	B	Ab0	Cb3	Eb5	F#7	8•9	B•C	E•F	12	2D	48	63	7E	99	B4	CF	EA
6	33	60	87	114	141	168	195	222	249	276	C	Bb0	C#3	Eb5	Gb7	8•A	B•D	F•8	13	2E	49	64	7F	9A	B5	D0	EB
7	34	61	88	115	142	169	196	223	250	277	D	Bb0	Db3	Fb5	Ab7	8•B	B•E	F•9	14	2F	4A	65	80	9B	B6	D1	EC
8	35	62	89	116	143	170	197	224	251	278	E	Cb1	Eb3	F#5	Ab7	8•C	B•F	F•A	15	30	4B	66	81	9C	B7	D2	ED
9	36	63	90	117	144	171	198	225	252	279	F	C#1	Eb3	Gb5	Bb7	8•D	C•8	F•B	16	31	4C	67	82	9D	B8	D3	EE
10	37	64	91	118	145	172	199	226	253	280	G	Db1	Fb3	Ab5	Bb7	8•E	C•9	F•C	17	32	4D	68	83	9E	B9	D4	EF
11	38	65	92	119	146	173	200	227	254	281	Cb1	Eb1	F#3	Ab5	Cb8	8•F	C•A	F•D	18	33	4E	69	84	9F	BA	D5	F0
12	39	66	93	120	147	174	201	228	255	282	C#1	Eb1	Gb3	Bb5	C#8	9•8	C•B	F•E	19	34	4F	6A	85	A0	BB	D6	F1
13	40	67	94	121	148	175	202	229	256	283	Db1	Fb1	Ab3	Bb5	Db8	9•9	C•C	F•F	1A	35	50	6B	86	A1	BC	D7	F2
14	41	68	95	122	149	176	203	230	257	284	Eb1	F#1	Ab3	Cb6	Eb8	9•A	C•D	00	1B	36	51	6C	87	A2	BD	D8	F3
15	42	69	96	123	150	177	204	231	258	285	Eb1	Gb1	Bb3	C#6	Eb8	9•B	C•E	01	1C	37	52	6D	88	A3	BE	D9	F4
16	43	70	97	124	151	178	205	232	259	286	Fb1	Ab1	Bb3	Db6	Fb8	9•C	C•F	02	1D	38	53	6E	89	A4	BF	DA	F5
17	44	71	98	125	152	179	206	233	260	287	F#1	Ab1	Cb4	Eb6	F#8	9•D	D•8	03	1E	39	54	6F	8A	A5	C0	DB	F6
18	45	72	99	126	153	180	207	234	261	288	Gb1	Bb1	C#4	Eb6	Gb8	9•E	D•9	04	1F	3A	55	70	8B	A6	C1	DC	F7
19	46	73	100	127	154	181	208	235	262	289	Ab1	Bb1	Db4	Fb6	Ab8	9•F	D•A	05	20	3B	56	71	8C	A7	C2	DD	F8
20	47	74	101	128	155	182	209	236	263	290	Ab1	Cb2	Eb4	F#6	Ab8	A•8	D•B	06	21	3C	57	72	8D	A8	C3	DE	F9
21	48	75	102	129	156	183	210	237	264	291	Bb1	C#2	Eb4	Gb6	Bb8	A•9	D•C	07	22	3D	58	73	8E	A9	C4	DF	FA
22	49	76	103	130	157	184	211	238	265	292	Bb1	Db2	Fb4	Ab6	Bb8	A•A	D•D	08	23	3E	59	74	8F	AA	C5	E0	FB
23	50	77	104	131	158	185	212	239	266	293	Cb0	Eb2	F#4	Ab6	Cb9	A•B	D•E	09	24	3F	5A	75	90	AB	C6	E1	FC
24	51	78	105	132	159	186	213	240	267	294	C#0	Eb2	Gb4	Bb6	C#9	A•C	D•F	0A	25	40	5B	76	91	AC	C7	E2	FD
25	52	79	106	133	160	187	214	241	268	295	Db0	Fb2	Ab4	Bb6	Db9	A•D	E•8	0B	26	41	5C	77	92	AD	C8	E3	FE
26	53	80	107	134	161	188	215	242	269	296	Eb0	F#2	Ab4	Cb7	Eb9	A•E	E•9	0C	27	42	5D	78	93	AE	C9	E4	FF

#11 **.MDK**

MIDI file in the author's own MIDIDESK text file format. Each line contains 5-6 elements:

1. seconds
2. milliseconds
3. track - an ASCII character from '=' (#61) to '|' (#124), 96 in all
4. command (a character: **P** for 'play', **D** for 'damp', **S** for 'sustain', **V** for 'volume', **W** for 'pitch wheel' and **I** for 'instrument' (= 'program change').
5. first data byte
6. second data byte (only used for 'play')

A Concise Glossary of Terms used in this Manual (most of them also in the Index)

General standards (only as a reminder)

- hexadecimal — Numeric system based on 16 digits: [0123456789ABCDEF] – e.g. 3E7 = 999
- bit — Smallest information unit; range 0-1 (2 values)
- nibble — 4 bits, expressable as one hexadecimal digit; range 0-F or 0 to 15 (16 values)
- byte — 8 bits or 2 nibbles (e.g. 63 hex = 99); range 00-FF or 0 to 255 (256 values)
- integer — 2 bytes, 4 nibbles or 16 bits; range -32768 to +32767 (65536 values)
- ASCII — 256-symbol set, in which some 95 (e.g. [!"#%+-], [0-9], [A-Z] &c.) are standard
- MIDI — 256-number set: [00-7F]=music data (pitches &c.), [80-FF]=status (e.g. 'play!')
  - status — Command type (e.g. 'play!') given by the 1st nibble in a MIDI data byte stream
  - channel — One of 16 paths given by the 2nd nibble in a MIDI data byte stream
- pixel — Point on the computer screen, known by x/y-coordinates & colour
- octave — Pitch interval resulting from the doubling or halving of a frequency
- semitone — One-twelfth of an octave (frequency factor $\sqrt[12]{2}$)
- cent — One-hundredth of a semitone (frequency factor $\sqrt[1200]{2}$)

Mouse

Shape changes according to ↓boxes

- Ö_/_Ö — Click on the left/right mouse button
- pick/spurn — Ö_ on a file name to pick it, outside the menu to reject all
- ✹ (inactive mouse) — Only visible, not to be clicked or poised
- 🖉 (file-pick mouse) — Indicating possible file-selection by Ö_ing (↓Selector Box &al)
- ⊟ (plus-minus mouse) — For (de-/in-)crementation by Ö_/_Ö (↓PRGM Box &al)
- ◘ (one-click mouse) — Inviting a Ö_to set or unset a button (↓Top Left Buttons, ↓Ostinato)
- ■ (latch-on mouse) — Slipping into nearby places for typed-key input (↓Header Box &al)
- 🖳 (busy mouse) — Requesting patience until computation is done
- poising — Holding or pointing the mouse at a given spot without clicking

Operating AUTOBUSK

- executing — The program is composing; use [Blank] to interrupt or [Esc] to stop
- stopped — The program has been stopped; use [Return] to go to the start or [!] to exit
- interrupted — The program has been interrupted; use [Esc] to stop or [Return] to resume
- instruction line — Line at screen bottom, telling the user what to do or what's going on
- start time — Moment (in seconds) from which time is counted &/or a ↓.PRK file is read
- stop time — Moment (in seconds) at which the program was interrupted or stopped
- time indicator — Number near bottom right indicating the above two items, e.g. 8.375"
- speed of the clock — Changeable as a whole by [↑]/[↓]; [ClrHome] for normal speed
- performance tolerance — Allowed deviation from due time; default 40ms; alter by [←]/[→]
- exiting — Use [!] when stopped or interrupted to leave the program altogether

Top-Left Buttons — Set/unset by the one-click mouse (◘)

- [0] .IDP/.HRM/.PRK — To call the Selector Box for files with these extensions
- [R] — To start/stop ↓.MDK file recording to disk
- [S] — To stop/re-start MIDI output at any time
- [T] — To ignore/re-heed the computer clock while executing
- [α] — To read/write the ↓.α file on-line or to write it on leaving the program
- [0] — To allow the program (when not executing) to export incoming MIDI
- [0] — To ↓flash a ↓PRƐM-configuration to internal memory (at most 99 times)

  flash — One of at most 99 internally [0]-stored ↓PRƐM-configurations

  flush — A flash, flushed by ↑_Ö to the FLUSHES folder on disk as e.g. #09.PRM
- [»] — To re-set the ↑start time of an execution
- [1] — To automatically pick the first Selector Box files next time the program starts

Boxes — For alteration or display

- PRƐM Box (🖴) — Displaying the current value of a total of 18 ↓PRƐMs

  PRƐM — Collective term for ↓Parameters, ↓Routes & ↓Material

  tag — Symbol allotted to the PRƐM, one of the following 18: □HIMOPRSW0123456789"^□

  value — Number not exceeding one byte capacity (range 0-255)

  (de)activation — The ↑one-click mouse (◘) makes a PRƐM open/immune to change

  - = / × — Symbols just left of the PRƐM Box indicating openness/immunity

  Parameter — 12 central controls, set independently in each of 3 ↓streams:

  - metriclarity — Ranging from 'ametric' (or 'syncopated') to having a clear pulse
  - pulse length — Length in milliseconds of the current pulse
  - eventfulness — Ranging from silent to time-saturated (every pulse attacked)
  - event length — Ranging from one pulse ('staccato') to 255 pulses ('legato')
  - melody scope — Maximum interval (semitones) by which the melody may jump
  - tonic pitch — Keynote (non-octave repeating) to which the ↓scale is allocated

    octave number — The program counts Middle-C as the start of Octave #4
  - chordal weight — Number of simultaneous pitches in the ↓stream
  - harmoniclarity — Ranging from 'atonical' (or 'atonal') to a clear key-feeling
  - pitch centre — Note central to the pitch activity of the ↓stream
  - pitch range — Maximum permissible semitones each side of the pitch centre
  - dynamics — Average MIDI 'velocity' in the current ↓stream
  - attenuation — Maximum deviation from the average MIDI 'velocity'

  Route — Path from the act of composition to the resulting sound;

  - sound/controller — The 'program-changed' sound - or controller number used
  - MIDI channel — The MIDI channel along which the current stream is sent

  Material — The tangible starting-point of the compositional process;

  - stream — 3 independent MIDI event sequences, termed Left/Middle/Right
  - metre number — One of at most 6 pulse sets, defined in terms of ↓stratification
  - scale number — One of at most 6 pitch sets, defined in ↑cents from the keynote
  - outset pulse — The pulse of the incumbent metre at which execution begins

- Header Box (■) — 2 columns of 18 ↑MIDI status nibbles each, to externally MIDInfluence
  - left: the Stream & PRɆM allottment (↓Allocation Box)
  - right: the PRɆM values (↑PRɆM Box)

- Allocation Box (■) — 2 columns of 18 symbols each, the left & right respectively being
  - Stream allocation: **L**eft, **M**iddle, **R**ight, **▣** (L+M), **▤** (L+R), **▥** (M+R) & **⊞** (=all 3)/
  - PRɆM allocation: Parameters **1-12**, also **scale**, **metre**, **pulse**, **sound**, **midi**, **switch**
  - **headers**: Information at lower centre-right for the Header & Allocation Boxes
    - **USED / OPEN**: MIDI-status-nibbles now used / still available for allocation
    - **PstREaMtag**: PRɆM-tag &/or stream accessible via Header Box left column
    - **PREM value**: PRɆM value alterable via Header Box right column
    - **keycommand**: MIDI-status-nibble for external operation by keyboard activity export
    - **clock info**: MIDI-status-nibble for external synchronizing by clock news export
    - **MIDI range**: Expected useful range of MIDI-input's ↓'relevant byte'

- MIDI Input Monitor — Shows MIDI input ↓'relevant bytes' (x) against MIDI ↑channel (y)

- Scale Display Box — Shows upto 6 scales by name (e.g. **major**) or as semitone-steps

- Metre Display Box — Shows upto 6 metres by name (e.g. **2/4**) or as ↓stratification

- Help Box — Shows special (language-dependent) keys devoted to the following causes:
  - **exit now!**: To leave the program immediately (Eng./Ger.: [%])
  - **show ping**: To beep on the first beat of the metres (Eng./Ger.: [^P])
  - **wipe midI**: To empty the MIDI Input Monitor (Eng./Ger.: [^K])
  - **show midO**: To write/stop writing to the MIDI Output Monitor (Eng./Ger.: [^L])
  - **clock out**: To export explicit clock news on demand (Eng./Ger.: [^0])
  - **send Lstr**: To export **L**-stream ↓Explicit MIDI Output (Eng./Ger.: [^R])
  - **send Mstr**: To export **M**-stream ↓Explicit MIDI Output (Eng./Ger.: [^F])
  - **send Rstr**: To export **R**-stream ↓Explicit MIDI Output (Eng./Ger.: [^V])
  - **send all3**: To export all-stream ↓Explicit MIDI Output (Eng./Ger.: [$])
  - **Lostinato**: To start/stop **L**-stream ↓(pitch+)rhythm ostinato (Eng./Ger.: [^U])
  - **Mostinato**: To start/stop **M**-stream ↓(pitch+)rhythm ostinato (Eng./Ger.: [^J])
  - **Rostinato**: To start/stop **R**-stream ↓(pitch+)rhythm ostinato (Eng./Ger.: [^M])
  - **timelatch**: To equate a stream's next attack time to another's (Eng./Ger.: [Delete])
  - **pulslatch**: To equate a stream's next pulse number to another's (Eng./Ger.: [~])

- **PRMPROC** Box (✎) — 9 programs in the like-named ↓folder, Ö_-picked to process **.PRM** files

- Selector Box (✎) — To select input files of type ↓**.IDP**, ↓**.HRM** & ↓**.PRK**; Ö_ to ↑pick/spurn

- Alternative Byte Box (■) MIDI input's ↓'alternative' byte (e.g. 'velocity') for a PREM
  - relevant byte — The first data byte (except in 'controllers') is 'relevant' here
  - alternative byte — The 'non-relevant' data byte: nos. [2221-222] for status [89ABCDEF]

- Remap Box (■) To swap output MIDI status if wanted, e.g. 'note-on' becomes 'pitch-wheel'

- Hexagon Shows the key-pair which also (in-/de-)crements a mouse-pointed PREM

- Ostinato Boxes (◘) To 'freeze' the rhythm (& pitch?) within the stream's metric cycle
  - rhythm ostinato — Ö or key (↑Help Box) to freeze/free the rhythm or to free the pitch
  - pitch+rhythm ostinato — [Shift]+Ö or +key (↑Help Box) to freeze/free rhythm & pitch

- Explicit MIDI-Out Box (■) Holds typed-in MIDI code exportable by keys (↑Help Box)

- MIDI Output Monitor Shows MIDI code being exported; distinguishes L,M & R

- **PREPROC** Box (✎) 3 programs in the like-named ↓folder, Öed to create ↓**.IDP**/**.HRM** files

Files — Data or programs

- formats
  - binary file — Compressed data file, processable only by appropriate programs
  - text file — A more verbose ASCII file, read- & writeable by normal text editors
  - "unlimited" size — Term used to indicate constraints not imposed by AUTOBUSK
- extensions
  - **.CTS** — Text file containing one scale per line, expressed in cent values
  - **.HRM** — Compiled binary scale file listing scale interval sizes, ↓ratios & ↓harmonicities
  - **.JST** — List of intervals constrained by choice of prime factors, powers &c.
  - **.MTR** — Text file containing one metre per line, expressed in terms of ↓stratification
  - **.IDP** — Compiled text file listing metric pulse indispensibilities
  - **.PRM** — Time-tagged ↓score text file containing 4 PREM attributes (5 elements) per line:
    - time-tag — Moment in seconds & milliseconds at which the line's event happens
    - stream — The symbol L, M, or R denoting the stream being treated
    - tag — One of 25 symbols [HIMOPRSW,0-9,"^sm@!#] for the event type (18 PREMs+7 extras)
    - value — The one-byte value [0-255] allotted to the event type
  - **.PRK** — Compressed form of **.PRM** for on-line reading by AUTOBUSK
  - **.α** — Text file containing user-defined basic high-level settings
  - **.Φ** — Binary file listing upto 99 ↑flushes, always updated on leaving the program
  - **.N** — Binary file containing the 756 encoded symbols in the program's internal font
  - **.MDK** — MIDI file in the format of the author's MIDIDESK package

Folders — 2 for programs, 3 for data

- **FLUSHES** — Containing upto 99 ↑flushes, hard-disk text file copies of the program's ↑flashes
- **PRKSCORE** — Containing ↑**.PRK** files, for on-line reading by AUTOBUSK
- **PRMSCORE** — Containing ↑**.PRM** files for processing by the 9 ↓**PRMPROC** attendant programs
- **PREPROC** — Containing 4 programs for creating .HRM & .IDP files for AUTOBUSK:
  - **ASC** — Imported ASCII text editor for entering &/or editing **.CTS** & **.MTR** files
    - **TEMPUS** — ASCII text-editor recommended for use as **ASC** & ↓**EDIT**
  - **HRM** — Scale compiling program, converting **.CTS** into **.HRM** files
    - octavic equivalence — The general belief that pitches an octave apart are 'the same'
    - ratio — Pair of numbers representing a frequency relationship, e.g. 2:3 for the fifth
    - components — Numerator & denominator, i.e. the number pair in a ratio
    - 3-/5- limit — A ratio's largest prime number factor, here 3 & 5, respectively
    - rationalisation — Ratio close to a given interval serving as its harmonic explanation
    - tuning alternatives — Ratios vying as harmonic explanations for a nearby interval
    - tuning tolerance — simply: Permitted gap between an interval & its rationalisation
    - harmonicity — Function H of a ratio reflecting its harmonic 'strength', e.g. 2:3 at +27%
    - specific harmonicity — Reciprocal of the sum of intra-scalar inharmonicities (1/H)
    - scale stretch — Microtonal scales must be thus disguised as 12-tone-tempered output
    - copy — Rationalised scale of limited range can be copied to form an extended one
    - indigestibility — Function ξ of a number, e.g. 22, 11, 9, 4, 30 & 6 for the nos. 13 to 18
    - sure/filed harmonicity — The harmonicity threshold ensured/stored by a run of **HRM**
    - harmonic intensity — The absolute (positive) value of the harmonicity
  - **JST** — Branching off from **HRM**, a program for forming a vocabulary list of interval ratios
    - header — Showing the constraints applied: largest prime & power, indigestibility &c.
  - **IDP** — Metre compiling program, for converting **.MTR** into **.IDP** files
    - indispensability — Function ψ of a pulse, reflecting its 'strength' in the metre it is in
    - stratification — Continued product of primes for defining a metre, e.g. 2×2×3 for $^{12}/_{16}$
- **PRMPROC** — Containing 9 programs for processing **.PRM** score files:
  - **EDIT** — Imported ASCII text editor for entering &/or editing **.PRM** & other files
  - **FILL** — Program for PREM-interpolation between 2**.PRM** flush files to form a score file
  - **FUSE** — Program for synchronizing two or more **.PRM** score files
  - **JOIN** — Program for concatenating two or more **.PRM** score files
  - **PACK** — Program for compressing a **.PRM** into a **.PRK** score file
  - **PART** — Program for splitting a **.PRM** score file into two, according to given constraints
  - **PICK** — Program for curtailing the file menues of other programs by 'hiding' files
    - chosen/hidden list — 2 lists, files currently 'seen'/'unseen' by **PRMPROC** programs
  - **TIDY** — Program for eliminating redundancies in a **.PRM** score file
  - **VARY** — Program for the algorithmic alteration of PREM attributes
    - shift/spread/slide — 3 modes of alteration of PREM-values & of time-tags
    - Material Replacement Box — To swap streams, scales, metres, MIDI-channels
    - Parameter Transformation Box — To shift/spread/slide PREM-values & time-tags
    - Time Function, Counters — To display the progress of time & of the processing

» button 12, 48
i button 12, 44, 48
3-limit 26, 51
5-limit 25-26, 51
.N 1, 3, 46, 50
α button 11-12, 17, 48
.α 1, 11-12, 14, 44, 48, 50
Alformation 44
Allocation Box 14, 16, 49
Alternative Byte Box 16, 45, 50
alternative/relevant byte 16, 50
ASC 1-3, 10, 18-19, 30, 51
ASCII 2-3, 15, 18, 46, 47, 50-51
attenuation 8, 44, 48
binary file 1-2, 11, 42, 44-46, 50
busy mouse 6, 47
chordal weight 8, 44, 48
chosen list/hidden list 37, 51
clock info 16, 49
clock out 16, 49
crement 6, 39, 40, 47, 50
.CTS 2-3, 18-20, 22, 25, 42, 50-51
denominator 25, 27, 42, 51
dynamics 8, 16, 44, 48
EDIT 1, 3, 30, 51
event length 8, 44, 48
eventfulness 7-9, 44, 48
exit now 17, 49
Explicit MIDI-Out Box 10-11, 45, 49-50
file-pick mouse 6, 47
FILL 3, 30
flash 1-2, 4, 11-12, 14, 17, 45, 48, 51
flush 2-3, 12, 30-31, 44, 48, 50-51
FLUSHES 1, 12, 30-31
FUSE 1, 3, 32, 35
Gauss 21
harmonic intensity 28, 51
harmoniclarity 8, 44, 48
Header Box 13-15, 46-47, 49
Help Box 5, 7, 9-10, 13, 16, 49-50
hexadecimal code 9-10, 14, 47
Hexagon 5-7, 50
hidden list/chosen list 37, 51
.HRM 2-4, 10, 12, 22-23, 25, 37, 42, 48-51
.IDP 2-3, 10, 12, 26, 37, 43, 48-51
inactive mouse 6, 47
indigestibility 26-27, 42, 51
indispensability 8, 25-26, 50-51
instruction line 6, 9, 11, 28, 30-35, 37-39, 41, 47
interrupted 10-11, 16, 28, 47
JOIN 1, 3, 33, 51
.JST 1-2, 20, 27, 29, 42, 50
keycommand 15, 49
latch-on mouse 6, 16, 47
Lostinato 10, 49
material 1, 7-8, 42, 48,51
Material Replacement Box 40, 51
.MDK 2, 11, 46, 48,50
melody scope 8, 44, 48
Metre Display Box 7, 49
metre number 7, 43, 48
metriclarity 5, 7-8, 43, 48
MIDI channel 8-10, 14, 40, 43, 45, 47-49, 51
MIDI Input Monitor 13, 16, 49
MIDI Output Monitor 9-11, 15-16, 49-50
MIDI range 15, 49
Mostinato 10, 49
.MTR 2-3, 18, 25-26, 43, 50-51
nibble 13-16, 47, 49

numerator 25, 27, 42, 51
octave number 8, 48
octavic equivalence 19, 51
one-click mouse 6, 9-10, 47-48
OPEN 13, 15-16, 49
Ostinato Boxes 10, 50
Ö_ 3-4, 6, 11-12, 20, 28, 30-35, 37-40, 47, 49-50
_Ö 3, 6, 12, 28, 31-35, 40
PACK 1, 3, 34, 51
Parameter Transformation Box 40
PART 1, 3, 35-36, 44, 47-48, 51
performance tolerance 17, 47
Φ button 11-12, 45, 48
.Φ 1-2, 4, 11-12, 17, 45, 50
PICK 1, 3, 37, 51
pick/spurn 3-4, 10, 47, 49
pitch centre 8, 44, 48
pitch range 8, 27, 43, 48
pitch+rhythm ostinato 10, 50
poising the mouse 6-7, 12, 35, 40, 47
plus-minus mouse 6, 15
PREM 1-12, 14-16, 30, 35, 39-40, 43-45, 47-51
PREM deactivation 9
PREPROC 1-2, 10, 19-20, 37, 51
.PRK 2-4, 7, 9-10, 12, 34, 37, 44, 47-51
PRKSCORE 1, 51
.PRM 2-3, 12, 30, 32-35, 37-40, 43-44, 48-51
PRMPROC 1, 3, 17, 30, 37, 44, 51
PRMSCORE 1, 41, 51
pulse length 7, 17, 43, 48
pulselatch 7, 49
R button 11, 48
rationalisation 19, 21, 23, 51
relevant/alternative byte 16, 49-50
Remap Box 13, 16, 50
rhythm ostinato 10, 43, 49-50
Rostinato 10, 49
Route 1, 8, 16, 48
S button 11, 48
scale number 7, 43, 48
Scale Display Box 7, 49
Selector Box 3-4, 10, 12, 47-49
send all3 10, 49
send Lstr 10, 49
send Mstr 10, 49
send Rstr 10, 49
shift/spread/slide 39-40, 51
show midO 9, 49
show ping 9, 49
sound/controller 8, 14, 35, 48
specific harmonicity 22-23, 51
speed of the computer 17, 47
spurn/pick 3-4, 10, 47, 49
start time 12, 47-48
stopped at 9, 16, 47-48
stratification 3, 25, 48-51
stream 1, 6-11, 14-16, 35-36, 40, 43-45, 48-51
Stretch 23-25, 51
T button 11, 48
TEMPUS 2-3, 18, 51
text file 1, 42-44, 46, 50-51
0 button 11, 16, 48
TIDY 1, 3, 38, 51
time counter box 34, 36, 38
time indicator 9, 47
timelatch 7, 49
time-tag 1-3, 31-33, 39-41, 43, 50-51
tonic pitch 8, 44, 48
tuning alternatives 20, 51
tuning tolerance 21, 51
USED 13, 49
unlimited 42-44, 50
VARY 1, 3, 39, 41, 51
wipe midI 13, 49

Bibliography

Clarence Barlow
- *Bus Journey to Parametron*
Feedback Papers 21-23 (131 pages), Feedback Studio Cologne 1980

Klarenz Barlow
- *Kommentar zu „Bus Journey“*
[Ger.] Neuland 1 (S.114-118) 1980, H.Henck Deinstedt

Clarence Barlow
- *Two Essays on Theory* Computer Music Journal Vol.11 No.1 (pp.44-60),
MIT Press Cambridge Massachussetts 1987

Georg Hajdu
- *Harmonische Energie - Motor bei der Evolution "relevanter" Tonsysteme*
[Ger.] Feedback Papers 35, (S.1-25), Feedback Studio Köln 1989

Georg Hajdu
- *Die Domestifikation des Zufalls*
in Klarenz Barlows Computerprogramm AUTOBUSK [Ger.]
Hausarbeit, Musikhochschule Köln 1990, Georg Hajdu Münster

Klarenz Barlow
- *Zu „Orchideæ Ordinariæ or The Twelfth Root Of Truth“*
[Ger.] Feedback Papers 36, (26 Seiten - insbes. S.16), Feedback Studio Köln 1990

Klarenz Barlow
- *Harmonik und Quantität*
[Ger.] Ex Machina 90: Das Buch zum Festival (S.47-55),
Folkwang-Hochschule Essen 1990

Clarence Barlow
- *AUTOBUSK,*
an Algorithmic Real-Time Pitch & Rhythm Improvisation Programme
Proceedings of the XVIth International Computer Music Conference (pp.166-168)
Glasgow 1990

Klarenz Barlow
- *„. . .eine arithmetische Übung der Seele. . .“,*
[Ger.] Musikinformatik und Medientechnik, Bericht Nr.34 (S.11-32),
Musikwissenschaftliches Institut, Universität Mainz 1998

Clarence Barlow
- *On the Quantification of Harmony and Metre*
The Ratio Book (349 pages), ed. C.Barlow (pp.2-23)
Royal Conservatory The Hague (in preparation)

Claudia Maria Zey
- *Çoğluotobüsişletmesi und AUTOBUSK -*
zum stochastischen Kompositionskomplex von Klarenz Barlow (1971-2000)
[Ger.] Dissertation (ca. 500 Seiten), Universität des Saarlandes (in Vorbereitung)