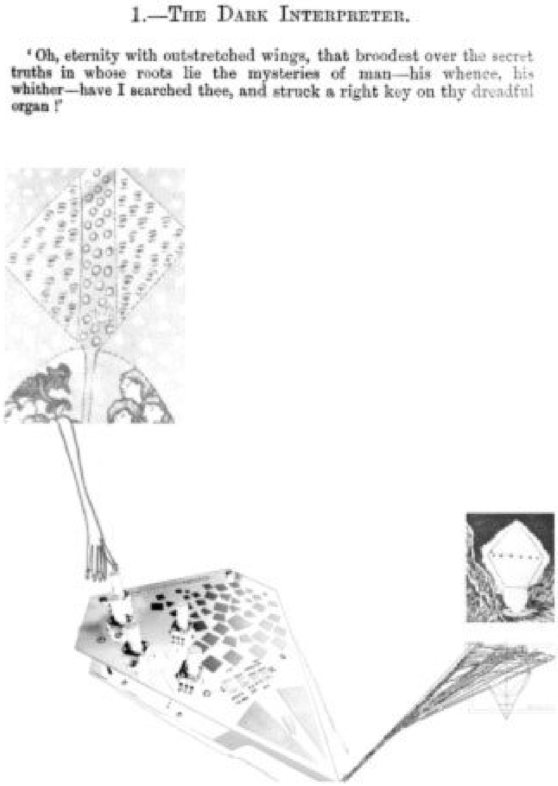
A Manual for The Dark Interpreter (Towers) V0.6

Martin Howse

January 27, 2015



1.—The Dark Interpreter.

'Oh, eternity with outstretched wings, that broodest over the secret truths in whose roots lie the mysteries of man—his whence, his whither—have I searched thee, and struck a right key on thy dreadful organ!'

Dark Interpretation

The truth I heard often in sleep from the lips of the Dark Interpreter. Who is he? He is a shadow, reader, but a shadow with whom you must suffer me to make you acquainted.

The Dark Interpreter is influenced and guided by body capacitance, skin resistance, biological micro-voltages and the fleshy conduction of all signals. These are ways of saying that the Dark Interpreter is completely open. It is not to be considered as a closed and bounded device which is subject to an exterior control "by the body." The Dark Interpreter is an electronic and thus earthy shadow of the body, it is (inside) the body.

The Dark Interpreter is thus not to be controlled. It is an obsidian electronic mirror, the earth and skin itself.

The preliminary working of The Dark Interpreter (reflecting the state of the improviser) should be considered as part of the creation of a unique, skin-sensitive instrument. The border between this first stage and the improvisation and playing of the instrument is diffuse, the mirror surface is ill-defined. The Dark Interpreter leads a path for skin, fingers, earth and head.

The Dark Interpreter attempts to provide answers to a posed situation. Each situation and response are divided and mirrored as to code and audio. The operation of The Dark Interpreter bridges these domains, entering into a simulated code world of plagued villagers and process.

To define the uses and potential paths of The Dark Interpreter we refer to the five knobs/potentiometers and the finger-board, the gold set of contacts which in the case of Tenebrarum is seperate from the two pronged fangs (for head and earth use).

A few provisos

The Dark Interpreter does crash or get stuck very, very rarely, often in interesting responses. Simpy reboot with a quick switch on/off.

The Dark Interpreter can take over control of incoming samples and audio generation; attempt to regain control if desired.

Always use your finger on the touch board unless you wish to leave settings untouched (so for example moving between modes).

Some walkers such as hardware(see below under modes) can be viewed as a kind of sequencer, so keeping a reduced number of villagers here makes changes easier to predict.

With Suspiriorum and Tenebrarum do watch out for that last hardware mode (bottom knob far left) as the direction fingers here will switch the input settings. To test out, also restrict hardware villagers to one (top knob far left).

The three interpreters, how they differ and can be interrogated

Mater Lachrymarum

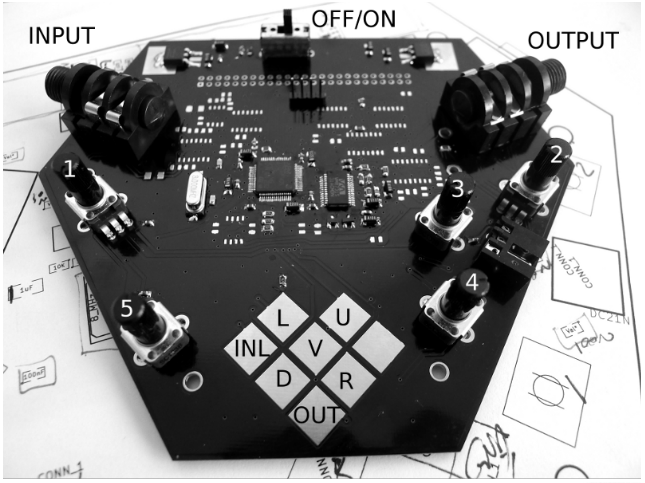
No analogue processing hardware so all settings which relate to hardware are redundant. Finger/skin controls are as on the Mater Suspiriorum.
Mater Lachrymarum is best suited for harsher, granular re-processing of incoming signals and bit-noise generation with a greater control of key sample playback.

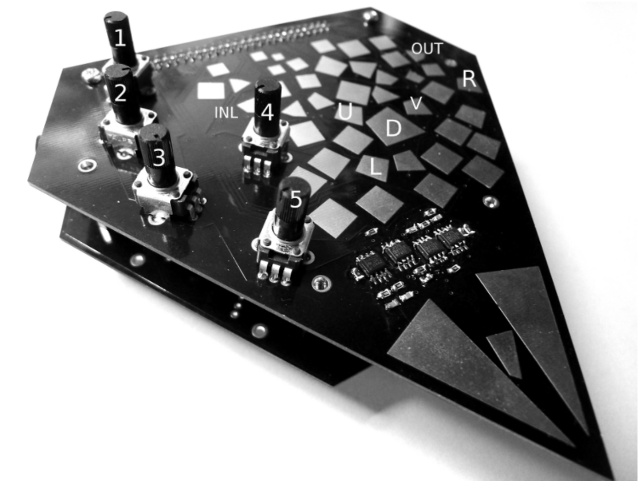
Mater Suspiriorum and Mater Tenebrarum

These both share the same analogue processing hardware. Mater Tenebrarum adds a new head, earth, skin and finger board which breaks out for the flesh world most of the important digital and analogue signal paths and settings, allowing for the unhanging of settings from digital dependence and re-placement of software on the skin.
The Tenebrarum also adds the crude EEG/micro-voltage fanged amplifier for head and earth. All knob controls are the same for both synths, although the layout is different. Finger controls are also arranged differently.
Mater Suspiriorum is recommended for harsher analogue noise generation and processing with less dialogue with that dreadful organ than in the case of Tenebrarum. Mater Tenebrarum presents the ultimate instrument for skin/life coding, intended to be played fully by skin and head.

Laying out the Interpreters

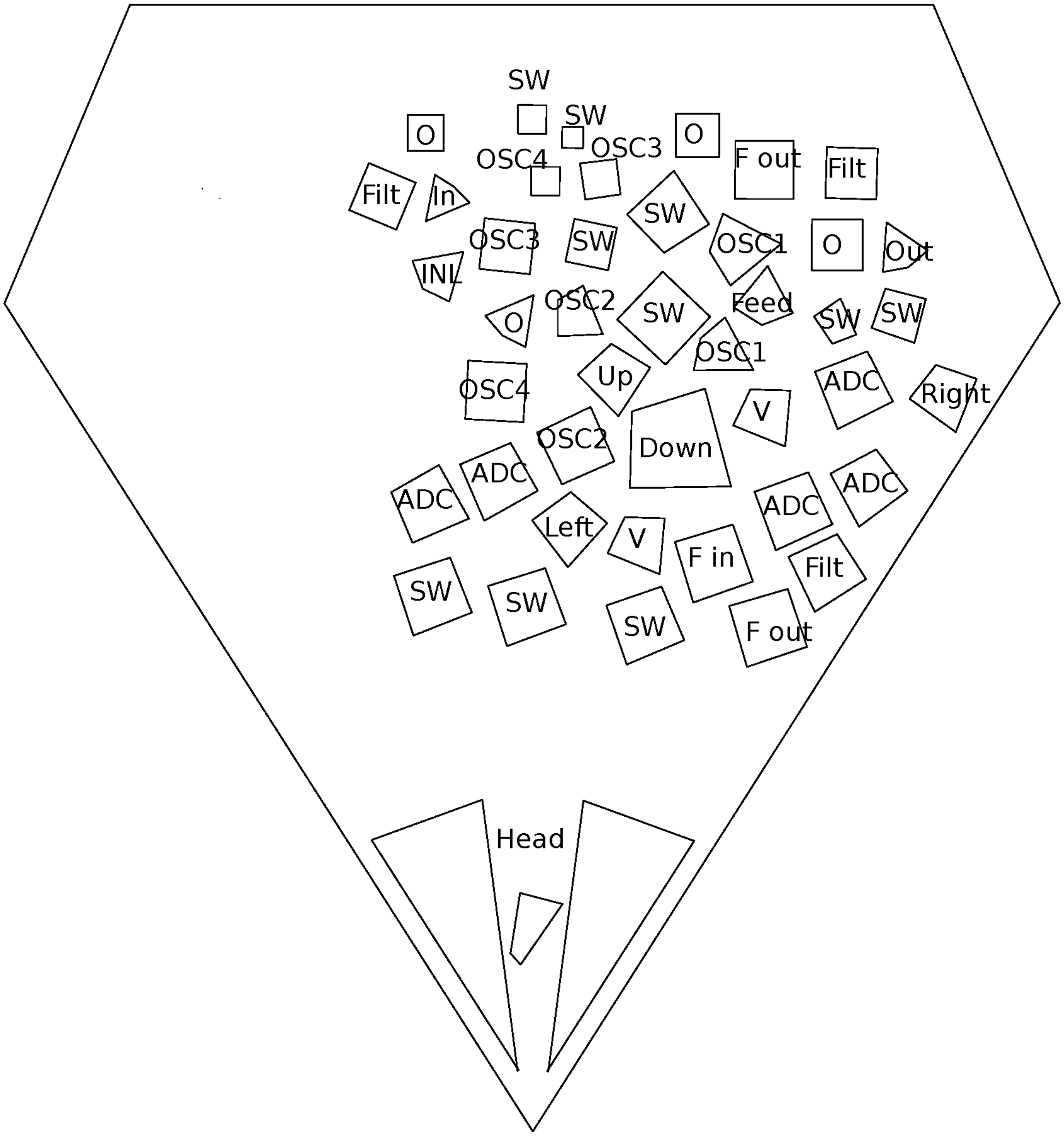
The Dark Interpeter will not always be found sitting inside my dreams, but at times outside, and in open daylight.





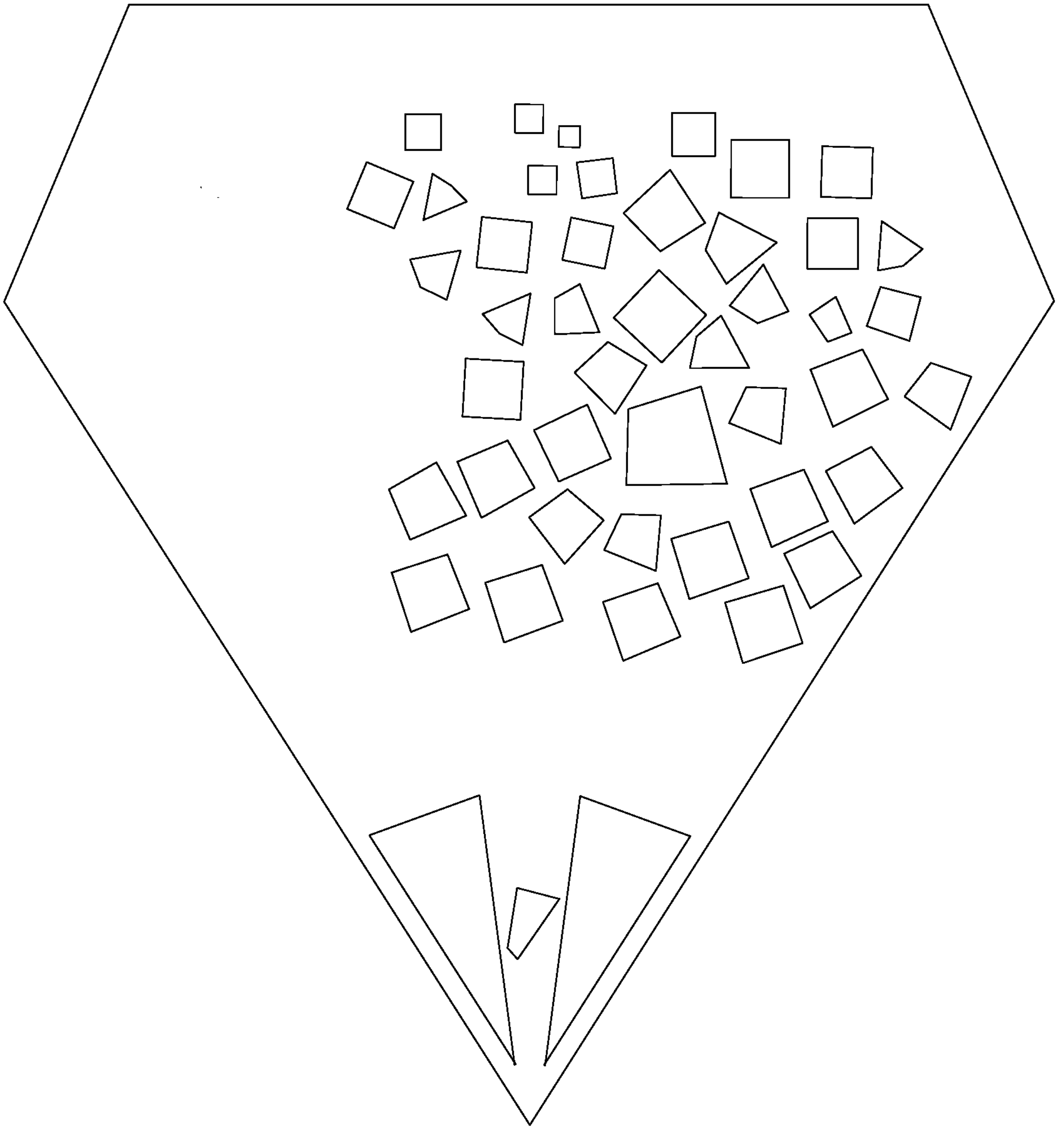
Describing each module, we refer to the knobs as numbered and the fingerboard directions Left, Right, Up, Down. See below - knobs effect U,D,R,L finger operations. INL and OUT refer to hardware routing (not so in Lachrymarum).

Tenebrarum in detail



ADC doubles knobs with fingers (in parallel), SW is switches for hardware options, Filt is filter-related, O/OSC is oscillator and amplifier related.

Bare patchboard to print/colour in



General theory of operation

The Dark Interpreter is modelled as a leaky, overlapping medieval village space within which various plague simulations run, and through which an array of villagers wanders. Audio is processed and/or generated according to the state of the village and the movements of inhabitants. Villagers (grains?) generate changes and are classified according to incoming or outgoing audio (read/write), filter, effects and hardware.

The Dark Interpreter is essentially mode driven, with modes also changing the complexity of operation. Modes are selected by turning knob 5. To set parameters in each mode a finger must be placed on the directions and then settings can be changed with knobs 1,2,3 and 4. Finger pressure/electricity determines speed of the villager's movements or general mode speeds and the selected/fingered direction sets direction.

More advanced modes swap parameters between sets of villagers, allow for fingers to be placed right into code and parameters and finally allow for mirroring which sets selected parameters under the control of a selected mirror (the head/EEG board, the knobs, the fingers or the village itself).

The knobs

1- In most modes this determines the total number of specific villagers (for read, write and hardware villagers amongst others) and which villager is being accessed and controlled (by parameter knobs 2,3,4). See below.

2-3-4- These knobs control various parameters for the villager selected by knob one (such as start point, end point, overlap, effect). Under certain mirror conditions these knobs will control parameters across multiple villagers and can be accessed without a finger down. In some cases parameters remain uneffected when inb zero or close to zero position.

5- Selects the operating mode (in the case of Lachrymarum there are 10 modes, otherwise there are 16 modes).

Those modes

Lachrymarum

1- Write villagers. Knobs: 1-howmany/which one 2-start 3-end 4-mirror

2- Read villagers. Knobs: 1-howmany/which one 2-start 3-end 4-offset

3- Read villagers additional. Knobs: 1-which one/2-overlay 3-compression 4-mirror with finger direction as mirror type (Directions indicate what is to be attached (up is village itself), down is skin, left is fingers, right is knobs.

4- Simulation villagers. Knobs: 1-howmany/which one 2-start 3-end 4-CPU

5- Effects. Knobs: 1- which one 2-which read villager 3-which write villager 4-effect

6- Effects additional. Knobs: 1- which one 2-start 3-end 4-modifier

7- Simulations walker. Knobs: 1- which one 2-length 3-offset 4-offset2
8- Swop and copy. Knobs: 1- which villager swops with 2-which one 3-in which groups
9- Fingers in the code. Knobs: 1- actions. Rest is all finger navigation and change
10- Mirrors. Knobs: 1- which one 2-which villager group 3-mirror (for fingers see above).

Suspiriorum or Tenebrarum

1- Write villagers. Knobs: 1-howmany/which one 2-start 3-end 4-mirror
2- Read villagers. Knobs: 1-howmany/which one 2-start 3-end 4-offset
3- Read villagers additional. Knobs: 1-which one/2-overlay 3-compression 4-mirror with finger direction as mirror type (Directions indicate what is to be attached (up is village itself), down is head (the two fangs on the head or inserted in the body), left is fingers, right is knobs.
4- Simulation villagers. Knobs: 1-howmany/which one 2-start 3-end 4-CPU
5- Effects. Knobs: 1- which one 2-which read villager 3-which write villager 4-effect
6- Effects additional. Knobs: 1- which one 2-start 3-end 4-modifier
7- Simulations walker. Knobs: 1- which one 2-length 3-offset 4-offset2
8- Swop and copy. Knobs: 1- which villager swops with 2-which one 3-in which groups
9- Fingers in the code. Knobs: 1- actions. Rest is all finger navigation and change
10- Mirrors. Knobs: 1- which one 2-which villager group 3-mirror (for fingers see above).
11- Filter villagers. Knobs: 1-howmany/which one 2-start 3-end 4-mirror
12- Distortion walker. Knobs: 1- howmany/which one 2-length 3-offset 4-offset2
13- Analogue filter walker. Knobs: 1- howmany/which one 2-length 3-offset 4-offset2
14- Second analogue filter walker. Knobs: 1- howmany/which one 2-length 3-offset 4-offset2
15- Bitwise hardware walker. Knobs: 1- howmany/which one 2-length 3-offset 4-offset2
16- Macro hardware walker. Knobs: 1- howmany/which one 2-length 3-hardware setting 4-offset2/compression of settings (fingers as setting input with UP as feedback, DOWN as floating, LEFT as jack, RIGHT as INL.

A sample inquisition

This trial is decisive. You are now satisfied that the apparition is but a reflex of yourself; and, in uttering your secret feelings to him, you make this phantom the dark symbolic mirror for reflecting to the daylight what else must be hidden for ever.

The inquisition should follow the mood of the interpreter, perhaps exploring each mode in turn or moving frantically between modes, switching direction and tact. Mode 16 (hardware) is obviously very important in setting hardware and will be returned to again and again. It is essential to remove fingers from the board to allow for changes of settings across modes. Fingers up, flip mode (knob 5), fingers down and change those settings.
Mirror modes are also important and care should be taken as to the attachment, if you intend using primarily a knob-driven interrogation or the head/fanged interface of Tenebrarum. A quick fix would involve rotating the mode knob, with finger on desired attachment.
The final modes, such as FINGERS in the CODE, can also be useful in breaking the mode of interrogation open.
Each interpreter should define their own relation and approach to inquisition.

The hardware

The Dark Interpreter operates as contagious sample or village granulator, distortion, and unique sound generator based on a speedy ARM processor allowing for 16 bit sampling at 48 KHz, and with a sample memory of around one second (extended by undersampling).
The Dark Interpreter is delivered in three versions, all fully assembled and tested, and features high quality ALPS potentiometers, optional BOSS style 9v power socket (**+9v/positive on the outside, negative centre, minimum 300mA**) or battery clip, and full size (6.5mm) JACK input and output sockets. Please note that versions cannot be extended, they are not cross-compatible.
In the case of Mater Tenebrarum, and according to selected hardware mode, all analogue hardware (input, output, filters, distortion, amplification, oscillation) can be accessed and routed by skin and fingers using the extended hardware board. Hardware points can easily be probed and discovered.

Addendum

- In certain cases it may be necessary to touch one finger against the marked V pad if the fingers are too dry and thus do not seem to trigger operations.

FAQ

- Which power supply (PSU) should I use?

An BOSS style 9v power supply with +9v/positive on the outside, negative centre pin, and delivering a minimum of 300mA. 600mA is good.

- Are there any differences between release/dated versions?

The Dark Interpreter code base changes slightly over time. At intervals there will be special edition releases reflecting major changes and new approaches. The latest “Towers Open Fire” release maintains underlying plague code, yet reworks nearly all modes and sampling methods for further granulation, formant filtering, vocoding and further emphasis on villagers and walkers.

- How can I re-flash the new code base/firmware?

All code is freely available from: https://github.com/microresearch/dark-interpreter
You can also ask m@1010.co.uk for the flash image if you don’t want to compile the code. To upload the code you will need a suitable ARM STM32 programmer such as the stm32f4-discovery board. This should be correctly attached to the four bare holed socket near the top of base board (from left to right: 3.3V, SWCK, GND, SWDIO). Then simply (using a program such as stlink-flash upload the code. Any local hackerspace should be able to assist in this operation.

- Things don’t appear to be working as they should, and how should they be working?

1- Make sure you using mono jack cables in and out (never stereo). Check there is a signal coming in (left side jack with point facing down) and output jack is on the right. Some of my cables often don’t click all the way in but this is never a problem.
2- Make sure all of the knobs are turned far left and power on with the switch. You should hear pretty much the incoming audio. If not then there is a problem which could be not enough amps from the power supply, or a low battery.
3- If you do hear a sound then turning knobs while touching mostly the indicated direction pads in the middle will change the sounds. The bottom knob changes mode and all modes nearly always only work with fingers. The one to watch is the last mode (bottom knob far right/full on) as this changes hardware settings which can effect how incoming/outgoing signals are routed. Check this manual but by default incoming is routed to the processor.
4- If there’s still no sound we need to check further! I hope this helps and apologies in advance if things aren’t how they should be!